Constrained Optimization via Particle Evolutionary Swarm Optimization Algorithm (PESO)

Angel E. Muñoz Zavala
Center for Research in Mathematics (CIMAT)
Guanajuato, Gto., México
aemz@cimat.mx

Arturo Hernández Aguirre
Center for Research in Mathematics (CIMAT)
Guanajuato, Gto., México
artha@cimat.mx

Enrique R. Villa Diharce
Center for Research in Mathematics (CIMAT)
Guanajuato, Gto., México
villadi@cimat.mx

ABSTRACT

We introduce the PESO (Particle Evolutionary Swarm Optimization) algorithm for solving single objective constrained optimization problems. PESO algorithm proposes two new perturbation operators: "c-perturbation" and "m-perturba tion". The goal of these operators is to fight premature convergence and poor diversity issues observed in Particle Swarm Optimization (PSO) implementations. Constraint handling is based on simple feasibility rules. PESO is compared with respect to a highly competitive technique representative of the state-of-the-art in the area using a well-known benchmark for evolutionary constrained optimization. PESO matches most results and outperforms other PSO algorithms.

Categories and Subject Descriptors

J.2 [**Computer Applications**]: Physical Sciences and Engineering—*Engineering, Mathematics and statistics*; G.1.6 [**Numerical Analysis**]: Optimization—*Stochastic programming, Constrained optimization*

General Terms

Algorithms

Keywords

Particle Swarm Optimization, Constrained Optimization, Hybrid-PSO

1. INTRODUCTION

In PSO algorithms, the social behavior of individuals is rewarded by the best member in the flock. The credibility on the best regulates how fast the flock is going to follow him, thus exploration is improved but convergence is reduced if the flock slowly follows the best. Too much credibility on the best quickly concentrates the flock on small areas, reducing exploration but accelerating convergence time. In a constrained search space, the trade-off becomes harder to balance since the constraint handling mechanism may increase the pressure on the flock to follow the best member which is trying to reach the feasible region, or the function optimum if already inside that region.

Several authors have noted the speed-diversity trade-off in PSO algorithms [9]. Also, many noted the need to maintain population diversity when the design problem includes constraints [13]. Although several approaches have been proposed to handle constraints, recent results indicate that simple "feasibility and dominance" (FAD) rules handle them very well. FAD rules have been enhanced with (extra) mechanisms to keep diversity, therefore, improving exploration and the consequent experimental results. Nonetheless, one important conclusion recently reached by Mezura [10], is that the constraint handling mechanism must be tied to the search mechanism, even more, tied to the way the operators explore the search space. For instance, several algorithms combining FAD rules and selection based on Pareto ranking have frequently been defeated by those combining FAD rules and selection based on Pareto dominance [5].

The proposal conveyed by this paper is the combination of a constraint handling mechanism (FDA rules) and PSO algorithm enhanced with perturbation operators. The goal is to avoid the explicit controls and extra processing needed to keep diversity [5]. This paper introduces a new approach called PESO, which is based on the swarm algorithm originally proposed in 1995 by Kennedy and Eberhart: Particle Swarm Optimization (PSO) [7]. Our approach includes constraint handling and selection mechanism based on feasibility rules; a ring topology organization that keeps track of the local best; and two perturbation operators aimed to keep diversity and exploration.

The remainder of this paper is organized as follows. In Section 2, we introduce the problem of interest. Section 3 presents recent approaches to handle constraints in PSO algorithms. Section 4 introduces our approach and provides details of the algorithm. In Section 5, a benchmark of 13 test functions is listed. Section 7 provides a comparison of results with respect to state-of-the-art algorithms for constrained optimization. Finally, our conclusion and future work are provided in Section 8.

GECCO'05, June 25–29, 2005, Washington, DC, USA.

2. PROBLEM STATEMENT

We are interested in the general nonlinear programming problem in which we want to:

$$\text{Find } \vec{x} \text{ which optimizes } f(\vec{x}) \tag{1}$$

subject to:

$$g_i(\vec{x}) \leq 0, \quad i = 1, \ldots, n \tag{2}$$

$$h_j(\vec{x}) = 0, \quad j = 1, \ldots, p \tag{3}$$

where $\vec{x}$ is the vector of solutions $\vec{x} = [x_1, x_2, \ldots, x_r]^T$, n is the number of inequality constraints and p is the number of equality constraints (in both cases, constraints could be linear or non-linear). For an inequality constraint that satisfies $g_i(\vec{x}) = 0$, then we will say that is active at $\vec{x}$. All equality constraints h_j (regardless of the value of $\vec{x}$ used) are considered active at all points of $\mathcal{F}$ ($\mathcal{F}$ = feasible region).

3. RELATED WORK

Lately, significant effort and progress has been reported in the literature as researchers figure out ways to enhance the PSO algorithm with a constraint handling mechanism. Coath and Halgamuge [1], proposed the "feasible solutions method" (FSA), and the "penalty function method" (PFA) for constraint handling. FSA requires initialization of all particles inside the feasible space; they reported this goal is hard to achieve for some problems. FPA requires careful fine tuning of the penalty function parameters as to discourage premature convergence. Zhang and Xie [14], proposed DEPSO, a hybrid approach that makes use of a reproduction operator similar to that used in differential evolution. In DEPSO this operator is only applied to *pbest* but, in PESO a similar perturbation is added to every particle. Toscano and Coello [13] also perturb all particles but accordingly to a probability value that varies with the generation number (as proposed by Fieldsend and Singh [3]). We compare our results against their recently published experiments.

4. THE PESO ALGORITHM

The Particle Swarm Optimization (PSO) algorithm is a population - based search algorithm based on the simulation of the social behavior of birds within a flock. In PSO, individuals, referred to as particles, are "flown" through a hyperdimensional search space. PSO is a kind of symbiotic cooperative algorithm, because the changes to the position of particles within the search space are based on the social-psychological tendency of individuals to emulate the success of other individuals.

The feature that drives PSO is social interaction. Individuals (particles) within the swarm learn from each other, and based on this shared knowledge tend to become more similar to their "better" neighbors. A social structure in PSO is determined through the formation of neighborhoods. These neighborhoods are determined through labels attached to every particle in the flock (so it is not a topological concept). Thus, the social interaction is modeled by spreading the influence of a "global best" all over the flock as well as neighborhoods are influenced by the best neighbor and their own past experience.

Figure 1 shows the neighborhood structures that have been proposed and studied [6]. Our approach, PESO, adopts

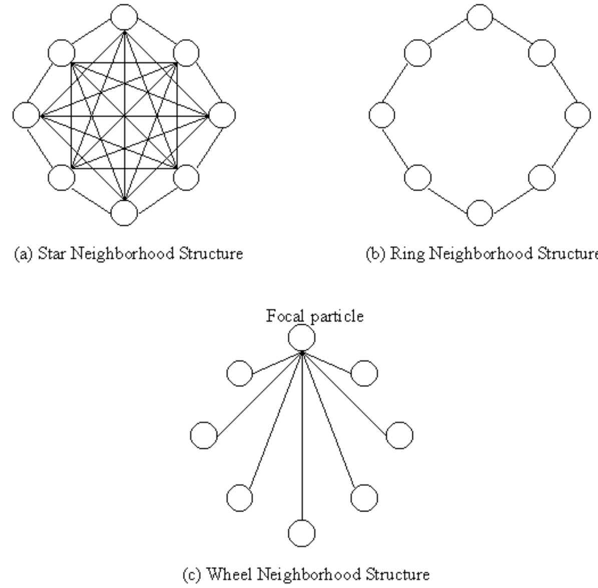

Figure 1: Neighborhood structures for PSO

the **ring** topology. In the ring organization, each particle communicates with its n immediate neighbors. For instance, when $n = 2$, a particle communicates with its immediately adjacent neighbors as illustrated in Figure 1(b). The neighborhood is determined through an index label assigned to all individuals. This version of the PSO algorithm is referred to as *lbest* (LocalBest). It should be clear that the ring neighborhood structure properly represents the LocalBest organization. It has the advantage that a larger area of the search space is traversed, favoring search space exploration (although convergence has been reported slower) [2, 8].

PSO-LocalBest has been reported to excel over other topologies when the maximum velocity is restricted. PESO's experimental results with and without restricted velocity reached similar conclusions noted by Franken and Engelbretch [4](thence, PESO algorithm incorporates this feature). Figure 2 shows the standard PSO algorithm adopted by our approach. The pseudo-code of **LocalBest** function is shown in Figure 3.

P_0 = Rand(LI, LS)
F_0 = Fitness (P_0)
$PBest_0$= P_0
$FBest_0$= F_0
Do
  $LBest_i$ = **LocalBest** ($PBest_i$, $FBest_i$)
  S_i = **Speed** (S_i, P_i, $PBest_i$, $LBest_i$)
  $P_{i+1} = P_i + S_i$
  F_{i+1} = Fitness (P_{i+1})
  For $k = 0$ **To** n
    $< PBest_{i+1}[k], FBest_{i+1}[k] > =$
      Best ($FBest_i[k]$, $F_{i+1}[k]$)
  End For
End Do

Figure 2: Pseudo-code of *PSO* algorithm with local best

Constraint handling and selection mechanism are described by a single set of rules called "feasibility and dominance".

These rules are: 1) given two feasible particles, pick the one with better fitness value; 2) if both particles are infeasible, pick the particle with the lowest sum of constraint violation, and 3), given a pair of feasible and infeasible particles, the feasible particle wins. These rules are implemented by function **Best** in PESO's main algorithm, see Figure 5.

```
For k = 0 To n
  LBest_i[k] = Best(FBest_i[k − 1], FBest_i[k + 1])
End For
```

Figure 3: Pseudo-code of LocalBest($PBest_i$,$FBest_i$)

The *speed* vector drives the optimization process and reflects the socially exchanged information. Figure 4 shows the pseudo-code of **Speed** function, where $c1 = 0.1$, $c2 = 1$, and w is the inertia weight. The inertia weight controls the influence of previous velocities on the new velocity.

```
For k = 0 To n
  For j = 0 To d
    r1 = c1 * U(0, 1)
    r2 = c2 * U(0, 1)
    w = U(0.5, 1)
    S_i[k, j] = w * S_i[k, j] +
      r1 * (PBest_i[k, j] - P_i[k, j]) +
      r2 * (LBest_i[k, j] - Pbest_i[k, j])
  End For
End For
```

Figure 4: Pseudo-code of Speed(S_i, P_i, $PBest_i$, $LBest_i$)

4.1 Perturbation operators

PESO algorithm makes use of two perturbation operators to keep diversity and exploration. PESO has three stages: in first stage the standard PSO algorithm [8] is performed, then the perturbations are applied in the next two stages.

The main algorithm of PESO is shown in Figure 5.

The goal of the second stage is to add a perturbation in a way similar to the so called "reproduction operator" found in *differential evolution* algorithm. This perturbation, called C-Perturbation, is applied all over the flock to yield a set of temporal particles $Temp$. Each member of the $Temp$ set is compared with the corresponding (father) member of $PBest_{i+1}$, so the perturbed version replaces the father if it has a better fitness value. Figure 6 shows the pseudo-code of the **C-Perturbation** operator.

In the third stage every vector is perturbed again so a particle could be deviated from its current direction as responding to external, maybe more promissory, stimuli. This perturbation is performed with some probability on each dimension of the particle vector, and can be explained as the addition of random values to each particle component. The perturbation, called M-Perturbation, is applied to every particle in the current population to yield a set of temporal particles $Temp$. Again, as for C-Perturbation, each member of $Temp$ is compared with its corresponding (father) member of the current population, and the better one wins. Figure 7 shows the pseudo-code of the **M-Perturbation** operator. The perturbation is performed with probability $p = 1/d$, where d is the dimension of the decision variable vector.

These perturbations, in differential evolution style, have the advantage of keeping the self-organization potential of the flock as no separate probability distribution needs to be computed [12]. Zhang and Xie also try to keep the self-organization potential of the flock by applying mutations (but only) to the particle best (in their DEPSO system) [14]. In PESO, the self-organization is not broken as the link between father and perturbed version is not lost. Thus, the perturbation can be applied to the entire flock. Note that these perturbations are suitable for real-valued function optimization.

```
P_0 = Rand(LI, LS)
F_0 = Fitness ( P_0 )
PBest_0 = P_0
Do
  LBest_i = LocalBest ( PBest_i, FBest_i )
  S_i = Speed ( S_i, P_i, PBest_i, LBest_i)
  P_{i+1} = P_i + S_i
  F_{i+1} = Fitness ( P_{i+1} )
  For k = 0 To n
    < PBest_{i+1}[k], FBest_{i+1}[k] > =
      Best ( FBest_i[k], F_{i+1}[k] )
  End For
  Temp = C − Perturbation (P_{i+1} )
  FTemp = Fitness ( Temp )
  For k = 0 To n
    < PBest_{i+1}[k], FBest_{i+1}[k] > =
      Best ( PBest_{i+1}[k] , FTemp[k] )
  End For
  Temp = M − Perturbation (P_{i+1} )
  FTemp = Fitness ( Temp )
  For k = 0 To n
    < PBest_{i+1}[k], FBest_{i+1}[k] > =
      Best ( PBest_{i+1}[k] , FTemp[k] )
  End For
  P_i = P_{i+1}
End Do
```

Figure 5: Main Algorithm of $PESO$

```
For k = 0 To n
  For j = 0 To d
    r = U(0, 1)
    p1 = Random(n)
    p2 = Random(n)
    p3 = Random(n)
    Temp[k, j] = P_{i+1}[p1, j] + r (P_{i+1}[p2, j] - P_{i+1}[p3, j])
  End For
End For
```

Figure 6: Pseudo-code of C − Perturbation(P_{i+1})

```
For k = 0 To n
  For j = 0 To d
    r = U(0, 1)
    If r ≤ 1/d Then
      Temp[k, j] = Rand(LI, LS)
    Else
      Temp[k, j] = P_{i+1}[k, j]
  End For
End For
```

Figure 7: Pseudo-code of M − Perturbation(P_{i+1})

5. EXPERIMENTS

Next, we show the contribution of the perturbation operators by means of the well known extended benchmark of Runarsson and Yao [11].

Four experiments were done:

- PSO: The standard PSO algoritmh using the parameters values described in Section 4.
- PSO-C: The PSO algorithm with the C-perturbation operator.
- PSO-M: The PSO algorithm with the M-perturbation operator.
- PESO: Our proposed PSO enhanced with C-perturbation and the M-perturbation operators.

A total of 50 particles were used by each generation, yielding $350,000$ objective function evaluations. The results of 30 runs for all benchmark functions are show in Tables 1, 2, 3, 4 and 5. The equality constraints were transformed to inequality constraints $|h_j| \leq \epsilon$, where $\epsilon = 1E - 3$.

Table 5: The *Standard Deviation* results of 30 runs for benchmark problems

TF	PSO	PSO-C	PSO-M	PESO
g01	0.6152	0	0.6102	0
g02	0.0924	0.0790	0.0556	0.0535
g03	0.1536	0.0448	0.1382	4.842E-06
g04	0	0	0	0
g05	24.7214	51.1914	76.6719	5.1575
g06	0	0	0	0
g07	8.1949	0.0185	2.6584	0.0696
g08	0	0	0	0
g09	0.0002	1.658E-07	0.0004	2.617E-07
g10	189.4323	51.4196	110.1945	59.3290
g11	2.563E-09	0	4.001E-07	0
g12	0.0022	0	0	0
g13	0.3625	0.2176	0.3298	0.2212

The contribution of the perturbation operators is noteworthy, they improve the performance of the standard PSO in 6 out of 13 benchmark functions. The main contribution comes from the C-perturbation, while the M-perturbation helps to maintain the exploration. A brief discussion about this topic is presented in the next section.

6. CONTRIBUTION OF THE PERTURBATION OPERATORS

As noted before, the proposed perturbation operators are designed to be cooperative with feasibility and dominance rules. The reader may stop here and wonder how much and how each stage of the PESO algorithm contributes towards the optimum? In Figure 8, the contribution of each stage of PESO on the **best particle** between two consecutive generations is shown at each generation. The plot stops around generation 800, when the current best function value is 0.770731. The series at the bottom represents the contribution of PSO (the first stage); the series at the middle shows the contribution of C-perturbation, and M-perturbation is shown in the upper series. The contribution is shown as a percentage. Notice that in most generations the improvement is not distributed among stages, but it comes from only one of them. The identity of the best particle is not important; contributions made by each stage over each particle in the flock were recorded, and the particle with best fitness value was recovered from the population. Reading from Figure 8, it is clear that M-perturbation was important at the first part of the exploration to get to the feasible region. In almost all generations, the best individual comes from either PSO stage, or C-perturbation stage. An analysis of how all the flock is affected at each stage is under way. Initial experimental results show that C-perturbation is important since PESO is prone to stagnation when it is not used; at the same time it is not the common engine propelling the best individual. Usually, the PSO stage refines the solutions, while C and M-perturbations help during exploration.

7. COMPARISON OF RESULT

7.1 PESO vs SR and Toscano's PSO

Stochastic Ranking, SR, [11], still is *the* state of the art algorithm for constrained optimization. The algorithms is a simple evolution strategy enhanced with a stochastic constraint handling mechanism. Toscano and Coello's PSO incorporates a constraint handling mechanism into the standard PSO algorithm.

In Table 6 the results of PESO and SR are shown. Note that PESO is better than SR in problem g06, g07, and g10, but SR is better in g02 and g13. Also, PESO average stays closer to the optimum and is better than SR in problems g06 and g10. SR average is better than PESO in problem g10. Nonetheless, the results are very competitive and comparable for all other problems. In brief, PESO improves the results of SR in problems g06, g07, and g10, whilst was unable to improve the result of SR on problems g02 and g13.

In Table 7 we show the results of PESO and Toscano and Coello's PSO (TCPSO). It can be seen that PESO is clearly better than TCPSO in problem g10, but TCPSO is better in problems g02 and g13. Although the best results for the rest of the problems are comparable, PESO outperforms TCPSO in the average results for problems g04, g05,g07,g10, and g13. TCPSO is not better than PESO for the other problems' average. Note that TCPSO worst values are really far from the average, a poor performance not shown by PESO.

7.2 PESO vs DEPSO

A few problems of the original 11-problems benchmark of Michalewicz were solved by DEPSO system [14]. Zhang's DEPSO incorporates a reproduction operator used in differential evolution into PSO algorithm.

Only the results shown in Table 8 are available. PESO outperforms DEPSO in problems g02, g07, g09, and g10, whilst other problems are comparative.

8. CONCLUSIONS AND FUTURE WORK

We have introduced PESO, a simple PSO algorithm enhanced with two perturbation operators for constrained optimization. These operators do not destroy the flock concept that inspired PSO algorithms, neither its self-organization capacity. PESO is simple and easy to implement, besides it does not need the adjustment of several control and mutation parameters. The results proved highly competitive on the benchmark. Future work includes the solution of other constrained problems reviewed by the specialized literature, as well as an extension for multiobjective optimization.

Acknowledgements

The authors of the paper acknowledge support for this work from Council of Science and Technology from the State of

Table 1: The *Best* results for 30 runs of the benchmark problems

TF		Optimal	PSO	PSO-C	PSO-M	PESO
g01	Min	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000
g02	Max	0.803619	0.669158	0.753791	0.746811	0.792608
g03	Max	1.000000	0.993930	1.005010	0.984102	1.005010
g04	Min	-30665.539	-30665.538672	-30665.538672	-30665.538672	-30665.538672
g05	Min	5126.4981	5126.484154	5126.484154	5126.484154	5126.484154
g06	Min	-6961.81388	-6961.813876	-6961.813876	-6961.813876	-6961.813876
g07	Min	24.306209	24.370153	24.307272	24.412799	24.306921
g08	Max	0.095825	0.095825	0.095825	0.095825	0.095825
g09	Min	680.630057	680.630057	680.630057	680.630057	680.630057
g10	Min	7049.3307	7049.380940	7049.562341	7049.500131	7049.459452
g11	Min	0.750000	0.749000	0.749000	0.749000	0.749000
g12	Max	1.000000	1.000000	1.000000	1.000000	1.000000
g13	Min	0.053950	0.085655	0.149703	0.078469	0.081498

Table 2: The *Mean* results of 30 runs for benchmark problems

TF		Optimal	PSO	PSO-C	PSO-M	PESO
g01	Min	-15.000000	-14.710417	-15.000000	-14.799964	-15.000000
g02	Max	0.803619	0.419960	0.603599	0.606307	0.721749
g03	Max	1.000000	0.764813	0.980979	0.704545	1.005006
g04	Min	-30665.539	-30665.538672	-30665.538672	-30665.538672	-30665.538672
g05	Min	5126.4981	5135.973344	5148.910347	5157.899867	5129.178298
g06	Min	-6961.81388	-6961.813876	-6961.813876	-6961.813876	-6961.813876
g07	Min	24.306209	32.407274	24.328212	27.618821	24.371253
g08	Max	0.095825	0.095825	0.095825	0.095825	0.095825
g09	Min	680.630057	680.630100	680.630057	680.630150	680.630057
g10	Min	7049.3307	7205.499975	7073.953425	7180.910960	7099.101385
g11	Min	0.750000	0.749000	0.749000	0.749000	0.749000
g12	Max	1.000000	0.998875	1.000000	1.000000	1.000000
g13	Min	0.053950	0.569358	0.720624	0.682109	0.626881

Table 3: The *Median* results of 30 runs for benchmark problems

TF		Optimal	PSO	PSO-C	PSO-M	PESO
g01	Min	-15.000000	-15.000000	-15.000000	-14.999999	-15.000000
g02	Max	0.803619	0.407260	0.576324	0.605312	0.731693
g03	Max	1.000000	0.782523	1.004989	0.715459	1.005008
g04	Min	-30665.539	-30665.538672	-30665.538672	-30665.538672	-30665.538672
g05	Min	5126.4981	5126.484153	5128.244729	5126.566275	5126.538302
g06	Min	-6961.81388	-6961.813876	-6961.813876	-6961.813876	-6961.813876
g07	Min	24.306209	28.836717	24.323802	27.010157	24.371253
g08	Max	0.095825	0.095825	0.095825	0.095825	0.095825
g09	Min	680.630057	680.630057	680.630057	680.630057	680.630057
g10	Min	7049.3307	7137.908533	7054.760909	7150.103784	7069.926219
g11	Min	0.750000	0.749000	0.749000	0.749000	0.749000
g12	Max	1.000000	1.000000	1.000000	1.000000	1.000000
g13	Min	0.053950	0.492288	0.742127	0.796763	0.631946

Table 4: The *Worst* results of 30 runs for benchmark problems

TF		Optimal	PSO	PSO-C	PSO-M	PESO
g01	Min	-15.000000	-13.000000	-15.000000	-12.999994	-15.000000
g02	Max	0.803619	0.299426	0.496639	0.506702	0.614135
g03	Max	1.000000	0.464009	0.874539	0.434060	1.004989
g04	Min	-30665.539	-30665.538672	-30665.538672	-30665.538672	-30665.538672
g05	Min	5126.4981	5249.824796	5351.313705	5497.454471	5148.859414
g06	Min	-6961.81388	-6961.813876	-6961.813876	-6961.813876	-6961.813876
g07	Min	24.306209	56.054769	24.377214	35.924678	24.593504
g08	Max	0.095825	0.095825	0.095825	0.095825	0.095825
g09	Min	680.630057	680.631353	680.630058	680.632379	680.630058
g10	Min	7049.3307	7894.812453	7251.773336	7459.747421	7251.396244
g11	Min	0.750000	0.749000	0.749000	0.7490021	0.749000
g12	Max	1.000000	0.994375	1.000000	1.000000	1.000000
g13	Min	0.053950	1.793361	1.102358	1.241422	0.997586

Guanajuato, CONCyTEG, Project No. 04-02-K117-037-II. The first author also acknowledges the support from CONCyTEG through Project No. 04-02K119-065-01, and from National Council of Science and Technology, CONACyT, to

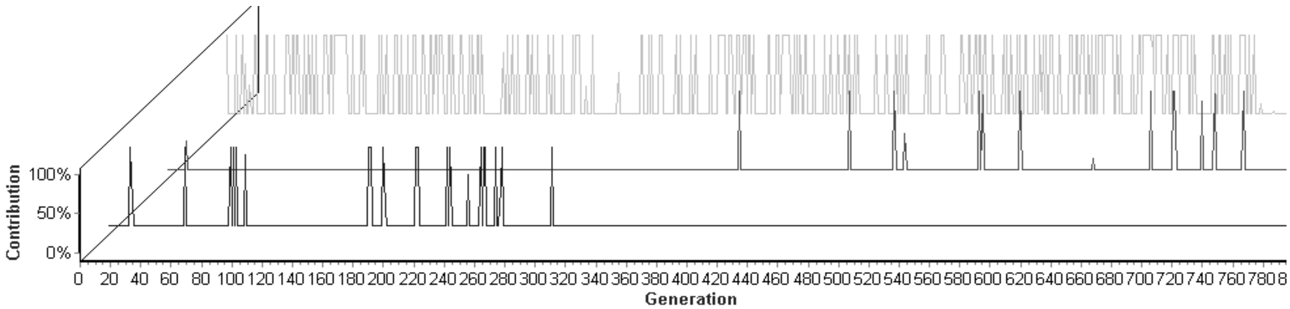

Figure 8: Contribution of each stage of PESO algorithm for function g02

Table 6: Results of PESO and SR for benchmark problems

		Best Result		Mean Result		Worst Result	
TF	**Optimal**	**PESO**	**SR**	**PESO**	**SR**	**PESO**	**SR**
g01	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000
g02	0.803619	0.792608	0.803515	0.721749	0.781975	0.614135	0.726288
g03	1.000000	1.005010	1.000000	1.005006	1.000000	1.004989	1.000000
g04	-30665.539	-30665.538672	-30665.539	-30665.538672	-30665.539	-30665.538672	-30665.539
g05	5126.4981	5126.484154	5126.497	5129.178298	5128.881	5148.859415	5142.472
g06	-6961.81388	-6961.813876	-6961.814	-6961.813876	-6875.940	-6961.813876	-6350.262
g07	24.306209	24.306921	24.307	24.371253	24.374	24.593504	24.642
g08	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825
g09	680.630057	680.630057	680.630	680.630057	680.656	680.630058	680.763
g10	7049.3307	7049.459452	7054.316	7099.101386	7559.192	7251.396245	8835.655
g11	0.750000	0.749000	0.750000	0.749000	0.750000	0.749000	0.750000
g12	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
g13	0.053950	0.081498	0.053957	0.626881	0.057006	0.997587	0.216915

Table 7: Results of PESO and Toscano's PSO for benchmark problems

		Best Result		Mean Result		Worst Result	
TF	**Optimal**	**PESO**	**PSO**	**PESO**	**PSO**	**PESO**	**PSO**
g01	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000
g02	0.803619	0.792608	0.803432	0.721749	0.790406	0.614135	0.750393
g03	1.000000	1.005010	1.004720	1.005006	1.003814	1.004989	1.002490
g04	-30665.539	-30665.538672	-30665.500	-30665.538672	-30665.500	-30665.538672	-30665.500
g05	5126.4981	5126.484154	5126.640	5129.178298	5461.081333	5148.859415	6104.750
g06	-6961.813880	-6961.813876	-6961.810	-6961.813876	-6961.810	-6961.813876	-6961.810
g07	24.306209	24.306921	24.351100	24.371253	25.355771	24.593504	27.316800
g08	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825
g09	680.630057	680.630057	680.638	680.630057	680.852393	680.630058	681.553
g10	7049.3307	7049.459452	7057.5900	7099.101386	7560.047857	7251.396245	8104.310
g11	0.750000	0.749000	0.749999	0.749000	0.750107	0.749000	0.752885
g12	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
g13	0.053950	0.081498	0.068665	0.626881	1.716426	0.997587	13.669500

pursue graduate studies at the Center for Research in Mathematics.

9. REFERENCES

[1] G. Coath and S. K. Halgamuge. A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems. In *Proceedings of the 2003 Congress on Evolutionary Computation*, pages 2419–2425. IEEE, December 2003.

[2] R. Eberhart, R. Dobbins, and P. Simpson. *Computational Intelligence PC Tools*. Academic Press Professional, 1996.

[3] J. Fieldsend and S. Singh. A multi-objective algorithm based upon particle swarm optimization, and efficient data structure and turbulence. In *Proceedings of 2002 U.K. Workshop on Computational Intelligence*, pages 37–44. The European Network on Intelligent Technologies for Smart Adaptive Systems, September 2002.

[4] N. Franken and A. P. Engelbrecht. Comparing pso structures to learn the game of checkers from zero knowledge. In *Proceedings of the 2003 Congress on Evolutionary Computation*, pages 234–241. IEEE, December 2003.

[5] A. Hernandez-Aguirre, S. Botello, and C. Coello. Passss: An implementation of a novel diversity strategy to handle constraints. In *Proceedings of the 2004 Congress on Evolutionary Computation CEC-2004*, volume 1, pages 403–410. IEEE Press, June 2004.

[6] J. Kennedy. Small worlds and mega-minds: Effects of

Table 8: The Best Results of PESO and DEPSO for Michalewicz' benchmark

TF	Optimal	PESO	DEPSO
g01	-15.000000	-15.000000	-15.000000
g02	0.803619	0.792608	0.7868
g03	1.000000	1.005010	1.0050
g04	-30665.539	-30665.538672	-30665.5
g05	5126.4981	5126.484154	5126.484
g06	-6961.81388	-6961.813876	-6961.81
g07	24.306209	24.306921	24.586
g08	0.095825	0.095825	0.095825
g09	680.630057	680.630057	680.641
g10	7049.3307	7049.459452	7267.4
g11	0.750000	0.749000	0.74900

neighborhood topology on particle swarm performance. In *IEEE Congress on Evolutionary Computation*, pages 1931–1938. IEEE, July 1999.

[7] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of the IEEE International Conference On Neural Networks*, pages 1942–1948. IEEE, November 1995.

[8] J. Kennedy and R. Eberhart. *The Particle Swarm: Social Adaptation in Information-Processing Systems*. McGraw-Hill, London, 1999.

[9] R. Mendes, J. Kennedy, and J. Neves. The fully informed particle swarm: Simpler, maybe better. *IEEE Transactions on Evolutionary Computation*, 8(3):204–210, June 2004.

[10] E. Mezura. *Alternatives to Handle Constraints in Evolutionary Optimization*. PhD thesis, CINVESTAV-IPN, Mexico, DF, 2004.

[11] T. Runarsson and X. Yao. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 4(3):284–294, September 2000.

[12] R. Storn. Sytem design by constraint adaptation and differential evolution. *IEEE Transactions on Evolutionary Computation*, 3(1):22–34, April 1999.

[13] G. Toscano and C. Coello. A constraint-handling mechanism for particle swarm optimization. In *Proceedings of the 2004 Congress on Evolutionary Computation*, pages 1396–1403. IEEE, June 2004.

[14] J. Zhang and F. Xie. Depso: Hybrid particle swarm with differential evolution operator. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pages 3816–3821. IEEE, October 2003.

APPENDIX

Next, we enumerate the test problems used for our experiments. This is the well known Michalewicz' benchmark extended by Runarsson and Yao [11].

1. **g01** Minimize: $f(\vec{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$
subject to:

$$\begin{aligned}
g_1(\vec{x}) &= 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \le 0\\
g_2(\vec{x}) &= 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \le 0\\
g_3(\vec{x}) &= 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \le 0\\
g_4(\vec{x}) &= -8x_1 + x_{10} \le 0\\
g_5(\vec{x}) &= -8x_2 + x_{11} \le 0\\
g_6(\vec{x}) &= -8x_3 + x_{12} \le 0\\
g_7(\vec{x}) &= -2x_4 - x_5 + x_{10} \le 0\\
g_8(\vec{x}) &= -2x_6 - x_7 + x_{11} \le 0\\
g_9(\vec{x}) &= -2x_8 - x_9 + x_{12} \le 0
\end{aligned}$$

where the bounds are $0 \le x_i \le 1$ $(i = 1, \ldots, 9)$, $0 \le x_i \le 100$ $(i = 10, 11, 12)$ and $0 \le x_{13} \le 1$. The global optimum is at $x^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$ where $f(x^*) = -15$. Constraints g_1, g_2, g_3, g_4, g_5 and g_6 are active.

2. **g02** Maximize: $f(\vec{x}) = \left| \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right|$
subject to:

$$\begin{aligned}
g_1(\vec{x}) &= 0.75 - \prod_{i=1}^{n} x_i \le 0\\
g_2(\vec{x}) &= \sum_{i=1}^{n} x_i - 7.5n \le 0
\end{aligned}$$

where $n = 20$ and $0 \le x_i \le 10$ $(i = 1, \ldots, n)$. The global maximum is unknown; the best reported solution is $f(x^*) = 0.803619$. Constraint g_1 is close to being active $(g_1 = -10^{-8})$.

3. **g03** Maximize: $f(\vec{x}) = \left(\sqrt{n}\right)^n \prod_{i=1}^{n} x_i$
subject to:

$$h(\vec{x}) = \sum_{i=1}^{n} x_i^2 - 1 = 0$$

where $n = 10$ and $0 \le x_i \le 1$ $(i = 1, \ldots, n)$. The global maximum is at $x_i^* = 1/\sqrt{n}$ $(i = 1, \ldots, n)$ where $f(x^*) = 1$.

4. **g04** Minimize: $f(\vec{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$
subject to:

$$\begin{aligned}
g_1(\vec{x}) &= 85.334407 + 0.0056858x_2x_5\\
&+ 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92 \le 0\\
g_2(\vec{x}) &= -85.334407 - 0.0056858x_2x_5\\
&- 0.0006262x_1x_4 + 0.0022053x_3x_5 \le 0\\
g_3(\vec{x}) &= 80.51249 + 0.0071317x_2x_5\\
&+ 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \le 0\\
g_4(\vec{x}) &= -80.51249 - 0.0071317x_2x_5\\
&- 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \le 0\\
g_5(\vec{x}) &= 9.300961 + 0.0047026x_3x_5\\
&+ 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \le 0\\
g_6(\vec{x}) &= -9.300961 - 0.0047026x_3x_5\\
&- 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \le 0
\end{aligned}$$

where: $78 \le x_1 \le 102$, $33 \le x_2 \le 45$, $27 \le x_i \le 45$ $(i = 3, 4, 5)$. The optimum solution is $x^* = (78, 33, 29.995256025682, 45, 36.775812905788)$ where $f(x^*) = -30665.539$. Constraints g_1 y g_6 are active.

5. **g05** Minimize: $f(\vec{x}) = 3x_1 + 0.000001x_1^3 + 2x_2 + (0.000002/3)x_2^3$
subject to:

$$\begin{aligned} g_1(\vec{x}) &= -x_4 + x_3 - 0.55 \leq 0 \\ g_2(\vec{x}) &= -x_3 + x_4 - 0.55 \leq 0 \\ h_3(\vec{x}) &= 1000\sin(-x_3 - 0.25) \\ &+ 1000\sin(-x_4 - 0.25) + 894.8 - x_1 = 0 \\ h_4(\vec{x}) &= 1000\sin(-x_3 - 0.25) \\ &+ 1000\sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0 \\ h_5(\vec{x}) &= 1000\sin(-x_4 - 0.25) \\ &+ 1000\sin(x_4 - x_3 - 0.25) + 1294.8 = 0 \end{aligned}$$

where $0 \leq x_1 \leq 1200$, $0 \leq x_2 \leq 1200$, $-0.55 \leq x_3 \leq 0.55$, and $-0.55 \leq x_4 \leq 0.55$. The best known solution is $x^* = (679.9453, 1026.067, 0.1188764, -0.3962336)$ where $f(x^*) = 5126.4981$.

6. **g06** Minimize: $f(\vec{x}) = (x_1 - 10)^3 + (x_2 - 20)^3$
subject to:

$$\begin{aligned} g_1(\vec{x}) &= -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0 \\ g_2(\vec{x}) &= (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0 \end{aligned}$$

where $13 \leq x_1 \leq 100$ and $0 \leq x_2 \leq 100$. The optimum solution is $x^* = (14.095, 0.84296)$ where $f(x^*) = -6961.81388$. Both constraints are active.

7. **g07** Minimize:

$$\begin{aligned} f(\vec{x}) = \; & x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 \\ & +4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 \\ & +7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45 \end{aligned}$$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0 \\ g_2(\vec{x}) &= 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0 \\ g_3(\vec{x}) &= -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0 \\ g_4(\vec{x}) &= 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 \leq 120 \\ g_5(\vec{x}) &= 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0 \\ g_6(\vec{x}) &= x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leq 0 \\ g_7(\vec{x}) &= 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 \leq 30 \\ g_8(\vec{x}) &= -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0 \end{aligned}$$

where $-10 \leq x_i \leq 10$ $(i = 1, \ldots, 10)$. The global optimum is $x^* = (2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927)$ where $f(x^*) = 24.3062091$. Constraints g_1, g_2, g_3, g_4, g_5 and g_6 are active.

8. **g08** Maximize: $f(\vec{x}) = \frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1+x_2)}$
subject to:

$$\begin{aligned} g_1(\vec{x}) &= x_1^2 - x_2 + 1 \leq 0 \\ g_2(\vec{x}) &= 1 - x_1 + (x_2 - 4)^2 \leq 0 \end{aligned}$$

where $0 \leq x_1 \leq 10$ and $0 \leq x_2 \leq 10$. The optimum solution is located at $x^* = (1.2279713, 4.2453733)$ where $f(x^*) = 0.095825$. The solutions is located within the feasible region.

9. **g09** Minimize:

$$\begin{aligned} f(\vec{x}) = \; & (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 \\ & +10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7 \end{aligned}$$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \leq 0 \\ g_2(\vec{x}) &= -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \leq 0 \\ g_3(\vec{x}) &= -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \leq 0 \\ g_4(\vec{x}) &= 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0 \end{aligned}$$

where $-10 \leq x_i \leq 10$ $(i = 1, \ldots, 7)$. The global optimum is $x^* = (2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227)$ where $f(x^*) = 680.6300573$. Two constraints are active (g_1 and g_4).

10. **g10** Minimize: $f(\vec{x}) = x_1 + x_2 + x_3$
subject to:

$$\begin{aligned} g_1(\vec{x}) &= -1 + 0.0025(x_4 + x_6) \leq 0 \\ g_2(\vec{x}) &= -1 + 0.0025(x_5 + x_7 - x_4) \leq 0 \\ g_3(\vec{x}) &= -1 + 0.01(x_8 - x_5) \leq 0 \\ g_4(\vec{x}) &= -x_1x_6 + 833.33252x_4 + 100x_1 \\ &- 83333.333 \leq 0 \\ g_5(\vec{x}) &= -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \leq 0 \\ g_6(\vec{x}) &= -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leq 0 \end{aligned}$$

where $100 \leq x_1 \leq 10000$, $1000 \leq x_i \leq 10000$, $(i = 2, 3)$, $10 \leq x_i \leq 1000$, $(i = 4, \ldots, 8)$. The global optimum is: $x^* = (579.3167, 1359.943, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979)$ where $f(x^*) = 7049.3307$. g_1, g_2 and g_3 are active.

11. **g11** Minimize: $f(\vec{x}) = x_1^2 + (x_2 - 1)^2$
subject to:

$$h(\vec{x}) = x_2 - x_1^2 = 0$$

where: $-1 \leq x_1 \leq 1$, $-1 \leq x_2 \leq 1$. The optimum solution is $x^* = (\pm 1/\sqrt{2}, 1/2)$ where $f(x^*) = 0.75$.

12. **g12** Maximize: $f(\vec{x}) = \frac{100-(x_1-5)^2-(x_2-5)^2-(x_3-5)^2}{100}$
subject to:

$$g_1(\vec{x}) = (x_1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \leq 0$$

where: $0 \leq x_i \leq 10$ $(i = 1, 2, 3)$ and $p, q, r = 1, 2, \ldots, 9$. The feasible region of the search space consists of 9^3 disjointed spheres. A point (x_1, x_2, x_3) is feasible if and only if there exist p, q, r such the above inequality holds. The global optimum is located at $x^* = (5, 5, 5)$ where $f(x^*) = 1$.

13. **g13** Minimize: $f(\vec{x}) = e^{x_1x_2x_3x_4x_5}$
subject to:

$$\begin{aligned} h_1(\vec{x}) &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0 \\ h_2(\vec{x}) &= x_2x_3 - 5x_4x_5 = 0 \\ h_3(\vec{x}) &= x_1^3 + x_2^3 + 1 = 0 \end{aligned}$$

where: $-2.3 \leq x_i \leq 2.3$ $(i = 1, 2)$ and $-3.2 \leq x_i \leq 3.2$ $(i = 3, 4, 5)$. The optimum solution is $x^* = (-1.717143, 1.595709, 1.827247, -0.7636413, -0.763645)$ where $f(x^*) = 0.0539498$.