A particle swarm paradigm for nonlinear constrained global optimization

A. Ismael F. Vaz and Edite M.G.P. Fernandes

Production and Systems Department - Engineering School

Minho University - Braga - Portugal

{aivaz,emgpf}@dps.uminho.pt

EUME 2005 - Vilnius - Lithuania

19-21 May 2005

Work partially supported by FCT grant POCTI/MAT/58957/2004 and Algoritmi research center

Outline

- Nonlinear constrained global optimization
- The Particle Swarm Paradigm
- The dominance concept
- The proposed algorithm
- Numerical results
- Conclusions

Nonlinear constrained global optimization

We address the following optimization problem

$$\begin{aligned} \min_{x \in R^n} & f(x) \\ s.t. \quad & c^{\mathcal{E}}(x) = 0 \\ & c^{\mathcal{I}}(x) \leq 0 \\ & l \leq x \leq u \end{aligned}$$

where $f(x) : R^n \rightarrow R$ is the objective function, $c^{\mathcal{E}}(x) : R^n \rightarrow R^{m_{\mathcal{E}}}$ are the $m_{\mathcal{E}}$ equality constraints functions, $c^{\mathcal{I}}(x) : R^n \rightarrow R^{m_{\mathcal{I}}}$ are the $m_{\mathcal{I}}$ inequality constraints functions, l and u are the simple bounds on the variables x. We allow $l_i = -\infty$ and/or $u_i = +\infty$, meaning that the variable x_i $(i = 1, \ldots, n)$ may not have a lower and/or an upper bound.

The Particle Swarm Paradigm (PSP)

The PSP is a population (swarm) based algorithm that mimics the social behavior of a set of individuals (particles).

An individual behavior is a combination of its past experience (cognition influence) and the society experience (social influence).

In the optimization context a particle p, at time instance t, is represented by its current position $(x^p(t))$, its best ever position $(y^p(t))$ and its travelling velocity $(v^p(t))$.

The new travel position and velocity

The new particle position is updated by

$$x^p(t+1) = x^p(t) + v^p(t+1),$$

where $v^p(t+1)$ is the new velocity given by

$$v_j^p(t+1) = \iota(t)v_j^p(t) + \mu\omega_{1j}(t)\left(y_j^p(t) - x_j^p(t)\right) + \nu\omega_{2j}(t)\left(\hat{y}_j(t) - x_j^p(t)\right),$$

for $j = 1, \ldots, n$, where $\iota(t)$ is a weighting factor (inertial), μ is the *cognition* parameter and ν is the *social* parameter. $\omega_{1j}(t)$ and $\omega_{2j}(t)$ are random numbers drawn from the uniform $(0, 1)$ distribution.

The best ever particle

$\hat{y}(t)$ is a particle position with global best function value so far, *i.e.*,

$$\hat{y}(t) = \arg\min_{a \in \mathcal{A}} f(a)$$
$$\mathcal{A} = \left\{ y^1(t), \ldots, y^s(t) \right\}.$$

In an algorithmic point of view we just have to keep track of the particle with the best ever function value.

The algorithm

1. Randomly initialize the swarm positions $\mathcal{S} = \{x^1(0), \ldots, x^s(0)\}$ and velocities $\mathcal{V} = \{v^1(0), \ldots, v^s(0)\}$.

2. Let $t = 0$ and $y^p(t) = x^p(t)$, $p = 1, \ldots, s$.

3. For all p in $\{1, \ldots, s\}$ do: If $f(x^p(t)) < f(y^p(t))$ then set $y^p(t+1) = x^p(t)$ else set $y^p(t+1) = y^p(t)$.

4. For all p in $\{1, \ldots, s\}$ do: Compute $v^p(t+1)$ and $x^p(t+1)$.

5. If the stopping criterion is true then stop. Otherwise set $t = t + 1$ go to step 3.

The (most used) stopping criterion

The stopping criterion used in the literature is related with the function value at the global optimum.

The algorithm stops if either the objective function at the best ever particle is approximately equal to the known objective minimum or a maximum number of iterations is exceeded.

We will use a different stopping criterion.

Features

1. Good
   (a) Easy to implement.
   (b) Easy to parallelize.
   (c) Easy to handle discrete variables.
   (d) Only uses objective function evaluations.

2. Not so good
   (a) Slow rate of convergence near an optimum.
   (b) High number of function evaluations.

Measuring infeasibility

To measure the infeasibility of a particle we propose the following function

$$\mathcal{H}(x) = e^{\sum_{i \in \mathcal{E}} \log(1+|c_i^{\mathcal{E}}(x)|) + \sum_{i \in \mathcal{I}} \log(1+[c_i^{\mathcal{I}}(x)]_+)}$$

where $[c]_+ = \max\{0, c\}$. The infeasibility function $\mathcal{H}(x) : R^n \to R$ does not account for the simple bound constraints as they are addressed by the projection

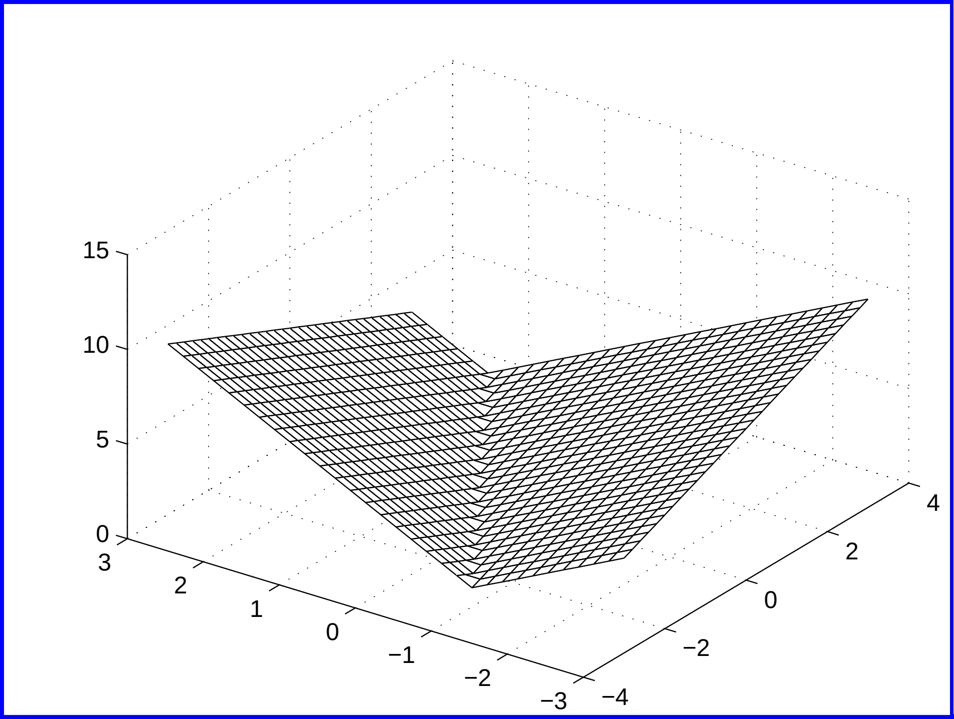
$$\mathcal{P}(x)_i = \left\{ \begin{array}{lll} x_i & \text{if} & l_i \leq x_i \leq u_i \\ l_i & \text{if} & x_i < l_i \\ u_i & \text{if} & x_i > u_i \end{array} \right. , i = 1, \ldots, n.$$

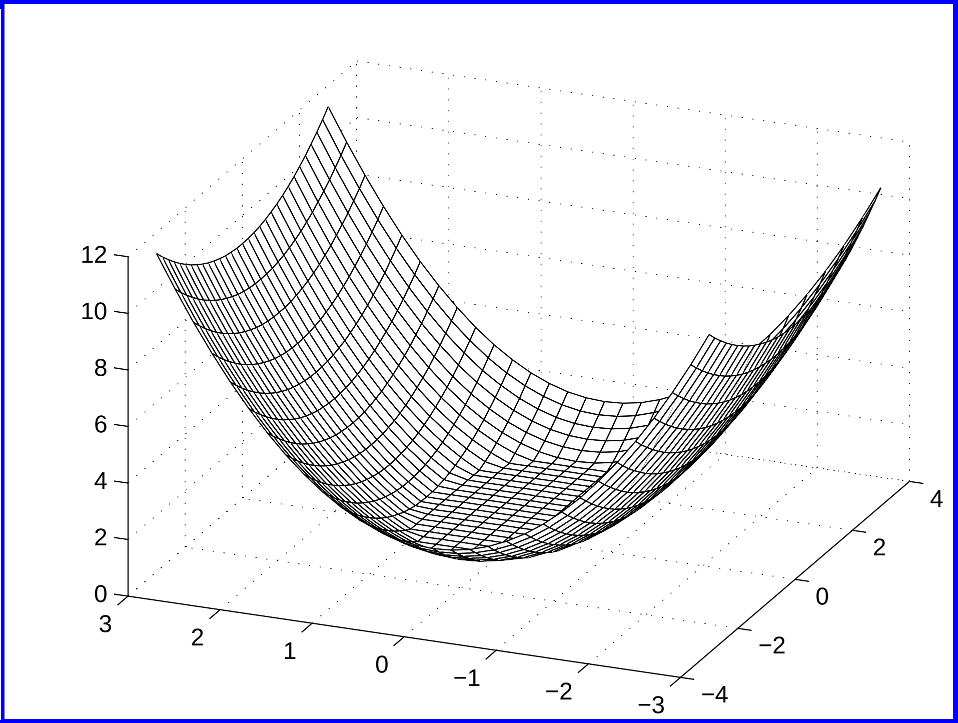
Example

As an example of the infeasibility measure consider the `hs014` problem from Hock and Schittkowski test suite

$$
\begin{aligned}
\min_{x \in R^2} f(x) &\equiv (x_1 - 2)^2 + (x_2 - 1)^2 \\
s.t. \quad c^{\mathcal{E}}(x) &\equiv x_1 - 2x_2 + 1 = 0 \\
c^{\mathcal{I}}(x) &\equiv \frac{x_1^2}{4} + x_2^2 - 1 \leq 0.
\end{aligned}
$$

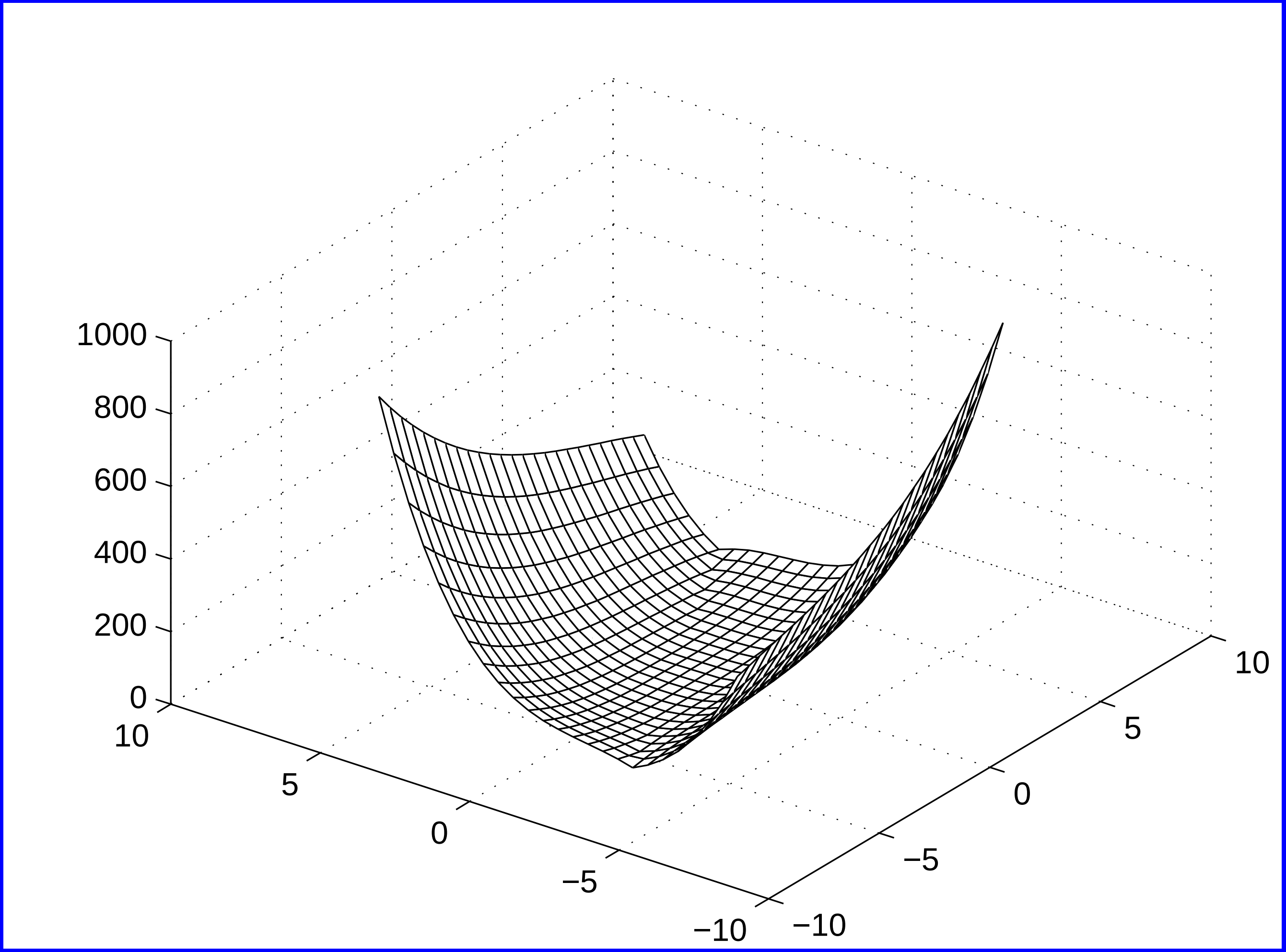
Example of $\mathcal{H}(x)$





Example

The combined plot for problem `hs014` is



Equivalent problem

For the infeasibility function $\mathcal{H}(x)$ we have

$$\mathcal{H}(x)\begin{cases} =1 & \text{if } x \text{ is feasible} \\ >1 & \text{if } x \text{ is infeasible.}\end{cases}$$

Thus, the previous problem can then be replaced by the equivalent problem

$$\begin{aligned} &\min_{x\in R^n} f(x) \\ &s.t. \quad \mathcal{H}(x)=1 \\ &\qquad l \leq x \leq u. \end{aligned}$$

The dominance concept in multiobjective optimization

Consider two objective functions $f^1(x)$ and $f^2(x)$ (the extension to more than two objective functions is straightforward)

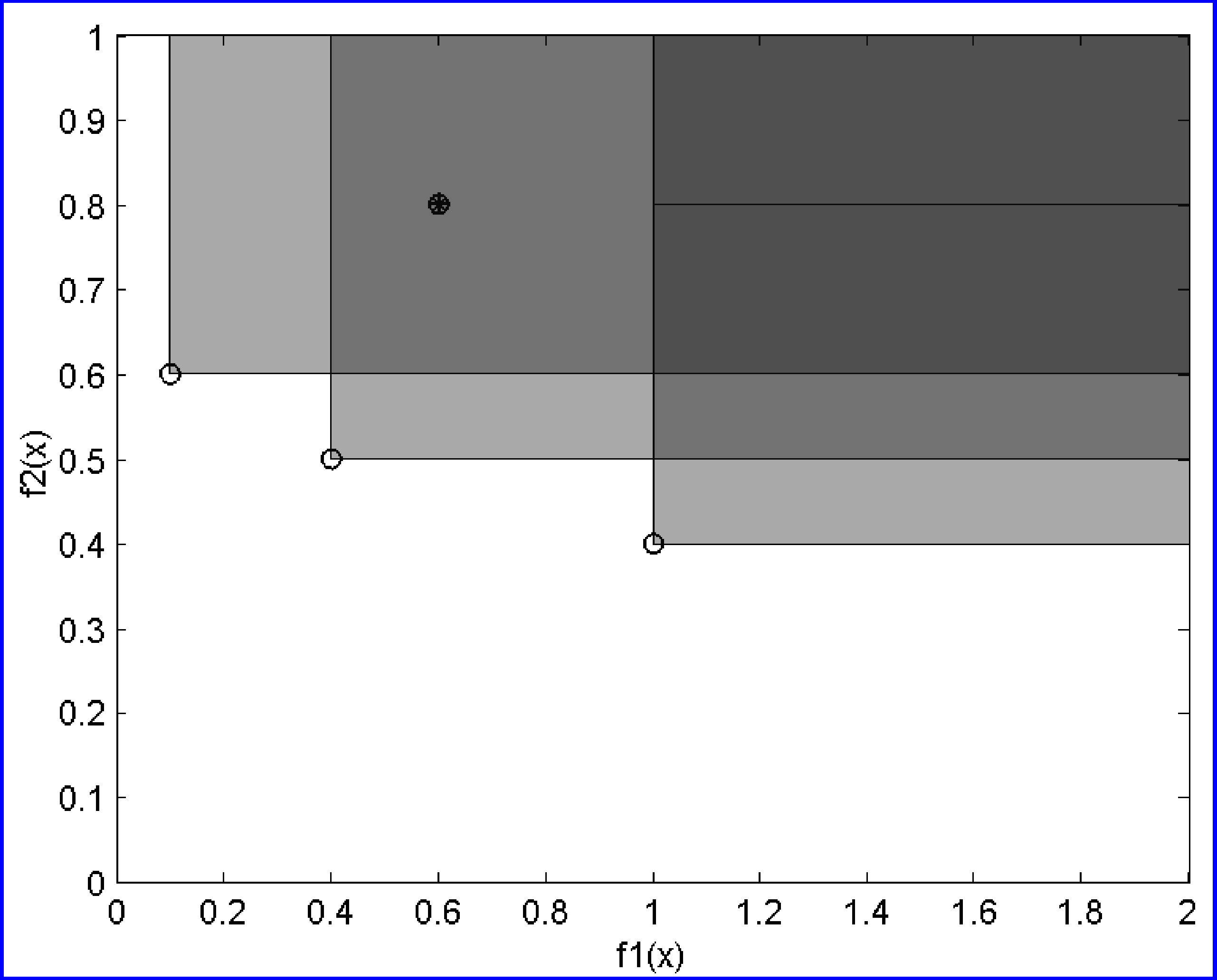
Let $\mathbf{f}(x) = (f^1(x), f^2(x))^T$.

$\mathbf{f}(x^i)$ is said to dominate $\mathbf{f}(x^j)$ if and only if $f^k(x^i) \leq f^k(x^j)$, $k = 1, 2$, and $f^k(x^i) < f^k(x^j)$ for at least one $k \in \{1, 2\}$.

A point x^i is said to be Pareto optimal if and only if there is no other point x^j such that $\mathbf{f}(x^j)$ dominates $\mathbf{f}(x^i)$.

Let $\mathcal{P}^*$ denote the set of all Pareto optimal solutions. Then the set $\{\mathbf{f}(x) : x \in \mathcal{P}^*\}$ is called the Pareto front.

Graphical interpretation



Equivalent problem

Finding a global solution to the latter problem is somehow equivalent to find a point $\bar{x}$ for which $(f(\bar{x}), \mathcal{H}(\bar{x}))$ dominates $(f(x), \mathcal{H}(x))$, for all $x \neq \bar{x}$ and $\mathcal{H}(\bar{x}) = 1$.

Priority is given in minimizing $\mathcal{H}(x)$ over minimizing $f(x)$.

For a point $\bar{x}$ we consider that progress was attained if either

$$\mathcal{H}(x) > \mathcal{H}(\bar{x})$$

or

$$((\mathcal{H}(x) \leq \mathcal{H}(\bar{x}) \quad \text{or} \quad \mathcal{H}(\bar{x}) \leq 1 + \epsilon) \quad \text{and} \quad f(x) > f(\bar{x}))\,,$$

The proposed algorithm

1. Randomly initialize the swarm $\mathcal{S} = \{x^1(0), \ldots, x^s(0)\}$ and swarm velocities $\mathcal{V} = \{v^1(0), \ldots, v^s(0)\}$.

2. Let $t = 0$ and $y^p(t) = x^p(t)$, $p = 1, \ldots, s$.

3. For all p in $\{1, \ldots, s\}$ do: If $\mathcal{H}(y^p(t)) > \mathcal{H}(x^p(t))$ or $((\mathcal{H}(y^p(t)) \leq \mathcal{H}(x^p(t))$ or $\mathcal{H}(x^p(t)) \leq 1 + \epsilon)$ and $f(y^p(t)) > f(x^p(t)))$ then set $y^p(t+1) = x^p(t)$ else set $y^p(t+1) = y^p(t)$.

4. For all p in $\{1, \ldots, s\}$ do: Compute $v^p(t+1)$ and $x^p(t+1)$.

5. If the stopping criterion is true then stop. Otherwise set $t = t + 1$ go to step 3.

The initial population

The initial swarm positions are randomly generated. If both simple bounds are finite ($l_i \neq -\infty$ and $u_i \neq \infty$) the i coordinate of particle p position is randomly generated by an uniform distribution on the interval $[l_i, u_i]$.

If one of the simple bounds is not finite then the algorithm requires an initial guess, $\hat{x}$. The coordinate i of particle p position is then randomly generated by using

$$x_i^p \sim \begin{cases} U\left(-\frac{\|\hat{x}\|\hat{x}_i}{2}, \frac{\|\hat{x}\|\hat{x}_i}{2}\right) & \text{if } l_i = -\infty \text{ and } u_i = \infty \\ U\left(2\hat{x}_i - u_i, u_i\right) & \text{if } l_i = -\infty \text{ and } u_i \neq \infty \\ U\left(l_i, 2\hat{x}_i - l_i\right) & \text{if } l_i \neq -\infty \text{ and } u_i = \infty. \end{cases} \tag{1}$$

The initial guess, when provided by the user, is included in the initial swarm.

The stopping criterion

A typical stopping criterion is to stop the algorithm when the search direction is approximately zero, meaning that progress is no longer possible.

This idea can also be used in the particle swarm context. Since in this case, we are dealing with a population of points, a possible extension for this criterion is to stop when all the search directions are approximately zero, *i.e.*,

$$\max_{p \in \mathcal{S}} \|v^p(t+1)\| \leq \epsilon$$

where $\epsilon > 0$ is a small tolerance.

When the stopping criterion is met, but the best ever particle position is infeasible ($\mathcal{H}(\hat{y}) > 1$) the swarm is reinitialized. The new swarm includes the best ever found particle and the other particles are randomly initialized following the previously described procedure.

Connection to AMPL

AMPL (`www.ampl.com`) is a mathematical programming language that allows the codification of optimization problems in a powerful and easy to learn language.

AMPL also provides an interface that allows a wide variety of solvers to communicate with it.

The implemented algorithm is available in the NLCPSOA (*NonLinear Constrained Particle Swarm Optimization Algorithm*) solver. The solver contains an interface to connect to AMPL allowing the user to write and solve a problem coded in AMPL in an extremely easy way.

NLCPSOA returns to AMPL the best ever found solution.

Test examples

We have used four engineering design problems and six additional problems

		Number of	
	Variables	Constraints	
Problem		Inequality	Equality
Vessel	4	4	0
Beam	4	7	0
Spring	3	4	0
Himmelblau's	5	3 (bound)	0
pso1	2	1	1
pso2	2	2	0
pso3	7	4	0
pso4	5	3 (bound)	0
pso5	5	3 (bound)	0
pso6	6	2	0

Numerical results

Vessel design problem		
	Reported	Obtained
x_1	0.8125	0.778169
x_2	0.4375	0.384649
x_3	42.09845	40.3196
x_4	176.6366	200
c_1	0.0	-9.94553E-09
c_2	-0.03588	-3.80778E-09
c_3	-5.8208E-11	-5.84856E-04
c_4	-63.3634	-40
f	6059.131296	5885.33

Beam design problem		
	Reported	Obtained
x_1	0.20573	0.201381
x_2	3.47049	3.23192
x_3	9.03662	10
x_4	0.20573	0.201381
c_1	0.0	-0.0292784
c_2	0.0	-4972.77
c_3	-5.5511151E-17	-3.58641e-07
c_4	-3.432983785	-3.32625
c_5	-0.0807296	-0.0763803
c_6	-0.2355403	-0.239099
c_7	-9.094947E-13	-0.00391764
f	1.72485084	1.81429

Numerical results - Cont.

Spring design problem		
	Reported	Obtained
$x_1(d)$	0.051466369	0.05
$x_2(D)$	0.351383949	0.310414
$x_3(N)$	11.60865920	15
c_1	-0.003336613	-3.30997e-06
c_2	-1.0970128E-4	-0.0173742
c_3	-4.0263180998	-186.267
c_4	-0.7312393333	-0.759724
f	0.0126661409	0.0131926

Himmelblau's design problem		
	Reported	Obtained
x_1	78.0	78
x_2	33.0	33
x_3	27.070997	27.1106
x_4	45.0	45
x_5	44.96924255	45
c_1	92.0	92
c_2	100.4047843	98.8726
c_3	20.0	20.0196
f	-31025.56142	-31012.1

Numerical results - Cont.

Problem	Obtained solution	Reported best solution
pso1	1.39347	1.39343
pso2	-6961.81	-6961.837
pso3	680.63	680.639
pso4	-30665.5	-31544.459
pso5	-31026.4	-31545.054
pso6	-213	-213.0

Numerical results - Cont.

Problem	Objective value	Function/Constraints evaluations
vessel	5885.33	879000
beam	1.81429	960300
spring	0.0131926	757800
himmelblau	-31012	784200
pso1	1.39347	1417200
pso2	-6961.81	1461900
pso3	680.63	1689600
pso4	-30665.5	975300
pso5	-31026.4	792900
pso6	-213	879900

Conclusions

- A new particle swarm optimization algorithm for nonlinear constrained optimization, which aims to minimize infeasibility and the objective, is described;
- It uses the dominance concept to handle constraints (easy implementation);
- Good numerical results with a set of problems;
- NLCPSOA connected to AMPL.
- This approach allows a bigger swarm to be used (in compare with previous approaches).

The End

email:	aivaz@dps.uminho.pt
	emgpf@dps.uminho.pt
Web	http://www.norg.uminho.pt/aivaz/
	http://www.norg.uminho.pt/emgpf/

First Page