COPSO: CONSTRAINED OPTIMIZATION VIA PSO ALGORITHM

Arturo Hernández Aguirre, Angel E. Muñoz Zavala, E. Villa Diharce and S. Botello Rionda

Comunicación Técnica No I-07-04/22-02-2007
(CC/CIMAT)

COPSO: Constrained Optimization via PSO algorithm

Arturo Hernández Aguirre, Angel E. Muñoz Zavala
† E. Villa Diharce and S. Botello Rionda

Center for Research in Mathematics (CIMAT),
Department of Computer Science, † Department of Statistics
A.P. 402, Guanajuato, Gto. CP. 36000, México

artha, aemz, villadi, botello@cimat.mx

Abstract

This paper introduces the COPSO algorithm (Constrained Optimization via Particle Swarm Optimization) for the solution of single objective constrained optimization problems. The approach includes two new perturbation operators to prevent premature convergence, and a new ring neighborhood structure. A constraint handling technique based on feasibility and sum of constraints violation, is equipped with an external file to store particles we termed "tolerant" . The goal of the file is to extend the life period of those particles that otherwise would be lost after the adjustment of the tolerance of equality constraints. COPSO is applied to various engineering design problems, and for the solution of state of the art benchmark problems. Experiments show that COPSO is robust, competitive and fast.

Keywords: Particle Swarm, Constrained Optimization, Swarm Intelligence, Evolutionary Algorithm

1 Introduction

Particle swarm optimization (PSO) algorithm is a population-based optimization technique inspired by the motion of a bird flock, or fish schooling. Such groups are social organizations whose overall behavior relies on some sort of communication amongst members, and cooperation. All members obey a set of simple rules that model the communication inside the flock, and between the flock and the environment. The global behavior, however, is far more complex and generally successful. For instance, a flock is usually successful at finding the best place for feeding, same which seems impossible to achieve by any single member. The PSO paradigm seems to follow the five basic principles of swarm intelligence: proximity, quality, diverse response, stability, and adaptability [1]. These principles translate into the following: a swarm should carry out simple space and time computations, respond to quality factors in the environment, react in various ways to stimulus, keep its mode of behavior at changing environments, but should change its mode if it is worth the computational price [2]. The last two principles, stability and adaptability, are opposite views of the same goal. Therefore, a trade-off between them is necessary since the strength of one may diminish the capacity to achieve the other.

A member of the flock is called "particle", thus a flock is a collection of particles. The popular term "flying the particles" means the exploration of the search space. Every particle knows its current position and the best position visited since the first fly. PSO performs exploration by continually sensing (reading) the search space at local level. The information collected by the particles is concentrated and sorted to find the best member (called global best). The new best member and the current best member are compared

[1]Source code available at request

and the best one is kept as global best. Its position is communicated to all flock members thus in the next fly the particles know where the best spot lies in the search space. Locating the next best spot is the main task of the flock for which exploration and therefore population diversity is crucial. In this scenario the flock explores the space but remains stable after changing its flying direction (fourth principle). At the same time, however, all flock members are attracted by the position of the global best. Thus the flock adapts to new attractors in space (fifth principle). Being simultaneously stable and adaptable is a major problem for PSO algorithms. A flock must keep flying and looking for better spots even when the current one seemed good.

PSO is a global optimization technique, therefore, it lacks an explicit mechanism to bias the search towards the feasible region in constrained search spaces. This paper presents a new PSO based approach called Constrained Optimization via PSO, (COPSO). COPSO performs the main PSO algorithm but executes two additional steps: the C-perturbation which is oriented to sustain global exploration by keeping diversity, and the M-perturbation oriented to local refinement of solutions. A review of several diversity control techniques for PSO is presented in Section 2. Next, a review of constraint handling techniques used by PSO algorithms is presented in Section 3. The general class of problems of interest is defined in Section 4. Section 5 presents the two basic PSO models and explains the advantages of the local best approach implemented by COPSO. The COPSO algorithm is thoroughly explained in Section 7. In Section 8, COPSO is used to solve a state of the art benchmark of 24 functions. Comparisons against four different approaches are provided (one of them based on PSO). Furthermore, in Section 9, five engineering design problems are solved and COPSO is compared to three PSO based approaches (found in the literature). Conclusions are provided in Section 10. The formal description of the problems chosen for the experiments is given in Appendix A and Appendix B.

2 Diversity control in PSO

In PSO, the source of diversity, called *variation*, comes from two sources. One is the difference between the particle's current position and the global best, and the other is the difference between the particle's current position and its best historical value. Although variation provides diversity, it can only be sustained by for a limited number of generations because convergence of the flock to the best is necessary to refine the solution. In an early analysis, Angeline shows that PSO may not converge neither refine solutions when variation is null, that is, when all the particles rest near by the best spot [3]. Although most approaches procure diversity by including ad-hoc operators to the basic PSO algorithm, a simply approach of Eberhart and Shi [4] (now adopted in the basic PSO model), keeps variation for as many generations as possible by slowing down the speed of the flock. In this approach, inertia coefficients are applied to the particle's previous velocity and the current one. Small coefficients let the particles reach the global best in a few number of iterations but large values do the opposite and favor exploration.

Many ad-hoc operators are inspired in evolutionary operators, for instance reproduction. Angeline [3], and also Eberhart [5], proposed population breeding. Therefore, two randomly chosen particles (parents) may reproduce and create offsprings. Lovbjerg [6], and more recently Settles [7] implemented breeding with some success. More investigations on reproduction as source of diversity were recently conducted by S. Das [8]. He adapted the reproduction operator of differential evolution [9, 10] to PSO, and reported robust performance in a small set of global optimization problems.

If these approaches keep diversity by preventing premature convergence, other approaches let premature convergence happen but later in the process they try to extinguish it. For instance, in Krinks approach, the particles as clustered and their density used as a measure of crowding [11]. Once such clusters are detected, their density is reduced by bouncing away the particles. Blackwell also investigated a mechanism that repels clustered particles [12].

The diversity of the flock is important to reach the optimum. The formal analysis of van den Bergh shows that the PSO algorithm is not a global optimizer, and that the flock will only converge to the best position visited, not the global optimum [13]. Therefore, the quest for diversity is a sound approach to approximate the global optimum since more diversity can be read as "more positions visited". van den Bergh also showed that local convergence is guaranteed if local exploration is performed around the global best [14]. These two ideas are captured by the perturbation operators of COPSO: the C-perturbation implements global exploration, whilst the M-perturbation performs exploration at a local level.

3 Related work on constraint handling techniques for PSO

Several authors have noted how important is to achieve the proper balance between the diversity control technique, the constraint handling technique, and the particular features of the search engine [15, 16, 17, 18]. A PSO algorithm in constrained space must take into account that the inherent tendency of PSO to premature convergence may be increased by a constraint handling technique that overestimates unfeasible particles. The next two issues are common to PSO algorithms in a constrained search space:

- Which particle is a good leader to guide the flock towards the feasible region? An example of such question is whether the sole distance from a particle to the feasible region can be used to identify the leader.
- How to maintain the exploration and exploitation capacity of the flock during generations? That is, the behavior of the flock needs to change, slowly, from exploration to exploitation.

Many initial approaches did not combine a diversity maintenance strategy with the constraint handling technique. For instance, Parsopoulos used a multi-stage assignment penalty function without diversity control [19]. Hu and Eberhart proposed a feasibility preservation strategy that determines the best particle [20, 21]. Both penalty and feasibility preservation strategies were analyzed by G. Coath [22] (whose experiments clearly detect the need of some form of diversity control). He and Prempain used a "fly-back" mechanism that returns an unfeasible particle to its previous feasible position [23]. An more important drawback of this technique is the requirement of a all-feasible initial population. A few more sophisticated approaches include diversity control. For instance Toscano and Coello [24], use a turbulence operator (a kind of mutation) combined with a feasibility tournament [25]. They succeeded in most problems but faced weak capacity to refine the solutions. Reproduction operators can also be combined with PSO in constrained optimization. For instance, W. Zhang [26] proposed to compute the velocity term by taking turns between PSO and differential evolution. At odd generations he would use the PSO formula; at even generations he would compute it as in the differential evolution formalism. Zhang [27], introduced a special technique, called *periodic mode*, to handle equality constraints. His approach consists in keeping the global-best near the boundary thus the flock which is constantly pulled to the border, can sustain exploration. We contrast our method with the best results of the recent proposals reviewed in this section.

4 Problem Statement

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

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

subject to:

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

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

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).

5 From global best to local best PSO algorithm

In PSO, the particles fly over a real valued n-dimensional search space and each one has three attributes: position $\vec{x}$, velocity $\vec{v}$, and best position visited $Pbest$. The best $Pbest$ is called global best, or leader, $GBest$. The next position of any member is computed by adding a displacement (named velocity) to its current position.

$$\vec{x}(t+1) = \vec{x}(t) + \vec{v}(t+1) \tag{1}$$

The velocity term combines the local information of the particle with global information of the flock, in the following way.

$$\vec{v}(t+1) = w * \vec{v}(t) + \phi 1 * (\vec{x}_{PBest} - \vec{x}) + \phi 2 * (\vec{x}_{GBest} - \vec{x}) \tag{2}$$

The equation above reflects the socially exchanged information. It resumes PSO three main features: distributed control, collective behavior, and local interaction with the environment [2, 28].The second term is called the cognitive component, while the last term is called the social component. w is the inertia weight, and $\phi 1$ and $\phi 2$ are called acceleration coefficients.

The best particle is called "the leader". The whole flock moves following the leader but the leadership can be passed from member to member. At every PSO iteration the flock is inspected to find the best member. Whenever a member is found to improve the function value of the current leader, that member takes the leadership. A leader can be global to all the flock, or local to a flock's neighborhood. In the latter case there are as many local leaders as neighborhoods. Having more than one leader in the flock translates into more attractors or good spots in space. Therefore, the use of neighborhoods is a natural approach to fight premature convergence [29].

Flock neighborhoods have a structure that define the way information is concentrated and then distributed among its members. The most common flock organizations are shown in Figure 1. The organization of the flock affects convergence and search capacity. The star structure has reported the fastest convergence time while the ring has been reported to traverse larger areas of the search space [2, 28]). The global best approach (see Equation 2), works on the star structure; local best only works on neighborhoods of particles such as the ring [30, 31]. Flock members organized in a ring communicate with n immediate neighbors, $n/2$ on each side (usually $n = 2$). Every particle is initialized with a permanent label which is independent of its geographic location in space. Finding the local best (LBest) neighbor of particle i is done by inspecting the particles in the neighborhood: $i+1, i+2, \ldots, i+n/2$ and $i-1, i-2, \ldots, i-n/2$ (COPSO uses the selfless model, therefore, particle i is not considered member of the neighborhood [32]). The approach introduced in this paper organizes the flock in a modified ring fashion called singly-linked ring. This structure improves the exploration capacity of the flock.

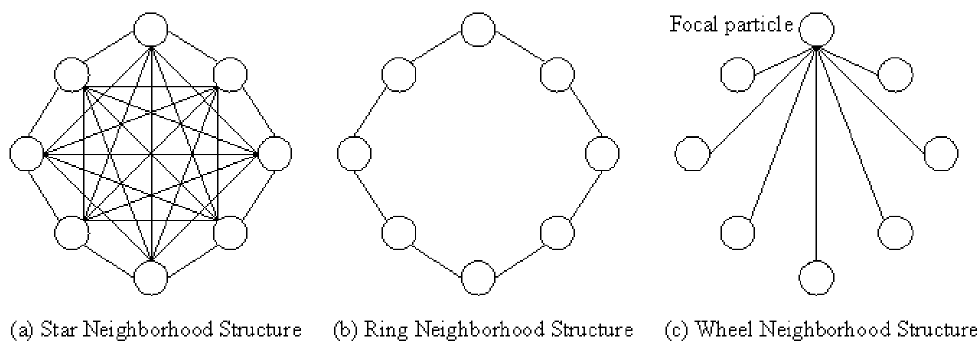

Figure 1: Neighborhood structures for PSO

The equation for local best PSO is similar to that of the global best version. One would simply substitute "GBest" by "LBest" in Equation 2. This is shown in Figure 2 in greater detail, where n is the population

```
%create members and measure their properties X_0 = Rand(LL, UL)
V_0 = Rand(-(UL-LL), (UL-LL))
F_0 = Fitness ( X_0 )
C_0 = SCV ( X_0 )
PBest_0= X_0
FBest_0= F_0
CBest_0= C_0
Function LBestPSO
For i = 1 To maxgenerations
  LBest_i = LocalBest ( FBest_i, CBest_i )
  V_{i+1} = Velocity ( V_i, X_i, PBest_i, LBest_i)
  X_{i+1} = X_i + V_{i+1}
  F_{i+1} = Fitness ( X_{i+1} )
  C_{i+1} = SCV ( X_{i+1} )
  [PBest_{i+1}, FBest_{i+1}, CBest_{i+1}] = ParticleBest ( PBest_i, X_{i+1}, FBest_i, F_{i+1}, CBest_i, C_{i+1} )
End For
```

```
Function Velocity
For k = 0 To n
  For j = 0 To d
    r1 = φ1 * U(0, 1);    r2 = φ2 * U(0, 1);    w = U(0.5, 1);
    V_{i+1}[k, j] = w * V_i[k, j] + r1 * (PBest_i[k, j] - X_i[k, j]) + r2 * (LBest_i[k, j] - X_i[k, j])
  End For
End For
```

Figure 2: Pseudo-code of *LBestPSO* algorithm

size, d is the dimension of the search space, $\phi1$ and $\phi2$ are constants set to 1 (not needed), and w is the inertia weight whose random value comes from a uniform distribution in $[0.5, 1]$. Of course, *ParticleBest* updates the local memory of each particle, and *LocalBest* finds the best in every neighborhood.

6 Singly-Linked Ring Neighborhood Structure

The ring neighborhood structure is commonly used by PSO implementations. Each particle is assigned a permanent label which is used to construct the neighborhoods. For each particle k, a neighborhood of size n is composed by the next $n/2$ linked particles, and by $n/2$ previous particles. For example, in a neighborhood of size $n = 2$, particle k has 2 neighbors: particles $k-1$ and $k+1$. The best particle of the neighborhood is the *LBest* of particle k. The ring structure is implemented by a doubly-linked list, as shown in Figure 3-a.

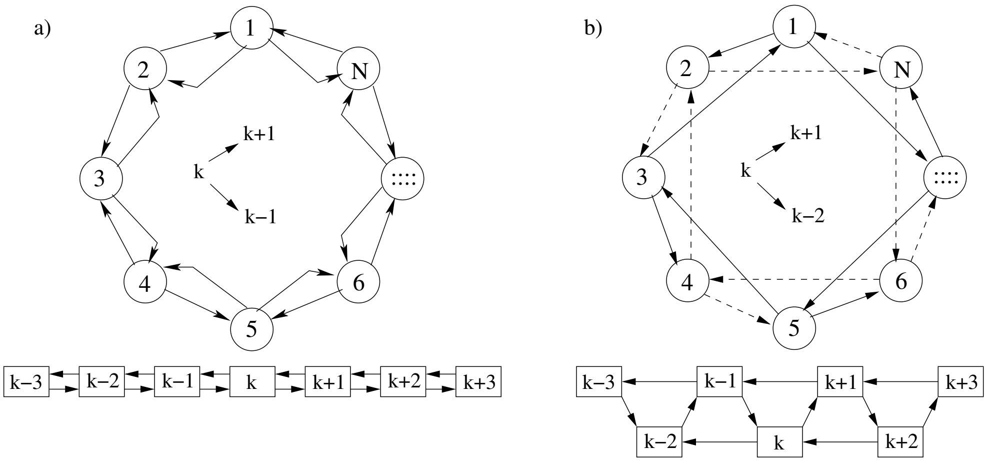

Figure 3: Ring neighborhood structures for PSO

COPSO uses an alternative ring implementation, the singly-linked list, shown in Figure 3-b. This structure improved the success of experimental results by a very important factor. Although more details are not provided, the advantages of the new structure can be explained as follows (see Figure 3). Assume the ring is based on a double link, and particle 3 is the best of particle's neighborhood 2 and 4. Since 2 and 4 have particle 3 in their own neighborhood, but 3 is the best, then particles 2 and 4 are directly pulled by 3. Simultaneously, particle 3 has particles 2 and 4 as neighbors. Therefore, 3 attracts 2 and 2 attracts 3. After some generations, particles 2 and 3 converge to the same point. Now, assume the ring is based on a single link, and particle 3 is again the best of particle's neighborhood 2 and 5. But particle 3 now has particles 1 and 4 as neighbors (not 2 and 5 as in the double link). Since particle 4 has particles 2 and 5 as neighbors, 3 attracts 2 but 2 only attracts 3 through particle 4. Therefore, the particle in between cancels the mutual attraction, and in consequence reduces the convergence of the flock.

For each particle i, the members of a neighborhood of size n are selected by the next algorithm.

1. Set $step = 1$
2. Set $switch = 1$ (pick from left or right side)
3. Include in the neighborhood the particle $i + switch * step$
4. Increment $step = step + 1$
5. Calculate $switch = -switch$
6. Repeat step 3 until $neighborhood\_size = n$.

7 COPSO = LBestPSO + Perturbations + Constraint Handling

COPSO improves the local best PSO algorithm with external procedures that keep diversity and guide the flock towards good spots without destroying its self organization capacity. Thus, only the memory of best visited location, PBest, may be altered by the perturbation operators. Flying the particles remains the main task of PSO. A view of COPSO algorithm with its three components is shown in Figure 7. In the first stage the standard LbestPSO algorithm (described in Figure 2) runs one iteration [28]. Then the perturbations are applied to $PBest$ in the next two stages. The goal of the second stage is to add a perturbation generated from the linear combination of three random vectors. This perturbation is preferred over other operators because it preserves the distribution of the population (also used for reproduction by the differential evolution algorithm [33]). In COPSO this perturbation is called C-Perturbation. It is applied to the members of PBest to yield a set of temporal particles $Temp$. Then each member of $Temp$ is compared with its corresponding father and PBest is updated with the child if it wins the tournament. Figure 4 shows the pseudo-code of the **C-Perturbation** operator.

```
For k = 0 To n
  For j = 0 To d
    r = U(0, 1)
    p1 = k
    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 4: Pseudo-code of **C-Perturbation**

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 implemented by adding small random numbers (from a uniform distribution) to every design variable. The perturbation, called M-Perturbation, is applied to every member of PBest to yield a set of temporal particles $Temp$. Then each member of $Temp$ is compared with its corresponding father and PBest is updated with the child if it wins the tournament. Figure 5 shows the pseudo-code of the **M-Perturbation** operator. The perturbation is added to every dimension of the decision vector with probability $p = 1/d$ (d is the dimension of the decision variable vector).

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

Figure 5: Pseudo-code of **M-Perturbation**

These perturbations have the additional advantage of keeping the self-organization potential of the flock since they only work on the PBest memory. In Figure 6 the position PBest is relocated to a new "best" after the perturbation operations. Notice this change is made to the particle's memory of best visited location. When PSO takes turn to perform its computations, it finds everything as left in the previous generation, except that the memory PBest may store a better position. In Figure 7 the main algorithm of COPSO is listed. p is a linearly decreasing probability from 1.0 to 0 (according to the function evaluations), LL and UL are the lower and upper limits of the search space. *LocalBest* and *ParticleBest* perform the obvious task as explained before. The procedure **Update tolerant file** is invoked only when the problem has equality constraints. This is explained next.

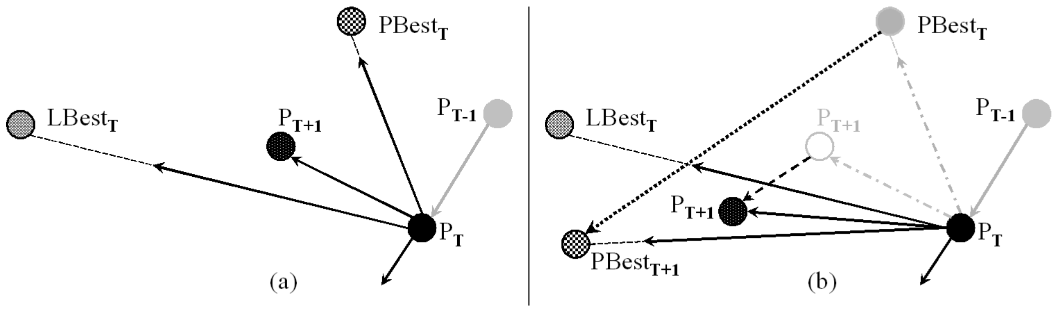

Figure 6: PBest updated after the C and M perturbations

7.1 Constraint handling approach

K. Deb introduced a feasibility tournament selection based on the idea that any individual in a constrained search space must first comply with the constraints and then with the function value [25]. COPSO adopted such popular tournament selection whose rules have been included in the functions LocalBest and ParticleBest: 1) given two feasible particles, pick the one with better function value; 2) if both particles are infeasible, pick the particle with the lowest sum of constraint violation, and 3) from a pair of feasible and infeasible particles, pick the feasible one. The sum of constraint violations is, of course, the total value by which unsatisfied constraints are violated (computed by function **SCV** in Figure 7). Two additional characteristics of the constraint handling mechanism are:

- Dynamic tolerance for equality constraints. COPSO handles equality constraints by rewriting them as inequality constraints of the form $|h_j| \leq \epsilon$, where ϵ is called the tolerance. In COPSO, the tolerance is linearly decremented from 1.0 to a specified target value during the first 90% of function evaluations (1E-06 in our experiments). For the last 10% the tolerance is kept fixed; thus, the particles have additional time to achieve convergence.

- Storage of tolerant particles. We call *tolerant* a particle that remains feasible after two or more consecutive reductions of the tolerance. Thus, a tolerant particle is located very *near* the constraint boundary. Many other particles become unfeasible after the tolerance value is decremented. Procedure

```
% create members and measure their properties
X_0 = Rand(LL, UL)
V_0 = Rand(-(UL-LL), (UL-LL))
F_0 = Fitness ( X_0 )
C_0 = SCV ( X_0 )
PBest_0= X_0
FBest_0= F_0
CBest_0= C_0
                    Stage 1
Function COPSO
For i = 1 To maxgenerations
  LBest_i = LocalBest ( FBest_i, CBest_i )
  V_{i+1} = Velocity ( V_i, X_i, PBest_i, LBest_i)
  X_{i+1} = X_i + V_{i+1}
  F_{i+1} = Fitness ( X_{i+1} )
  C_{i+1} = SCV ( X_{i+1} )
  [PBest_{i+1}, FBest_{i+1}, CBest_{i+1}] = ParticleBest ( PBest_i, X_{i+1}, FBest_i, F_{i+1}, CBest_i, C_{i+1} )
  Update tolerant file (PBest_{i+1})
                    Stage 2
  If (U(0,1) < p )
    Temp = C-Perturbation (PBest_{i+1} )
    FTemp = Fitness ( Temp )
    CTemp = SCV ( Temp )
    [PBest_{i+1}, FBest_{i+1}, CBest_{i+1}] = ParticleBest ( PBest_i, Temp, FBest_i, FTemp, CBest_i,
CTemp )
    Update tolerant file (PBest_{i+1})
  End If
                    Stage 3
  If (U(0,1) < p )
    Temp = M-Perturbation (PBest_{i+1} )
    FTemp = Fitness ( Temp )
    CTemp = SCV ( Temp )
    [PBest_{i+1}, FBest_{i+1}, CBest_{i+1}] = ParticleBest ( PBest_i, Temp, FBest_i, FTemp, CBest_i,
CTemp )
    Update tolerant file (PBest_{i+1})
  End If
End For
```

Figure 7: Pseudo-code of *COPSO* algorithm

Update tolerant file keeps a file of tolerant particles. Figure 8 describes the actions performed: the best among all PBest, called q, is inserted in the file. The tolerance is decremented and the whole file is evaluated. The best particle in the file substitutes q.

```
Update Tolerant File (PBest)
% TF: tolerant file, initially empty
q ← Particlebest(PBest)
If TF == full
   TF ← TF \ any member TF
Endif
TF ← TF ∪ q
TF ← SCV(TF)
bp ← Particlebest(TF)
PBest ← PBest \ q
PBest ← PBest ∪ bp
```

Figure 8: Procedure Update-tolerant-file

For COPSO the total amount of violation measured on the equality constraints helps to determine a better leader for the flock. Other approaches simply consider that any individual is as good as any other if they are located inside the margin of tolerance.

7.2 Refining Solutions

The cooperation between PSO and the perturbation operators have been carefully analyzed by the authors through out the many experiments conducted. The PSO stage performs very efficiently at refining solutions in a local space, but exploration is performed by the perturbation operators. Hence, the perturbation operators have their activity reduced along generations so a refining phase conducted by PSO may take place. The implementation of these cooperative activities is as follows: the perturbation operators are invoked with probability $p = 1$ when the flock is flown for the first time. This probability is constantly and linearly decremented reaching its final value of $p = 0$ at the last time the particles fly.

8 Experiments on benchmark functions

For all experiments, COPSO used the following parameters: factors $c1$ and $c2$ are not needed thus set to 1. The inertia weight w is a random number in the interval [0.5,1] with uniform distribution. Flock size of 100 members. Minimum tolerance value of 1×10^{-6} for all equality constraints. Total number of function evaluations is 350,000. A PC computer with Windows XP and C++ Builder Compiler, Pentium-4 processor at 3.00GHz, 1.00 GB of RAM. Two large sets of experiments were conducted, one on a benchmark of 24 functions, and another on engineering problems.

8.1 The benchmark problems

E. Mezura has extended to 24 functions the original benchmark of Runnarson and Yao, with 13 functions [34]. The definition of these functions is given in Appendix A. The basic statistics for 30 runs are shown in Table 1.

The Table 2 shows the COPSO's convergency in the benchmark problems. We present the objective function evaluations required to approximate the best-known optimum within a margin of 1E-4. The top value is $350,000$ function evaluations. In test problems g03, g05, g11, g13, g14, g15, g17, g20, g21, g22, and g23, the number of function evaluations required by equality constraints to reach a tolerance value of ϵ=1E-6, is $315,000$.

In the Table 2, we also show the number of feasible runs, *F.Runs*. A run that finds at least one feasible solution in less than $350,000$ fitness evaluations is called feasible. The column *S.Runs* shows the number of successful runs (when the best value found is within 1E-4 of the optimal the run is successful). The experiments show a poor performance of COPSO in test problems g20, g21, g22 and g23. These problems

Table 1: The results of COPSO on the benchmark

TF	Optimal	Best	Median	Mean	Worst	S. D.	F. S.
g01	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	0	30/30
g02	-0.803619	-0.803619	-0.803617	-0.801320	-0.786566	4.5945E-03	30/30
g03	-1.000000	-1.000005	-1.000005	-1.000005	-1.000003	3.1559E-07	30/30
g04	-30665.539	-30665.538672	-30665.538672	-30665.538672	-30665.538672	0	30/30
g05	5126.4981	5126.498096	5126.498096	5126.498096	5126.498096	0	30/30
g06	-6961.8138	-6961.813876	-6961.813876	-6961.813876	-6961.813876	0	30/30
g07	24.306209	24.306209	24.306210	24.306212	24.306219	3.3414E-06	30/30
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	0	30/30
g09	680.630057	680.630057	680.630057	680.630057	680.630057	0	30/30
g10	7049.248	7049.248020	7049.248638	7049.250087	7049.263662	3.6121E-03	30/30
g11	0.750000	0.749999	0.749999	0.749999	0.749999	0	30/30
g12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	0	30/30
g13	0.053950	0.053950	0.053950	0.053950	0.053965	2.7688E-06	30/30
g14	-47.761	-47.761108	-47.747212	-47.741430	-47.670921	2.1566E-02	30/30
g15	961.715	961.715171	961.715171	961.715171	961.715171	0	30/30
g16	-1.905	-1.905155	-1.905155	-1.905155	-1.905155	0	30/30
g17	8876.98068	8856.502344	8863.875542	8877.812811	8941.344349	30.1195	30/30
g18	-0.8660	-0.866025	-0.866025	-0.866001	-0.865568	8.7410E-05	30/30
g19	32.386	32.349645	32.386872	32.411596	32.571543	6.3055E-02	30/30
g20	0.967	*0.204095	*0.209711	*0.212003	*0.233281	6.9487E-03	0/30
g21	193.778349	205.852693	279.309106	273.298016	303.454837	23.8556	30/30
g22	12812.500	*157.513639	*3161.102678	*5376.226516	*18732.783872	5.0132E+03	0/30
g23	-400.0551	-361.856637	-136.564268	-138.407772	3.775736	84.521723	30/30
g24	-5.508	-5.508013	-5.508013	-5.508013	-5.508013	0	30/30

* Infeasible solution

have several equality constraints, in fact the problems g20 and g22 have more than 10 of them. In test problem g17, the tolerance value used in the equality constraints allows better results of COPSO than the optimal reported.

Table 2: COPSO's Convergency

TF	Best	Median	Mean	Worst	S.D.	Feasible Runs	Successful Runs
g01	90800	95000	95396.67	99400	2613.29	30	30
g02	142900	175800	179395.45	232100	28120.18	30	22
g03	315100	315100	315123.33	315600	97.14	30	30
g04	59600	65100	65086.67	70000	2713.28	30	30
g05	315100	315100	315256.67	315900	245.91	30	30
g06	47100	54200	53410.00	57000	2577.80	30	30
g07	185500	227600	233400.00	304500	32253.97	30	30
g08	3600	6850	6470.00	8500	1381.94	30	30
g09	69900	78500	79570.00	102400	7154.65	30	30
g10	167200	221300	224740.00	307200	38407.82	30	30
g11	315000	315000	315000.00	315000	0	30	30
g12	400	6900	6646.67	10400	2606.98	30	30
g13	315100	315150	315546.67	318100	710.87	30	30
g14	326900	326900	326900.00	326900	0	30	1
g15	315100	315100	315100.00	315100	0	30	30
g16	37200	41000	40960.00	45400	2210.88	30	30
g17	315100	316500	316608.70	318800	1061.69	30	23
g18	102200	153600	167088.89	252900	43430.30	30	27
g19	206800	259650	264414.29	331000	36456.84	30	14
g20	NR	NR	NR	NR	NR	0	0
g21	NR	NR	NR	NR	NR	30	0
g22	NR	NR	NR	NR	NR	0	0
g23	NR	NR	NR	NR	NR	30	0
g24	14900	19350	19156.67	22200	1927.24	30	30

NR: Optimal not reached

Firstly, we compare COPSO and PSO for constrained optimization. Since both are based on PSO this is the fairest comparison reported in the paper.

8.2 Comparison COPSO - PSO for constrained optimization

Toscano and Coello [24] proposed a constraint handling technique for PSO. Their approach handles constraints through a feasibility tournament, and keeps diversity by adding mutations to the velocity vector. The comparison is shown in Table 3. TC-PSO (Toscano and Coello's PSO) performed 340,000 fitness function evaluations, 10,000 less than COPSO, but it is not significative for the comparison. COPSOs performance is better than TC-PSO on test problems g02, g05, g07, g10, g11 and g13. (Hu and Eberhart [21], and Zhang [27] reported the first solutions to these problems with very limited success).

Table 3: Comparison of COPSO and Toscano-Coello PSO in the benchmark problems

		Best Result		Mean Result		Worst Result	
TF	**Optimal**	**COPSO**	**TC-PSO**	**COPSO**	**TC-PSO**	**COPSO**	**TC-PSO**
g01	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803619	-0.803432	-0.801320	-0.790406	-0.786566	-0.750393
g03	-1.000000	-1.000005	-1.004720	-1.000005	-1.003814	-1.000003	-1.002490
g04	-30665.539	-30665.538672	-30665.500	-30665.538672	-30665.500	-30665.538672	-30665.500
g05	5126.4981	5126.498096	5126.640	5126.498096	5461.081333	5126.498096	6104.750
g06	-6961.8138	-6961.813876	-6961.810	-6961.813876	-6961.810	-6961.813876	-6961.810
g07	24.306209	24.306209	24.351100	24.306212	25.355771	24.306219	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	680.630057	681.553
g10	7049.3307	7049.248020	7057.5900	7049.250087	7560.047857	7049.263662	8104.310
g11	0.750000	0.749999	0.749999	0.749999	0.750107	0.749999	0.752885
g12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
g13	0.053950	0.053950	0.068665	0.053950	1.716426	0.053965	13.669500

8.3 Comparison COPSO - Diversity-DE

Mezura, Velazquez and Coello [35], modified the Differential Evolution algorithm in a way that every parent may have more than one offspring. The winner is the best child but then the child is compared to the current parent. Another tournament is performed but this time the winner is found by tossing a coin and comparing by fitness value, or by constraint violation (similar to Stochastic Ranking [36]. The comparison is shown in Table 4, the number of fitness evaluations for both algorithms is 225000. The performance of COPSO and DE is very similar. A little advantage is shown by COPSO and Diversity-DE on test problems g09 and g10, respectively; but COPSO shows a better performance on test problem g13.

Table 4: Comparison of COPSO and Diversity-DE in the benchmark problems

		Best Result		Mean Result		Worst Result	
TF	**Optimal**	**COPSO**	**DE**	**COPSO**	**DE**	**COPSO**	**DE**
g01	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803618	-0.803619	-0.797141	-0.798079	-0.785217	-0.751742
g03	-1.000000	-1.000005	-1.000	-1.000004	-1.000	-1.000003	-1.000
g04	-30665.539	-30665.538672	-30665.539	-30665.538672	-30665.539	-30665.538672	-30665.539
g05	5126.4981	5126.498096	5126.497	5126.498096	5126.497	5126.498096	5126.497
g06	-6961.8138	-6961.813876	-6961.814	-6961.813876	-6961.814	-6961.813876	-6961.814
g07	24.306209	24.306211	24.306	24.306390	24.306	24.307496	24.306
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	680.630057	681.553
g10	7049.3307	7049.248435	7049.248	7049.318269	7049.266	7049.910984	7049.617
g11	0.750000	0.749999	0.75	0.749999	0.75	0.749999	0.75
g12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
g13	0.053950	0.053950	0.053941	0.054029	0.069336	0.0546054	0.438803

8.4 Comparison COPSO - ISRES.

Runarsson and Yao proposed the Stochastic Ranking algorithm for constrained optimization [36], and recently an improved version called “Improved Stochastic Ranking Evolution Strategy”, (ISRES), which is the state of the art [37]. Experiments for test problems g14 through g24 were developed using the ISRES's code available at Runarsson's page. The parameters used were the same to the suggested by the authors [37]. The comparison is shown in Table 5. Both algorithms performed the same number of fitness function evaluations, 350,000. Note that ISRES finds the best values for problems g07, g10, g17, g19 and g23 test problems. Also, ISRES average is closer to the optimum value and is better than COPSO in problems g07, g10, g17 and g23. But COPSO is better in problems g02 and g13, where it finds the optimum in all 30 runs. In test problem g21, COPSO found feasible solutions in all 30 runs, whereas ISRES only had 5 successful runs. Both COPSO and ISRES were unable to find feasible solutions for test problems g20 and g22.

Table 5: Comparison of COPSO and ISRES on the benchmark problems

		Best Result		Mean Result		Worst Result	
TF	**Optimal**	**COPSO**	**ISRES**	**COPSO**	**ISRES**	**COPSO**	**ISRES**
g01	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803619	-0.803619	-0.801320	-0.782715	-0.786566	-0.723591
g03	-1.000000	-1.000005	-1.001	-1.000005	-1.001	-1.000003	-1.001
g04	-30665.539	-30665.53867	-30665.539	-30665.53867	-30665.539	-30665.53867	-30665.539
g05	5126.4981	5126.498096	5126.497	5126.498096	5126.497	5126.498096	5126.497
g06	-6961.8138	-6961.813876	-6961.814	-6961.813876	-6961.814	-6961.813876	-6961.814
g07	24.306209	24.306209	24.306	24.306212	24.306	24.306219	24.306
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
g09	680.630057	680.630057	680.630	680.630057	680.630	680.630057	680.630
g10	7049.248	7049.248020	7049.248	7049.250087	7049.25	7049.263662	7049.27
g11	0.750000	0.749999	0.750	0.749999	0.750	0.749999	0.750
g12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
g13	0.053950	0.053950	0.053942	0.053950	0.066770	0.053965	0.438803
g14	-47.761	-47.761108	-47.761129	-47.741430	-47.759250	-47.670921	-47.735569
g15	961.715	961.715171	961.715171	961.715171	961.715171	961.715171	961.715171
g16	-1.905	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155	-1.905155
g17	8876.98068	8856.502344	8889.9003	8877.812811	8889.9442	8941.344349	8890.9516
g18	-0.8660	-0.866025	-0.866025	-0.866001	-0.866025	-0.865568	-0.866025
g19	32.386	32.349645	32.348689	32.411596	32.374095	32.571543	32.644735
g20	0.967	*0.204095	NA	*0.212003	NA	*0.233281	NA
g21	193.778349	205.852693	193.785034	273.298016	220.056989	303.454837	325.144812
g22	12812.500	*157.513639	NA	*5376.226516	NA	*18732.7838	NA
g23	-400.0551	-361.856637	-400.000551	-138.407772	-321.342939	3.775736	-47.842844
g24	-5.508	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013	-5.508013

* Infeasible solution, NA Not available

8.5 Comparison COPSO - SMES.

Mezura and Coello proposed the “Simple Multimember Evolutionary Strategy”, (SMES), which worked reasonable well on the first 13 problems but had a weak performance on the new problems (g14 through g23), mainly due to reduced exploration [17]. In Table 6 we show the comparison of COPSO and SMES. In this case both algorithms performed 240,000 fitness function evaluations. It can be seen that COPSO is clearly better than SMES in problems g05, g07, g10, g13, g14, g15, g17, g19, g21 and g23. Although the best values reported for the rest of the problems are comparable, COPSO outperforms SMES in the average results for problems g05, g06, g07, g10, g13, g14, g15, g17, g18, g19, g21 and g23. COPSO and SMES were unable to find feasible solutions for test problems g20 and g22. But, COPSO finds feasible solutions for test problems g17, g21 and g23, where SMES could not find feasible solutions in any single run.

Table 6: Results of COPSO and SMES for benchmark problems

		Best Result		Mean Result		Worst Result	
TF	**Optimal**	**COPSO**	**SMES**	**COPSO**	**SMES**	**COPSO**	**SMES**
g01	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000	-15.000000
g02	-0.803619	-0.803618	-0.803601	-0.800092	-0.785238	-0.785265	-0.751322
g03	-1.000000	-1.000005	-1.000000	-1.000004	-1.000000	-1.000003	-1.000000
g04	-30665.539	-30665.53867	-30665.539	-30665.53867	-30665.539	-30665.53867	-30665.539
g05	5126.4981	5126.498096	5126.599	5126.498096	5174.492	5126.498096	5304.167
g06	-6961.8138	-6961.813876	-6961.814	-6961.813876	-6961.284	-6961.813876	-6952.482
g07	24.306209	24.306211	24.327	24.306312	24.475	24.306539	24.843
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
g09	680.630057	680.630057	680.632	680.630057	680.643	680.630057	680.719
g10	7049.248	7049.248871	7051.903	7049.278821	7253.047	7049.668593	7638.366
g11	0.750000	0.749999	0.750000	0.749999	0.750000	0.749999	0.750000
g12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
g13	0.053950	0.053950	0.053986	0.053986	0.166385	0.054191	0.468294
g14	-47.761	-47.760600	-47.535	-47.617510	-47.368	-47.392351	-47.053
g15	961.715	961.715171	*961.698	961.715171	963.922	961.715171	967.787
g16	-1.905	-1.905155	-1.905	-1.905155	-1.905	-1.905155	-1.905
g17	8876.98068	8860.030895	*8890.1826	8886.641583	*8954.1364	8958.883372	*9163.6767
g18	-0.8660	-0.866025	-0.866	-0.840455	-0.716	-0.674349	-0.648
g19	32.386	32.351376	34.223	32.616663	37.208	33.782773	41.251
g20	0.967	*0.204095	*0.211364	*0.212003	*0.251130	*0.233281	*0.304414
g21	193.778349	236.928359	*347.9809	313.456855	*678.3924	690.568587	*985.7821
g22	12812.500	*157.513639	*2340.6166	*5376.2265	*9438.2549	*18732.7838	*17671.5351
g23	-400.0551	-369.765012	*-1470.1525	-114.136118	*-363.5082	199.238633	*177.2526
g24	-5.508	-5.508013	-5.508	-5.508013	-5.508	-5.508013	-5.507

* Infeasible solution

9 Engineering optimization problems

A set of five problems, frequently used by the specialized literature, have been chosen to compare COPSO against the results of several authors who have approached the problem via PSO. We also provide results from E. Mezura who used a state of the art differential evolution algorithm [38] (thus the reader may compare solutions from different algorithms). This section is organized by showing the comparisons between COPSO and each author's approach (a brief review of the work of all authors is provided in Section 3). Each comparison includes all possible problems reported. The full description of the five problems can be found in Appendix B. The problems chosen are the following:

1. Problem E01: Design of a Welded Beam
2. Problem E02: Design of a Pressure Vessel
3. Problem E03: Minimization of the Weight of a Tension/Compression Spring
4. Problem E04: Minimization of the Weight of a Speed Reducer
5. Problem E05: Himmelblau's Nonlinear Optimization Problem

9.1 Comparison COPSO and Mezura's Differential Evolution

In Table 7 we compare the results of COPSO and Mezura's algorithm [38]. The number of fitness function evaluations for both algorithms was set to 30,000 for all engineering design problems. COPSO shows better performance than Mezura's approach in engineering problems E01, E02 and E03 (the results presented by Mezura for engineering problem E04 are clearly unfeasible since the solution for variable x_2 is outside of the search space). The solution vector is showed in Table 13.

Table 7: The best results of COPSO and Mezura's algorithm for problems E01, E02, E03, and E04

		Best Result		Mean Result		St. Dev.	
TF	**Optimal**	**COPSO**	**Mezura**	**COPSO**	**Mezura**	**COPSO**	**Mezura**
E01	1.724852	1.724852	1.724852	1.724881	1.777692	1.2661E-05	8.8E-2
E02	6059.7143	6059.714335	6059.7143	6071.013366	6379.938037	15.101157	210
E03	0.012665	0.012665	0.012689	0.012666	0.013165	1.2803E-06	3.9E-4
E04	NA	2996.372448	*2996.348094	2996.408525	*2996.348094	0.028671	0

* Unfeasible solution
NA Not available

9.2 Comparison COPSO and Hu-Eberhart PSO

Hu and Eberhart solved four problems of the list. In Table 8 we show the results of COPSO and HE-PSO (Hu and Eberhart's PSO) [20]. The number of fitness function evaluations performed by HE-PSO were 200,000. Therefore, COPSO was also set to 200,000 as to have a fair comparison. COPSO is better than HE-PSO in engineering problem E03 (the result presented by HE-PSO in engineering problem E02 seems abnormal because the solution vector is equal to COPSO and Mezura's Differential Evolution.) The solution vectors are showed in Table 11.

Table 8: The best result of COPSO and HE-PSO on problems E01, E02, E03, and E05

		Best Result		Mean Result		St. Dev.	
TF	**Optimal**	**COPSO**	**HE-PSO**	**COPSO**	**HE-PSO**	**COPSO**	**HE-PSO**
E01	1.724852	1.724852	1.72485	1.724852	1.72485	0	0
E02	6059.7143	6059.714335	6059.131296	6059.714335	NA	0	NA
E03	0.012665	0.012665	0.0126661409	0.012665	0.012718975	0	6.446E-5
E05	NA	-31025.560242	-31025.56142	-31025.560242	-31025.56142	0	0

NA Not available

9.3 Comparison COPSO and Improved PSO of He-Prempain-Wu

The improved PSO algorithm of He, Prempain and Wu (HPW-PSO) is compared next. In Table 9 we show the results of COPSO and HPW-PSO [23]. The number of fitness function evaluations performed by HPW-PSO were 30,000 for engineering problems E01, E02, E03; and 90,000 for test problem g04. The same number of fitness evaluations were used by COPSO. The performance presented by COPSO is clearly better that HPW-PSO's performance in all problems.

Table 9: The best result of COPSO and HPW-PSO on problems E01, E02, E03, and g04

		Best Result		Mean Result		St. Dev.	
TF	**Optimal**	**COPSO**	**HPW-PSO**	**COPSO**	**HPW-PSO**	**COPSO**	**HPW-PSO**
E01	1.724852	1.724852	2.38095658	1.724881	2.381932	1.2661E-05	5.239371E-3
E02	6059.7143	6059.714335	6059.7143	6071.013366	6289.92881	15.101157	305.78
E03	0.012665	0.012665	0.01266528	0.012666	0.01270233	1.2803E-06	4.12439E-5
g04	-30665.539	-30665.538672	-30665.539	-30665.538672	-30643.989	0	70.043

9.4 Best solution vector found by each approach

In Table 10 we compare the solution vectors found for engineering design problem E01 (welded beam design).

In Table 11 we show the solution vector for engineer problem E02 (pressure vessel design).

Table 10: Solution vector for problem E01 (design of a welded beam)

	Best Solutions			
	COPSO	**Mezura**	**HE-PSO**	**HPW-PSO**
x_1	0.205730	0.205730	0.20573	0.24436898
x_2	3.470489	3.470489	3.47049	6.21751974
x_3	9.036624	9.036624	9.03662	8.29147139
x_4	0.205730	0.205730	0.20573	0.24436898
$g_1(x)$	-1.818989E-12	0.0	0.0	-5741.17693313
$g_2(x)$	-7.275957E-12	0.00002	0.0	-0.00000067
$g_3(x)$	9.4368957E-16	0.0	-5.5511151E-17	0.0
$g_4(x)$	-3.432983	-3.432984	-3.432983	-3.02295458
$g_5(x)$	-0.080729	-0.080730	-0.0807296	-0.11936898
$g_6(x)$	-0.235540	-0.235540	-0.2355403	-0.23424083
$g_7(x)$	-9.094947E-13	0.000001	-9.094947E-13	-0.00030900
$f(x)$	1.724852	1.724852	1.72485	2.3809565827

Table 11: Solution vector for problem E02 (design of a pressure vessel)

	Best Solutions			
	COPSO	**Mezura**	**HE-PSO**	**HPW-PSO**
x_1	0.8125	0.8125	0.8125	0.8125
x_2	0.4375	0.4375	0.4375	0.4375
x_3	42.098446	42.098446	42.09845	42.0984456
x_4	176.636596	176.636596	176.6366	176.63659584
$g_1(x)$	9.479179E-16	0.0	0.0	0.0
$g_2(x)$	-0.0358808	-0.035880	-0.03588	-0.03588083
$g_3(x)$	-3.342393E-10	0.0	-5.8208E-11	0.0
$g_4(x)$	-63.36340	-63.363404	-63.3634	-63.36340416
$f(x)$	6059.714335	6059.7143	6059.131219	6059.7143

In Table 12 we show the solution vectors found for engineering design problem E03 (tension/compression spring design).

Table 12: Solution vector found for problem E03 (design of a tension/compression spring)

	Best Solutions			
	COPSO	**Mezura**	**HE-PSO**	**HPW-PSO**
x_1	0.05168908	0.052836	0.051466369	0.05169040
x_2	0.35671831	0.384942	0.351383949	0.35674999
x_3	11.28893209	9.807729	11.60865920	11.28712599
$g_1(x)$	-6.728016E-15	-0.000001	-0.003336613	-0.00000449
$g_2(x)$	-3.199480E-16	-0.000000	-1.0970128E-4	0.0
$g_3(x)$	-4.053786	-4.106146	-4.0263180998	-4.05382661
$g_4(x)$	-0.727728	-0.708148	-0.7312393333	-0.72770641
$f(x)$	0.012665	0.012689	0.0126661409	0.0126652812

In Table 13 we show the solution vector for engineering problem E04 (minimization of the weight of a speed reducer).

In Table 14 we show the solution vector for engineering design problem E05 (Himmelblau's nonlinear optimization).

10 Final Remarks

This work, proposes a new algorithm called COPSO. It has shown high performance in constraint optimization problems of linear or nonlinear nature. The experimental results are highly competitive with respect

Table 13: Solution vector found for problem E04 (design of a speed reducer)

	Best Solutions	
	COPSO	**Mezura**
x_1	3.5	3.499999
x_2	0.7	0.699999
x_3	17	17
x_4	7.3	7.3
x_5	7.8	7.8
x_6	3.3502146	3.350215
x_7	5.2866832	5.286683
$g_1(x)$	-0.07391528	-0.073915
$g_2(x)$	-0.19799852	-0.197998
$g_3(x)$	-0.49917224	-0.499172
$g_4(x)$	-0.90464390	-0.901472
$g_5(x)$	-1.502162E-16	-0.000000
$g_6(x)$	-4.586717E-16	-0.000000
$g_7(x)$	-0.7025	-0.702500
$g_8(x)$	-6.342582E-17	0.000000
$g_9(x)$	-0.58333333	-0.583333
$g_10(x)$	-0.05132575	-0.051325
$g_11(x)$	-3.995827E-16	-0.010852
$f(x)$	2996.348165	2996.348094

Table 14: Best solutions found for problem E05 (Himmelblau's nonlinear optimization problem)

	Best Solutions	
	COPSO	**HE-PSO**
x_1	78	78
x_2	33	33
x_3	27.070997	27.070997
x_4	45	45
x_5	44.969242	44.96924255
$g_1(x)$	-4.378442E-15	0.0
$g_2(x)$	-92	-92
$g_3(x)$	-9.595215	-9.595215
$g_4(x)$	-10.404784	-10.404784
$g_5(x)$	-5	-5
$g_6(x)$	6.730727E-16	0.0
$f(x)$	-31025.560242	-31025.56142

to the state-of-the-art algorithms. Three important contributions of COPSO are worth to mention: A new neighborhood structure for PSO, the incorporation of perturbation operators without modifying the essence of the PSO, and a special handling technique for equality constraints.
The first contribution is the singly-linked neighborhood structure. It slows down the convergence of the flock, breaking the double-link that exists between the particles using the original ring neighborhood structure. COPSO implements a singly-linked ring with a neighborhood of size $n = 2$, but a general algorithm to build neighborhoods of size n is given.

Another relevant idea developed by COPSO, is the perturbation of the target to keep flock's diversity and space exploration. Two perturbation operators, C-perturbation and M-perturbation are applied to the $PBest$. It is equivalent to perturb the particle's memory and not its behavior (as it is performed by other approaches, that tend to destroy the flock's organization capacity).

The last property of COPSO is its special technique to handle equality constraints. It is performed through a external file that stores the real amount of equality constraint violation of any particle (remember that COPSO uses a dynamic tolerance value to allow unfeasible particles). The external file helps to keep the flock near the feasible region.

The results on the benchmark and engineering problems provide evidence that COPSO is highly competitive. Although, it should be improved to handle problems with a higher number of equality constraints, COPSO performed very well at solving the current state-of-the-art problems.

Acknowledgements

The second author acknowledges support from National Council of Science and Technology, CONACYT, to pursue graduate studies at the Centre for Research in Mathematics.

References

[1] Millonas, M. M., "Swarms, Phase Transitions, and Collective Intelligence," *Artificial Life III*, Addison-Wisley, 1994, pp. 417–445.

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

[3] Angeline, P., "Evolutionary Optimization versus Particle Swarm Optimization: philosophy and performance differences," *Proceedings of the Seventh Annual Conference on Evolutionary Programming, EP VII*, Lecture Notes in Computer Science No. 1447, March 1998, pp. 601–610.

[4] Shi, Y. and Eberhart, R., "A Modified Particle Swarm Optimizer," *Proceedings of the IEEE International Conference on Evolutionary Computation*, IEEE Press, November 1998, pp. 69–73.

[5] Eberhart, R. C. and Shi, Y., "Comparison between genetic algorithms and particle swarm optimization," *Proceedings of the Seventh Annual Conference on Evolutionary Programming, EP VII*, Lecture Notes in Computer Science No. 1447, March 1998, pp. 611–616.

[6] Lvbjerg, M., Rasmussen, T., and Krink, T., "Hybrid particle swarm optimiser with breeding and subpopulations," *Proceedings of the 2001 Genetic and Evolutionary Computation Conference*, July 2001.

[7] Settles, M. and Soule, T., "Breeding Swarms: A GA/PSO Hybrid," *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, Vol. 1, ACM Press, June 2005, pp. 161–168.

[8] Das, S., Konar, A., and Chakraborty, U. K., "Improving particle swarm optimization with differentially perturbed velocity," *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, ACM Press, June 2005, pp. 177–184.

[9] Storn, R. and Price, K., "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces," *Technical Report TR-95-012, International Computer Science Institute*, 1995.

[10] Storn, R., "System Design by Constraint Adaptation and Differential Evolution," *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 1, April 1999, pp. 22–34.

[11] Krink, T., Vesterstrom, J., and Riget, J., "Particle Swarm Optimization with Spatial Particle Extension," *Proceedings of the 2002 Congress on Evolutionary Computation, CEC2002*, Vol. 2, IEEE, May 2002, pp. 1474–1479.

[12] Blackwell, T. M., "Particle swarms and population diversity," *Soft Computing*, Vol. 9, No. 11, November 2005, pp. 793–802.

[13] van den Bergh, F., *An Analysis of Particle Swarm Optimizers*, Ph.D. thesis, University of Pretoria, South Africa, 2002.

[14] van den Bergh, F. and Engelbrecht, A., "A new locally convergent particle swarm optimizer," *Proceedings of the IEEE Conference on Systems, Man, and Cybernetics*, Vol. 3, IEEE Press, October 2002, p. 6.

[15] Hamida, S. B. and Petrowski, A., "The need for improving the exploration operators for constrained optimization problems," *Proceedings of the Congress on Evolutionary Computation. CEC-2000*, IEEE, July 2000, pp. 1176 – 1183.

[16] Hernandez-Aguirre, A., Botello, S., Coello, C., Lizarraga, G., and Mezura, E., "Handling constraints using multiobjective optimization concepts," *International Journal for Numerical Methods in Engineering*, Vol. 59, No. 13, 2004, pp. 1989–2017.

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

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

[19] Parsopoulos, K. and Vrahatis, M., "Particle swarm optimization method for constrained optimization problems," 2002.

[20] Hu, X., Eberhart, R., and Shi, Y., "Engineering optimization with particle swarm," 2003.

[21] Hu, X. and Eberhart, R., "Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization," *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics, SCI2002*, Vol. 5, IIIS, July 2002.

[22] Coath, G. and Halgamuge, S. K., "A comparison of Constraint-Handling Methods for the Application of Particle Swarm Optimization to Constrained Nonlinear Optimization Problems," *Proceedings of the 2003 Congress on Evolutionary Computation*, IEEE, December 2003, pp. 2419–2425.

[23] He, S., Prempain, E., and Wu, Q., "An Improved Particle Swarm Optimizer for Mechanical Design Optimization Problems," *Engineering Optimization*, Vol. 36, No. 5, October 2004, pp. 585–605.

[24] Toscano, G. and Coello, C., "A Constraint-Handling Mechanism for Particle Swarm Optimization," *Proceedings of the 2004 Congress on Evolutionary Computation*, Vol. 2, IEEE, June 2004, pp. 1396–1403.

[25] Deb, K., "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, Vol. 186, No. 2-4, 2000, pp. 311–338.

[26] Zhang, J. and Xie, F., "DEPSO: Hybrid Particle Swarm with Differential Evolution Operator," *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, IEEE, October 2003, pp. 3816–3821.

[27] Zhang, W.-J., Xie, X.-F., and Bi, D.-C., "Handling bloundary constraints for numerical optimization by Particle Swarm flying in periodic search space," *Proceedings of the 2004 Congress on Evolutionary Computation*, Vol. 2, IEEE, June 2004, pp. 2307–2311.

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

[29] Mendes, R., Kennedy, J., and Neves, J., "The Fully Informed Particle Swarm: Simpler, Maybe Better," *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3, June 2004, pp. 204–210.

[30] Kennedy, J., "Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance," *IEEE Congress on Evolutionary Computation*, IEEE, July 1999, pp. 1931–1938.

[31] Kennedy, J. and Mendes, R., "Population Structure and Particle Swarm Performance," *Proceedings of the 2002 Congress on Evolutionary Computation, CEC2002*, Vol. 2, IEEE, May 2002, pp. 1671–1676.

[32] Engelbrecht, A. P., *Fundamentals of Computational Swarm Intelligence*, John Wiley and Sons, Chichester, England, 2005.

[33] Price, K., Storn, R., and Lampinen, J., *Differential Evolution: A practical approach to global optimization*, Springer Verlag, Berlin, Germany, 2005.

[34] Mezura, E. and Coello, C., "Identifying on-line behavior and some sources of difficulty in two competitive approaches for constrained optimization," *Proceedings of the Conference on Evolutionary Computation, CEC 2005*, IEEE, September 2005, pp. 56–63.

[35] Mezura, E., Velazquez, J., and Coello, C., "Promising Infeasibility and Multiple Offspring Incorporated to Differential Evolution for Constrained Optimization," *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2006*, Association for Computing Machinery, June 2005, pp. 225–232.

[36] Runarsson, T. and Yao, X., "Stochastic Ranking for Constrained Evolutionary Optimization," *IEEE Transactions on Evolutionary Computation*, Vol. 4, No. 3, September 2000, pp. 284–294.

[37] Runarsson, T. and Yao, X., "Search Biases in Constrained Evolutionary Optimization," *IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews*, Vol. 35, No. 2, May 2005, pp. 233–243.

[38] Mezura, E. and Coello, C., "Useful Infeasible Solutions in Engineering Optimization with Evolutionary Algorithms," *Proceedings of the 4th Mexican International Conference on Artificial Intelligence, MICAI 2005*, Lecture Notes on Artificial Intelligence No. 3789, November 2005, pp. 652–662.

Appendix A: Benchmark functions

This first 13 functions conform the well known benchmark of Runarsson and Yao [36].

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:

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

where the bounds are $0 \leq x_i \leq 1$ $(i = 1, \ldots, 9)$, $0 \leq x_i \leq 100$ $(i = 10, 11, 12)$ and $0 \leq x_{13} \leq 1$. 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_7, g_8 and g_9 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:

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

where $n = 20$ and $0 \leq x_i \leq 10$ $(i = 1, \ldots, n)$. 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 $0 \leq x_i \leq 1$ $(i = 1, \ldots, n)$. 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:

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

where: $78 \leq x_1 \leq 102$, $33 \leq x_2 \leq 45$, $27 \leq x_i \leq 45$ $(i = 3, 4, 5)$. 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:

$$\begin{aligned} g_1(\vec{x}) &= -1 + 0.0025(x_4 + x_6) \leq 0 \\ g_2(\vec{x}) &= -1 + 0.0025(x_5 + x_7 - x_4) \leq 0 \\ g_3(\vec{x}) &= -1 + 0.01(x_8 - x_5) \leq 0 \\ g_4(\vec{x}) &= -x_1 x_6 + 833.33252 x_4 + 100 x_1 \\ &- 83333.333 \leq 0 \\ g_5(\vec{x}) &= -x_2 x_7 + 1250 x_5 + x_2 x_4 - 1250 x_4 \leq 0 \\ g_6(\vec{x}) &= -x_3 x_8 + 1250000 + x_3 x_5 - 2500 x_5 \leq 0 \end{aligned}$$

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:

$$g_1(\vec{x}) = (x_1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \leq 0 \quad (3)$$

where: $0 \leq x_i \leq 10$ $(i = 1, 2, 3)$ and p, q, r=1,2,...,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: $f(\vec{x}) = e^{x_1 x_2 x_3 x_4 x_5}$
subject to:

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

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 $f(x^*) = 0.0539498$.

Now, we list the new test problems proposed by Mezura and Coello [34].

1. **g14** Minimize: $f(\vec{x}) = \sum_{i=1}^{10} x_i \left(c_i + ln \frac{x_i}{\sum_{j=1}^{10} x_j} \right)$
where $c_1 = -6.089$ $c_2 = -17.164$ $c_3 = -34.054$ $c_4 = -5.914$ $c_5 = -24.721$ $c_6 = -14.986$ $c_7 = -24.100$ $c_8 = -10.708$ $c_9 = -26.662$ $c_{10} = -22.179$
subject to:

$$\begin{aligned} h_1(\vec{x}) &= x_1 + 2x_2 + 2x_3 + x_6 + x_{10} - 2 = 0 \\ h_2(\vec{x}) &= x_4 + 2x_5 + x_6 + x_7 - 1 = 0 \\ h_3(\vec{x}) &= x_3 + x_7 + x_8 + 2x_9 + x_{10} - 1 = 0 \end{aligned}$$

where the bounds are $0 \leq x_i \leq 1$ $(i = 1, \ldots, 10)$. The global optimum is at $x^* = (0.0407, 0.1477, 0.7832, 0.0014, 0.4853, 0.0007, 0.0274, 0.0180, 0.03$ where $f(x^*) = -47.761$.

2. **g15** Minimize: $f(\vec{x}) = 1000 - x_1^2 - 2x_2^2 - x_3^2 - x_1 x_2 - x_1 x_3$
subject to:

$$\begin{aligned} h_1(\vec{x}) &= x_1^2 + x_2^2 + x_3^2 - 25 = 0 \\ h_2(\vec{x}) &= 8x_1 + 14x_2 + 7x_3 - 56 = 0 \end{aligned}$$

where the bounds are $0 \leq x_i \leq 10$ $(i = 1, \ldots, 3)$. The global optimum is at $x^* = (3.512, 0.217, 3.552)$ where $f(x^*) = 961.715$.

3. **g16** Maximize: $f(\vec{x}) = 0.0000005843 y_{17} - 0.000117 y_{14} - 0.1365 - 0.00002358 y_{13} - 0.000001502 y_{16} - 0.0321 y_{12} - 0.004324 y_5 - 0.0001 \frac{c_{15}}{c_{16}} - 37.48 \frac{y_2}{c_{12}}$ where

$$
\begin{aligned}
y_1 &= x_2 + x_3 + 41.6 \\
c_1 &= 0.024 x_4 - 4.62 \\
y_2 &= \frac{12.5}{c_1} + 12 \\
c_2 &= 0.0003535 x_1^2 + 0.5311 x_1 + 0.08705 y_2 x_1 \\
c_3 &= 0.052 x_1 + 78 + 0.002377 y_2 x_1 \\
y_3 &= \frac{c_2}{c_3} \\
y_4 &= 19 y_3 \\
c_4 &= 0.04782(x_1 - y_3) + \frac{0.1956(x_1 - y_3)^2}{x_2} \\
&+ 0.6376 y_4 + 1.594 y_3 \\
c_5 &= 100 x_2 \\
c_6 &= x_1 - y_3 - y_4 \\
c_7 &= 0.95 - \frac{c_4}{c_5} \\
y_5 &= c_6 c_7 \\
y_6 &= x_1 - y_5 - y_4 - y_3 \\
c_8 &= (y_5 + y_4) 0.995 \\
y_7 &= \frac{c_8}{y_1} \\
y_8 &= \frac{c_8}{3798} \\
c_9 &= y_7 - \frac{0.0663 y_7}{y_8} - 0.3153 \\
y_9 &= \frac{96.82}{c_9} + 0.321 y_1 \\
y_{10} &= 1.29 y_5 + 1.258 y_4 + 2.29 y_3 + 1.71 y_6 \\
y_{11} &= 1.71 x_1 - 0.452 y_4 + 0.58 y_3 \\
c_{10} &= \frac{12.3}{752.3} \\
c_{11} &= (1.75 y_2)(0.995 x_1)
\end{aligned}
$$

$$
\begin{aligned}
c_{12} &= 0.995 y_{10} + 1998 \\
y_{12} &= c_{10} x_1 + \frac{c_{11}}{c_{12}} \\
y_{13} &= c_{12} - 1.75 y_2 \\
y_{14} &= 3623 + 64.4 x_2 + 58.4 x_3 + \frac{146312}{y_9 + x_5} \\
c_{13} &= 0.995 y_{10} + 60.8 x_2 + 48 x_4 - 0.1121 y_{14} - 5095 \\
y_{15} &= \frac{y_{13}}{c_{13}} \\
y_{16} &= 148000 - 331000 y_{15} + 40 y_{13} - 61 y_{15} y_{13} \\
c_{14} &= 2324 y_{10} - 28740000 y_2 \\
y_{17} &= 14130000 - 1328 y_{10} - 531 y_{11} + \frac{c_{14}}{c_{12}} \\
c_{15} &= \frac{y_{13}}{y_{15}} - \frac{y_{13}}{0.52} \\
c_{16} &= 1.104 - 0.72 y_{15} \\
c_{17} &= y_9 + x_5
\end{aligned}
$$

subject to:

$$
\begin{aligned}
g_1(\vec{x}) &= y_4 - \frac{0.28}{0.72} y_5 \geq 0 \\
g_2(\vec{x}) &= 1.5 x_2 - x_3 \geq 0 \\
g_3(\vec{x}) &= 21 - 3496 \frac{y_2}{c_{12}} \geq 0 \\
g_4(\vec{x}) &= \frac{62212}{c_{17}} - 110.6 - y_1 \geq 0
\end{aligned}
$$

$$
\begin{array}{rcccl}
213.1 & \leq & y_1 & \leq & 405.23 \\
17.505 & \leq & y_2 & \leq & 1053.6667 \\
11.275 & \leq & y_3 & \leq & 35.03 \\
214.228 & \leq & y_4 & \leq & 665.585 \\
7.458 & \leq & y_5 & \leq & 584.463 \\
0.961 & \leq & y_6 & \leq & 265.916 \\
1.612 & \leq & y_7 & \leq & 7.046 \\
0.146 & \leq & y_8 & \leq & 0.222 \\
107.99 & \leq & y_9 & \leq & 273.366 \\
922.693 & \leq & y_{10} & \leq & 1286.105 \\
926.832 & \leq & y_{11} & \leq & 1444.046 \\
18.766 & \leq & y_{12} & \leq & 537.141 \\
1072.163 & \leq & y_{13} & \leq & 3247.039 \\
8961.448 & \leq & y_{14} & \leq & 26844.086 \\
0.063 & \leq & y_{15} & \leq & 0.386 \\
71084.33 & \leq & y_{16} & \leq & 140000 \\
2802713 & \leq & y_{17} & \leq & 12146108
\end{array}
$$

where the bounds are $704.4148 \leq x_1 \leq 906.3855$, $68.6 \leq x_2 \leq 288.88$, $0 \leq x_3 \leq 134.75$, $193 \leq x_4 \leq 287.0966$ and $25 \leq x_5 \leq 84.1988$. The global optimum is at $x^* = (705.060, 68.600, 102.900, 282.341, 35.627)$ where $f(x^*) = 1.905$.

4. **g17** Minimize: $f(\vec{x}) = f_1(x_1) + f_2(x_2)$
   where

$$
f_1(x_1) = \left\{ \begin{array}{cc} 30x_1 & 0 \leq x_1 < 300 \\ 31x_1 & 300 \leq x_1 < 400 \end{array} \right\}
$$

$$
f_2(x_1) = \left\{ \begin{array}{cc} 28x_2 & 0 \leq x_2 < 100 \\ 29x_2 & 100 \leq x_2 < 200 \\ 30x_2 & 200 \leq x_2 < 1000 \end{array} \right\}
$$

   subject to:

$$
\begin{array}{rcl}
h_1(\vec{x}) & = & x_1 - 300 + \dfrac{x_3 x_4}{131.078} \cos(1.48577 - x_6) \\
& - & \dfrac{0.90798}{131.078} x_3^2 \cos(1.47588) = 0 \\
h_2(\vec{x}) & = & x_2 + \dfrac{x_3 x_4}{131.078} \cos(1.48477 + x_6) \\
& - & \dfrac{0.90798}{131.078} x_4^2 \cos(1.47588) = 0 \\
h_3(\vec{x}) & = & x_5 + \dfrac{x_3 x_4}{131.078} \sin(1.48477 + x_6) \\
& - & \dfrac{0.90798}{131.078} x_4^2 \sin(1.47588) = 0 \\
h_4(\vec{x}) & = & 200 - \dfrac{x_3 x_4}{131.078} \sin(1.48477 - x_6) \\
& + & \dfrac{0.90798}{131.078} x_3^2 \sin(1.47588) = 0
\end{array}
$$

   where the bounds are $0 \leq x_1 \leq 400$, $0 \leq x_2 \leq 1000$, $340 \leq x_3 \leq 420$, $340 \leq x_4 \leq 420$, $-1000 \leq x_5 \leq 1000$, $0 \leq x_6 \leq 0.5236$. The global optimum is at $x^* = (212.684440144685, 89.1588384165537, 368.447892659317, 409.03379817159, 4.16436988876356, 0.0680394595246655)$ where $f(x^*) = 8876.980680$.

5. **g18** Maximize: $f(\vec{x}) = 0.5(x_1 x_4 - x_2 x_3 + x_3 x_9 - x_5 x_9 + x_5 x_8 - x_6 x_7$
   subject to:

$$
\begin{array}{rcl}
g_1(\vec{x}) & = & 1 - x_3^2 - x_4^2 \geq 0 \\
g_2(\vec{x}) & = & 1 - x_9^2 \geq 0 \\
g_3(\vec{x}) & = & 1 - x_5^2 - x_6^2 \geq 0 \\
g_4(\vec{x}) & = & 1 - x_1^2 - (x_2 - x_9)^2 \geq 0 \\
g_5(\vec{x}) & = & 1 - (x_1 - x_5)^2 - (x_2 - x_6)^2 \geq 0 \\
g_6(\vec{x}) & = & 1 - (x_1 - x_7)^2 - (x_2 - x_8)^2 \geq 0 \\
g_7(\vec{x}) & = & 1 - (x_3 - x_5)^2 - (x_4 - x_6)^2 \geq 0 \\
g_8(\vec{x}) & = & 1 - (x_3 - x_7)^2 - (x_4 - x_8)^2 \geq 0
\end{array}
$$

$$
\begin{aligned}
g_9(\vec{x}) &= 1 - x_7^2 - (x_8 - x_9)^2 \geq 0 \\
g_{10}(\vec{x}) &= x_1 x_4 - x_2 x_3 \geq 0 \\
g_{11}(\vec{x}) &= x_3 x_9 \geq 0 \\
g_{12}(\vec{x}) &= -x_5 x_9 \geq 0 \\
g_{13}(\vec{x}) &= x_5 x_8 - x_6 x_7 \geq 0
\end{aligned}
$$

where the bounds are $-1 \leq x_i \leq 1$ $(i = 1, \ldots, 8)$ and $0 \leq x_9 \leq 1$. The global optimum is at $x^* = (0.9971, -0.0758, 0.5530, 0.8331, 0.9981, -0.0623$ where $f(x^*) = 0.8660$.

6. **g19** Maximize: $f(\vec{x}) = \sum_{i=1}^{10} b_i x_i - \sum_{j=1}^{5} \sum_{i=1}^{5} c_{ij} x_{10+i} x_{10+j} - 2 \sum_{j=1}^{5} d_j x_{10+j}^3$
   subject to:

$$
\begin{aligned}
g_j(\vec{x}) &= 2 \sum_{i=1}^{5} c_{ij} x_{10+i} + 3 d_j x_{10+j}^2 + e_j - \\
&\quad \sum_{i=1}^{10} a_{ij} x_i \geq 0 \qquad j = 1, \ldots, 5
\end{aligned}
$$

(*Note:* The $e_j, c_{ij}, d_j, a_{ij}, b_j$ are given in the Table 15.) where the bounds are $0 \leq x_i \leq 100$ $(i = 1, \ldots, 15)$. The global optimum is at $x^* = (0.0000, 0.0000, 5.1470, 0.0000, 3.0611, 11.8395, 0.0000, 0.0000, 0.1039, 0.0000, 0.3000, 0.3335, 0.4000, 0.4283, 0.2240)$ where $f(x^*) = -32.386$.

Table 15: Data for problem g19

j	1	2	3	4	5
e_j	-15	-27	-36	-18	-12
c_{1j}	30	-20	-10	32	-10
c_{2j}	-20	39	-6	-31	32
c_{3j}	-10	-6	10	-6	-10
c_{4j}	32	-31	-6	39	-20
c_{5j}	-10	32	-10	-20	30
d_j	4	8	10	6	2
a_{1j}	-16	2	0	1	0
a_{2j}	0	-2	0	0.4	2
a_{3j}	-3.5	0	2	0	0
a_{4j}	0	-2	0	-4	-1
a_{5j}	0	-9	-2	1	-2.8
a_{6j}	2	0	-4	0	0
a_{7j}	-1	-1	-1	-1	-1
a_{8j}	-1	-2	-3	-2	-1
a_{9j}	1	2	3	4	5
a_{10j}	1	1	1	1	1

b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
-40	-2	-0.25	-4	-4	-1	-40	-60	5	1

7. **g20** Minimize: $f(\vec{x}) = \sum_{i=1}^{24} a_i x_i$
   subject to:

$$
\begin{aligned}
h_i(\vec{x}) &= \frac{x_{i+12}}{b_{i+12} \sum_{j=13}^{24} \frac{x_j}{b_j}} - \frac{c_i x_i}{40 b_i \sum_{j=1}^{12} \frac{x_j}{b_j}} = 0 \\
&\qquad i = 1, \ldots, 12 \\
h_{13}(\vec{x}) &= \sum_{i=1}^{24} x_i - 1 = 0 \\
h_{14}(\vec{x}) &= \sum_{i=1}^{12} \frac{x_i}{d_i} + 0.7302(530) \left(\frac{14.7}{40} \right) \sum_{i=13}^{24} \frac{x_i}{b_i} \\
&- 1.671 = 0 \\
g_i(\vec{x}) &= -\frac{x_i + x_{i+12}}{\sum_{j=1}^{24} x_j + e_i} \geq 0 \qquad i = 1, 2, 3 \\
g_k(\vec{x}) &= -\frac{x_{k+3} + x_{k+15}}{\sum_{j=1}^{24} x_j + e_k} \geq 0 \qquad k = 4, 5, 6
\end{aligned}
$$

(*Note:* The a_i, b_i, c_i, d_i, e_i are given in the Table 16.) where the bounds are $0 \le x_i \le 1$ ($i = 1, \ldots, 24$). The global optimum is at $x^* = (9.537E-07, 0, 4.215E-03, 1.039E-04, 0, 0, 2.072E-01, 5.979E-01, 1.298E-01, 3.350E-02, 1.711E-02, 8.427E-03, 4.657E-10, 0, 0, 0, 0, 0, 2.868E-04, 1.193E-03, 8.332E-05, 1.239E-04, 2.070E-05, 1.829E-05)$ where $f(x^*) = 0.09670$.

Table 16: Data for problem g20

i	a_i	b_i	c_i	d_i	e_i
1	0.0693	44.094	123.7	31.244	0.1
2	0.0577	58.12	31.7	36.12	0.3
3	0.05	58.12	45.7	34.784	0.4
4	0.20	137.4	14.7	92.7	0.3
5	0.26	120.9	84.7	82.7	0.6
6	0.55	170.9	27.7	91.6	0.3
7	0.06	62.501	49.7	56.708	
8	0.10	84.94	7.1	82.7	
9	0.12	133.425	2.1	80.8	
10	0.18	82.507	17.7	64.517	
11	0.10	46.07	0.85	49.4	
12	0.09	60.097	0.64	49.1	
13	0.0693	44.094			
14	0.0577	58.12			
15	0.05	58.12			
16	0.20	137.4			
17	0.26	120.9			
18	0.55	170.9			
19	0.06	62.501			
20	0.10	84.94			
21	0.12	133.425			
22	0.18	82.507			
23	0.10	46.07			
24	0.09	60.097			

8. **g21** Minimize: $f(\vec{x}) = x_1$
subject to:

$$
\begin{aligned}
g_1(\vec{x}) &= -x_1 + 35x_2^{0.6} + 35x_3^{0.6} \le 0 \\
h_1(\vec{x}) &= -300x_3 + 7500x_5 - 7500x_6 - 25x_4x_5 \\
&+ 25x_4x_6 + x_3x_4 = 0 \\
h_2(\vec{x}) &= 100x_2 + 155.365x_4 + 2500x_7 - x_2x_4 \\
&- 25x_4x_7 - 15536.5 = 0 \\
h_3(\vec{x}) &= -x_5 + \ln(-x_4 + 900) = 0 \\
h_4(\vec{x}) &= -x_6 + \ln(x_4 + 300) = 0 \\
h_5(\vec{x}) &= -x_7 + \ln(-2x_4 + 700) = 0
\end{aligned}
$$

where the bounds are $0 \le x_1 \le 1000$, $0 \le x_2 \le 40$, $0 \le x_3 \le 40$, $100 \le x_4 \le 300$, $6.3 \le x_5 \le 6.7$, $5.9 \le x_6 \le 6.4$ and $4.5 \le x_7 \le 6.25$. The global optimum is at $x^* = (193.7783493, 0, 17.3272116, 100.0156586, 6.684592154, 5.991503693, 6.214545462)$ where $f(x^*) = 193.7783493$.

9. **g22** Minimize: $f(\vec{x}) = x_1$
subject to:

$$
\begin{aligned}
g_1(\vec{x}) &= -x_1 + x_2^{0.6} + x_3^{0.6} + x_4^{0.6} \le 0 \\
h_1(\vec{x}) &= x_5 - 100000x_8 + 1E07 = 0 \\
h_2(\vec{x}) &= x_6 + 100000x_8 - 100000x_9 = 0 \\
h_3(\vec{x}) &= x_7 + 100000x_9 - 5E07 = 0 \\
h_4(\vec{x}) &= x_5 + 100000x_{10} - 3.3E07 = 0 \\
h_5(\vec{x}) &= x_6 + 100000x_{11} - 4.4E07 = 0 \\
h_6(\vec{x}) &= x_7 + 100000x_{12} - 6.6E07 = 0 \\
h_7(\vec{x}) &= x_5 - 120x_2x_{13} = 0 \\
h_8(\vec{x}) &= x_6 - 80x_3x_{14} = 0 \\
h_9(\vec{x}) &= x_7 - 40x_4x_{15} = 0 \\
h_{10}(\vec{x}) &= x_8 - x_{11} + x_{16} = 0 \\
h_{11}(\vec{x}) &= x_9 - x_{12} + x_{17} = 0 \\
h_{12}(\vec{x}) &= -x_{18} + \ln(x_{10} - 100) = 0
\end{aligned}
$$

$$
\begin{aligned}
h_{13}(\vec{x}) &= -x_{19} + \ln(-x_8 + 300) = 0 \\
h_{14}(\vec{x}) &= -x_{20} + \ln(x_{16}) = 0 \\
h_{15}(\vec{x}) &= -x_{21} + \ln(-x_9 + 400) = 0 \\
h_{16}(\vec{x}) &= -x_{22} + \ln(x_{17}) = 0 \\
h_{17}(\vec{x}) &= -x_8 - x_{10} + x_{13}x_{18} - x_{13}x_{19} + 400 = 0 \\
h_{18}(\vec{x}) &= x_8 - x_9 - x_{11} + x_{14}x_{20} - x_{14}x_{21} + 400 = 0 \\
h_{19}(\vec{x}) &= x_9 - x_{12} - 4.60517x_{15} + x_{15}x_{22} + 100 = 0
\end{aligned}
$$

where the bounds are $0 \leq x_1 \leq 20000$, $0 \leq x_i \leq 1E06$ $i = (2,3,4)$, $0 \leq x_i \leq 4E07$ $i = (5,6,7)$, $100 \leq x_8 \leq 299.99$, $100 \leq x_9 \leq 399.99$, $100.01 \leq x_{10} \leq 300$, $100 \leq x_{11} \leq 400$, $100 \leq x_{12} \leq 600$, $0 \leq x_{10} \leq 500$ $i = (13,14,15)$, $0.01 \leq x_{16} \leq 300$, $0.01 \leq x_{17} \leq 400$ and $-4.7 \leq x_i \leq 6.25$ $i = (18,\ldots,22)$. The global optimum is at $x^* = (12812.5, 722.1602494, 8628.371755, 2193.749851, 9951396.436, 1884656$ $387.979596, 230.4860356, 251.5343684, 547.979596, 114.8336587, 27.30318607, 127.6585887, 52.020404,$ $160, 4.871266214, 4.610018769, 3.951636026, 2.486605539, 5.075173815)$ where $f(x^*) = 12812.5$.

10. **g23** Minimize: $f(\vec{x}) = -9x_5 - 15x_8 + 6x_1 + 16x_2 + 10(x_6 + x_7)$
    subject to:

$$
\begin{aligned}
h_1(\vec{x}) &= x_1 + x_2 - x_3 - x_4 = 0 \\
h_2(\vec{x}) &= 0.03x_1 + 0.01x_2 - x_9(x_3 + x_4) = 0 \\
h_3(\vec{x}) &= x_3 + x_6 - x_5 = 0 \\
h_4(\vec{x}) &= x_4 + x_7 - x_8 = 0 \\
g_1(\vec{x}) &= x_9x_3 + 0.02x_6 - 0.025x_5 \leq 0 \\
g_2(\vec{x}) &= x_9x_4 + 0.02x_7 - 0.015x_8 \leq 0
\end{aligned}
$$

where the bounds are $0 \leq x_i \leq 300$ $i = (1,2,6)$, $0 \leq x_i \leq 100$ $i = (3,5,7)$, $0 \leq x_i \leq 200$ $i = (4,8)$ and $0.01 \leq x_9 \leq 0.03$. The best known solution has a objective function value of $f(x^*) = -400.0551$.

11. **g24** Minimize: $f(\vec{x}) = -x_1 - x_2$
    subject to:

$$
\begin{aligned}
g_1(\vec{x}) &= -2x_1^4 + 8x_1^3 - 8x_1^2 + x_2 - 2 \leq 0 \\
g_2(\vec{x}) &= -4x_1^4 + 32x_1^3 - 88x_1^2 + 96x_1 + x_2 - 36 \leq 0
\end{aligned}
$$

where the bounds are $0 \leq x_1 \leq 3$, $0 \leq x_2 \leq 4$. The optimum is at $x^* = (2.3295, 3.17846)$ where $f(x^*) = -5.50796$.

Appendix B: Engineering optimization problems

Formal statement of the engineering design problems.

Problem E01: Design of a Welded Beam A welded beam is designed for minimum cost subject to constraints of shear stress (τ), bending stress in the beam (σ), buckling load on the bar (P_c), end deflection of the beam (δ), and side constraints. There are four design variables, $h(x_1), l(x_2), t(x_3), b(x_4)$. The formal statement of the problem is the following:

Minimize: $f(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= \tau(\vec{x}) - \tau_{max} \leq 0 \\ g_2(\vec{x}) &= \sigma(\vec{x}) - \sigma_{max} \leq 0 \\ g_3(\vec{x}) &= x_1 - x_4 \leq 0 \\ g_4(\vec{x}) &= 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \leq 0 \\ g_5(\vec{x}) &= 0.125 - x_1 \leq 0 \\ g_6(\vec{x}) &= \delta(\vec{x}) - \delta_{max} \leq 0 \\ g_7(\vec{x}) &= P - P_c(\vec{x}) \leq 0 \end{aligned}$$

where:

$$\begin{aligned} \tau(\vec{x}) &= \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x^2}{2R} + (\tau'')^2} \\ \tau' &= \frac{P}{\sqrt{2}x_1x_2} \\ \tau'' &= \frac{MR}{J} \\ M &= P\left(L + \frac{x_2}{2}\right) \\ R &= \sqrt{\frac{x_2^2}{2} + \left(\frac{x_1 + x_3}{2}\right)^2} \\ J &= 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\} \\ \sigma(\vec{x}) &= \frac{6PL}{x_4x_3^2} \\ \delta(\vec{x}) &= \frac{4PL^3}{Ex_3^3x_4} \\ P_c(\vec{x}) &= \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3\sqrt{\frac{E}{4G}}}{2L}\right) \end{aligned}$$

where $P = 6000lb$, $L = 14in$, $E = 30x10^6psi$, $G = 12x10^6psi$, $\tau_{max} = 13,600psi$, $\sigma_{max} = 30,000psi$, $\delta_{max} = 0.25in$, and $0.1 \leq x_1 \leq 2.0$, $0.1 \leq x_2 \leq 10.0$, $0.1 \leq x_3 \leq 10.0$ and $0.1 \leq x_4 \leq 2.0$. The best solution founded by Mezura [38] is $x^* = (0.205730, 3.470489, 9.036624, 0.205729)$ where $f(x^*) = 1.724852$.

Problem E02: Design of a Pressure Vessel A cylindrical vessel is capped at both ends by hemispherical heads. The objective is to minimize the total cost, including the cost of the materials forming the welding. There are four design variables: Thickness of the shell $T_s = x_1$, thickness of the head $T_h = x_2$, the inner radius $R = x_3$, and the length of the cylindrical section of the vessel $L = x_4$. T_s and T_h are discrete values which are integer multiples of 0.0625 inch. The formal statement of the problem is the following:

Minimize: $f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$

subject to:

$$\begin{aligned} g_1(\vec{x}) &= -x_1 + 0.0193x_3 \leq 0 \\ g_2(\vec{x}) &= -x_2 + 0.00954x_3 \leq 0 \\ g_3(\vec{x}) &= -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \leq 0 \\ g_4(\vec{x}) &= x_4 - 240 \leq 0 \end{aligned}$$

where $1 \leq x_1 \leq 99$, $1 \leq x_2 \leq 99$, $10 \leq x_3 \leq 200$ and $10 \leq x_4 \leq 200$. The best solution found by Mezura [38] is $x^* = (0.8125, 0.4375, 42.098446, 176.636596)$ where $f(x^*) = 6059.7143$.

Problem E03: Minimization of the Weight of a Tension/Compression Spring This problem consists of minimizing the weight of a tension/compression spring, subject to constraints of minimum deflection, shear stress, surge frequency, and limits on outside diameter and on design variables. The design variables are the wire diameter (x_1), the mean coil diameter, (x_2), and the number of active coils (x_3). The formal statement of the problem is as follows:

Minimize: $f(\vec{x}) = (x_3 + 2)x_2x_1^2$

subject to:

$$g_1(\vec{x}) = 1 - \frac{x_2^3x_3}{71785x_1^4} \leq 0$$

$$
\begin{aligned}
g_2(\vec{x}) &= \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0 \\
g_3(\vec{x}) &= 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0 \\
g_4(\vec{x}) &= \frac{x_2 + x_1}{1.5} - 1 \leq 0
\end{aligned}
$$

where $0.05 \leq x_1 \leq 2.0$, $0.25 \leq x_2 \leq 1.3$ and $2.0 \leq x_3 \leq 15.0$. The best solution founded by He [23] is $x^* = (0.051690, 0.356750, 11.287126)$ where $f(x^*) = 0.012665$.

Problem E04. Minimization of the Weight of a Speed Reducer The weight of the speed reducer is to be minimized subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stresses in the shaft. The variables $x_1, x_2, \ldots, x_7$ are the face width, module of the teeth, number of teeth in the pinion, length of the first shaft between bearings, length of the second shaft between bearings, and the diameter of the first and second shafts. The third variable is integer, the rest of them are continuous. The formal statement of the problem is as follows:

Minimize: $f(\vec{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)$
subject to:

$$
\begin{aligned}
g_1(\vec{x}) &= \frac{27}{x_1x_2^2x_3} - 1 \leq 0 \\
g_2(\vec{x}) &= \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0 \\
g_3(\vec{x}) &= \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0 \\
g_4(\vec{x}) &= \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0 \\
g_5(\vec{x}) &= \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9x10^6}}{110x_6^3} - 1 \leq 0 \\
g_6(\vec{x}) &= \frac{\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5x10^6}}{85x_7^3} - 1 \leq 0 \\
g_7(\vec{x}) &= \frac{x_2x_3}{40} - 1 \leq 0 \\
g_8(\vec{x}) &= \frac{5x_2}{x_1} - 1 \leq 0 \\
g_9(\vec{x}) &= \frac{x_1}{12x_2} - 1 \leq 0 \\
g_{10}(\vec{x}) &= \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0 \\
g_{11}(\vec{x}) &= \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0
\end{aligned}
$$

where $2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq x_4 \leq 8.3$, $7.8 \leq x_5 \leq 8.3$, $2.9 \leq x_6 \leq 3.9$ and $5.0 \leq x_7 \leq 5.5$. The best solution found by Mezura [38] is $x^* = (3.499999, 0.699999, 17, 7.300000, 7.800000, 3.350215, 5.286683)$ where $f(x^*) = 2996.348094$. But, it is unfeasible because $x_2 < 0.7$ in the solution reported by Mezura.

Problem E05: Himmelblau's Nonlinear Optimization Problem This problem was proposed by Himmelblau and similar to problem $g04$ of the benchmark except for the second coefficient of the first constraint. There are five design variables. The problem can be stated as follows:

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

$$
\begin{aligned}
g_1(\vec{x}) &= 85.334407 + 0.0056858x_2x_5 \\
&+ 0.00026x_1x_4 - 0.0022053x_3x_5 - 92 \leq 0 \\
g_2(\vec{x}) &= -85.334407 - 0.0056858x_2x_5 \\
&- 0.00026x_1x_4 + 0.0022053x_3x_5 \leq 0 \\
g_3(\vec{x}) &= 80.51249 + 0.0071317x_2x_5 \\
&+ 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \leq 0 \\
g_4(\vec{x}) &= -80.51249 - 0.0071317x_2x_5 \\
&- 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \leq 0 \\
g_5(\vec{x}) &= 9.300961 + 0.0047026x_3x_5 \\
&+ 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \leq 0 \\
g_6(\vec{x}) &= -9.300961 - 0.0047026x_3x_5 \\
&- 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \leq 0
\end{aligned}
$$

where: $78 \leq x_1 \leq 102$, $33 \leq x_2 \leq 45$, $27 \leq x_i \leq 45$ $(i = 3, 4, 5)$.