Cosine Function applied to the Inertia Control in the Particle Swarm Optimization

Tiago Silveira [1]
Humberto C. B. de Oliveira [1,2]
Ricardo Menezes Salgado [1]
Luiz Eduardo da Silva [1]
Geraldo Robson Mateus [2]

1 Federal University of Alfenas (UNIFAL) - Brazil
2 Federal University of Minas Gerais (UFMG) - Brazil

Abstract

- An adaptation of metaheuristic called Particle Swarm Optimization;

- The paper presents a new mechanism to reduce statistically the chances of the optimization process stagnating in local minima using PSO;

Particle Swarm Optimization

PSO

- The PSO is Biologically inspired;
- The PSO was based on 2 main areas:
  - Artificial life;
  - Evolutionary computation.

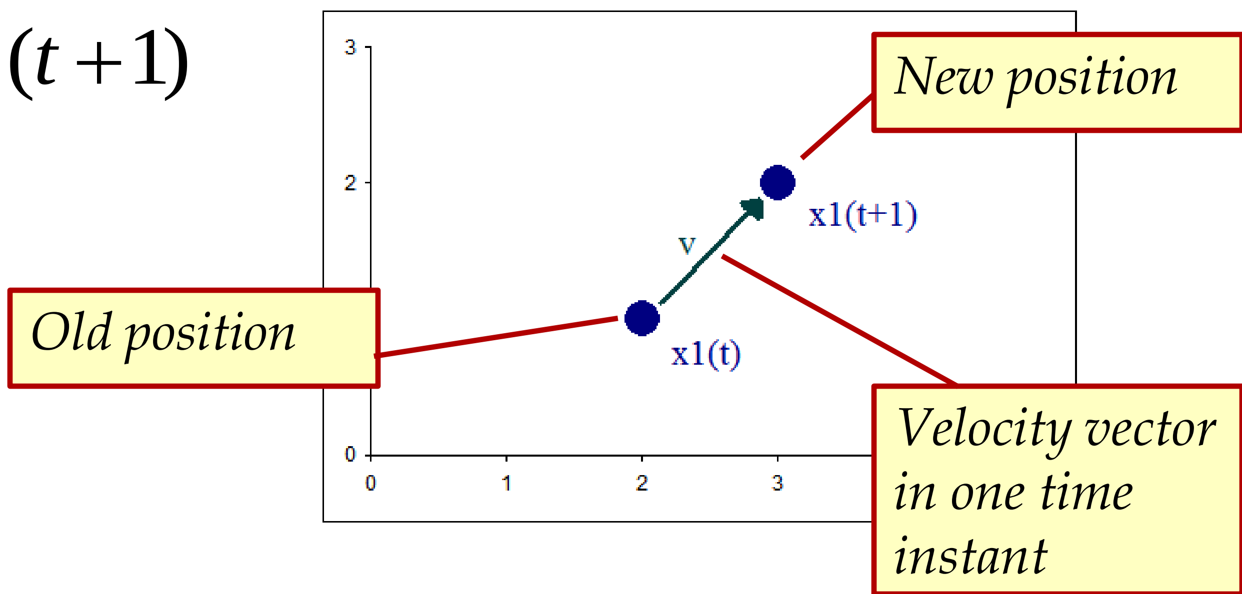
Biologically inspired computing

Particle Swarm Optimization

- The authors of PSO propose a mathematical model to define the particle velocity in a time instant:

$$v_i(t+1) = wv_i(t) + c_1 r_1 (p_i^*(t) - x_i(t)) + c_2 r_2 (p_b^*(t) - x_i(t))$$

$$x_i(t+1) = x_i(t) + v_i(t+1)$$



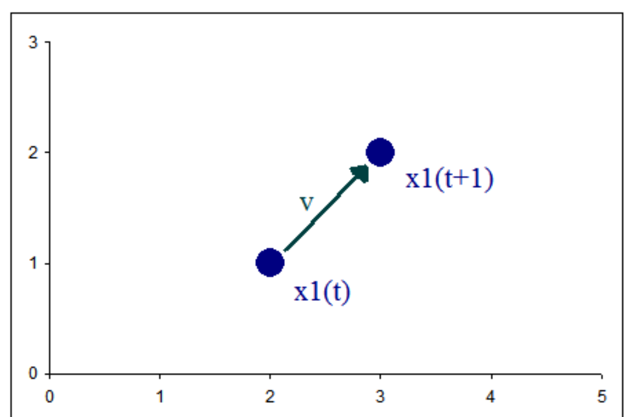
Particle Swarm Optimization

- The authors of PSO propose a mathematical function to define the particle velocity in a time instant

$$\boxed{v_i(t+1)} = wv_i(t) + c_1 r_1 (p_i^*(t) - x_i(t)) + c_2 r_2 (p_b^*(t) - x_i(t))$$

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

$v_i(t+1)$: **velocity in this time**



Particle Swarm Optimization

- The authors of PSO propose a mathematical function to define the particle velocity in a time instant

$$v_i(t+1) = w\boxed{v_i(t)} + c_1 r_1(p_i^*(t) - x_i(t)) + c_2 r_2(p_b^*(t) - x_i(t))$$

$v_i(t)$: **velocity vector in last time**

Particle Swarm Optimization

- The authors of PSO propose a mathematical function to define the particle velocity in a time instant:

$$v_i(t+1) = wv_i(t) + c_1 r_1 (\boxed{p_i^*(t)} - x_i(t)) + c_2 r_2 (p_b^*(t) - x_i(t))$$

$p_i^*(t)$: **best particle position**

Particle Swarm Optimization

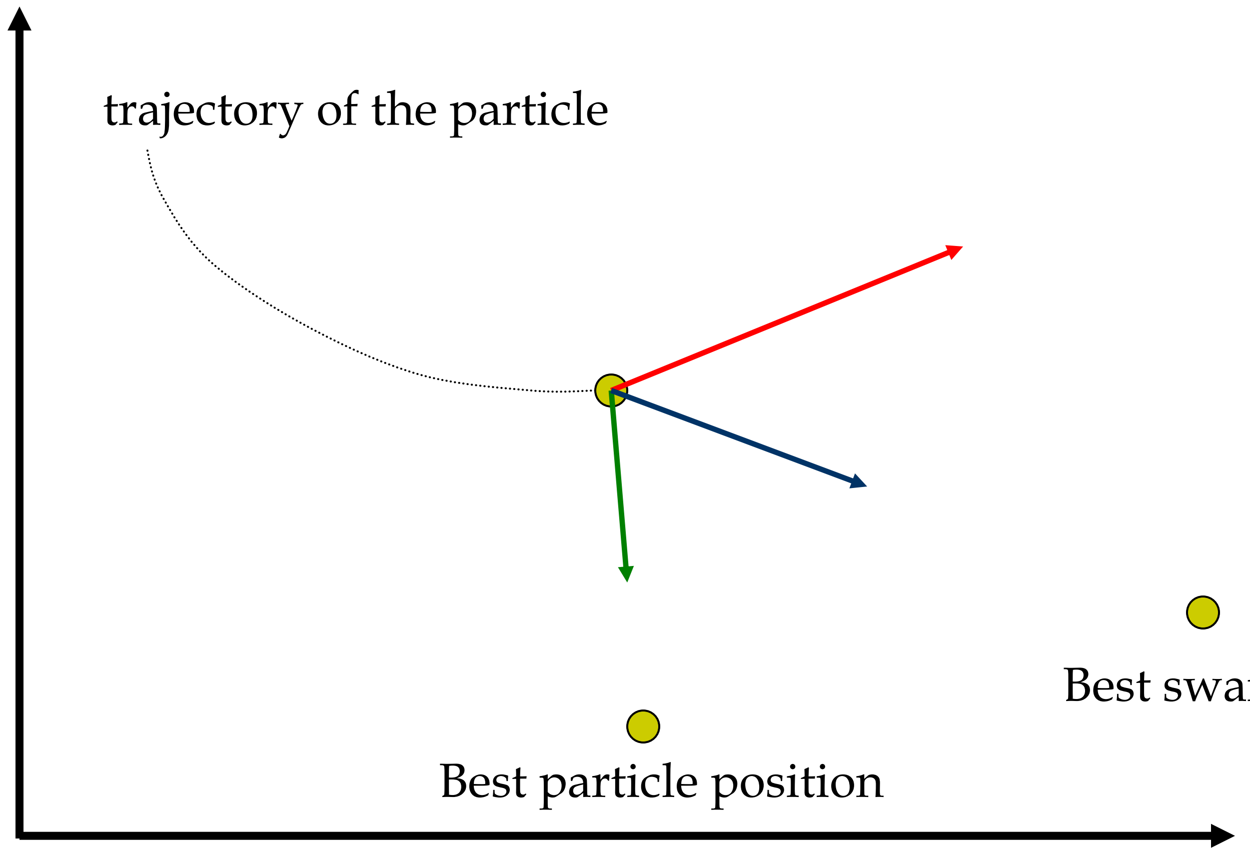
- The authors of PSO propose a mathematical function to define the particle velocity in a time instant:

$$v_i(t+1) = wv_i(t) + c_1 r_1 (p_i^*(t) - x_i(t)) + c_2 r_2 (\boxed{p_b^*(t)} - x_i(t))$$

$p_i^*(t)$: **best swarm position**

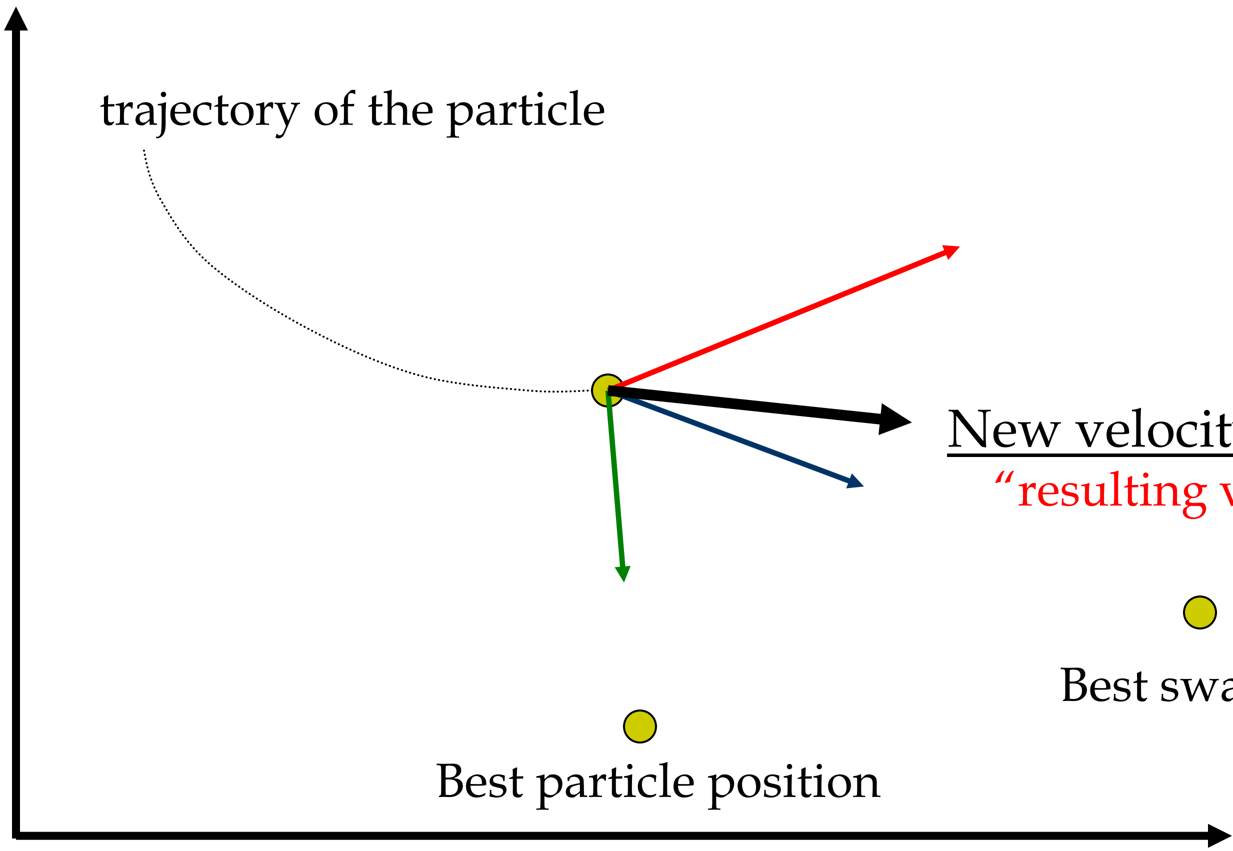
Particle Swarm Optimization

$$v_i(t+1) = \boxed{wv_i(t)} + \boxed{c_1 r_1 (p_i^*(t) - x_i(t))} + \boxed{c_2 r_2 (p_b^*(t) - x_i(t))}$$



Particle Swarm Optimization

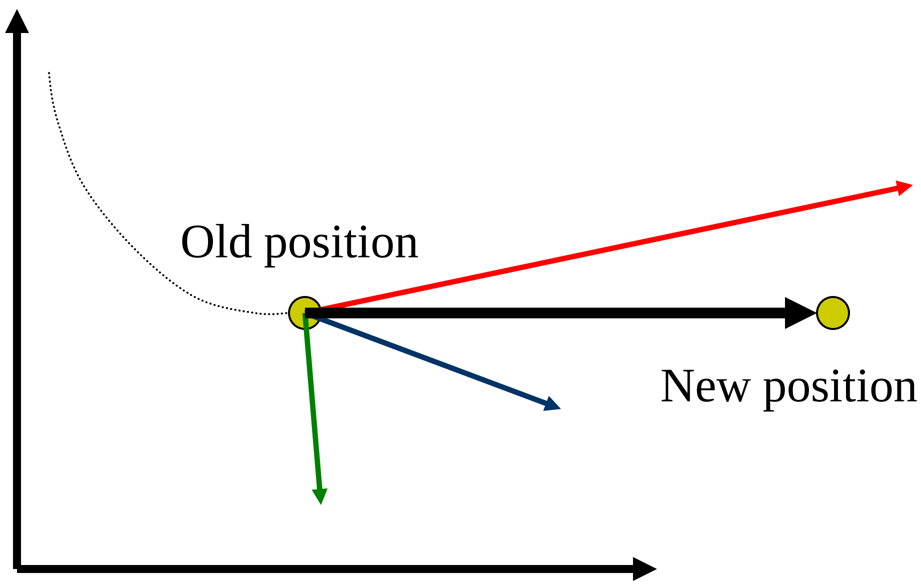
$$\boxed{v_i(t+1)} = wv_i(t) + c_1 r_1 (p_i^*(t) - x_i(t)) + c_2 r_2 (p_b^*(t) - x_i(t))$$

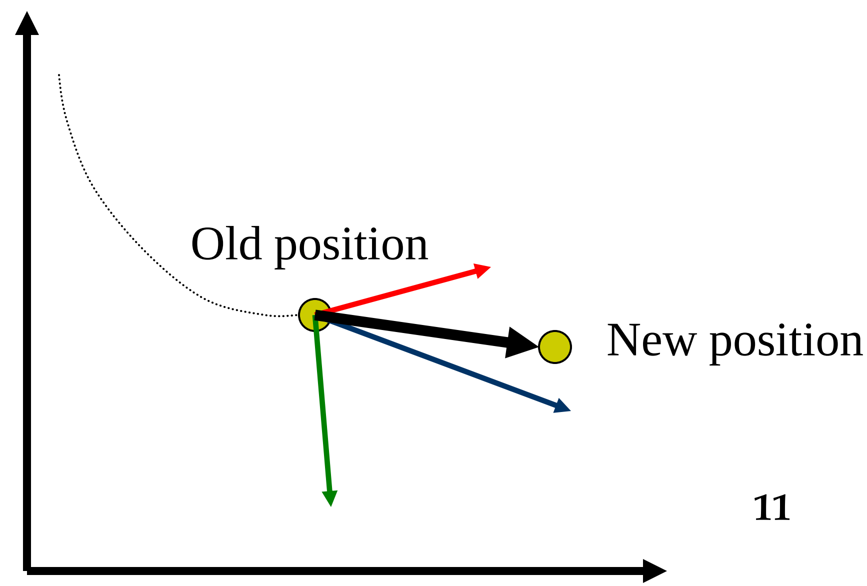


Particle Swarm Optimization

- The w parameter controls the swarm energy

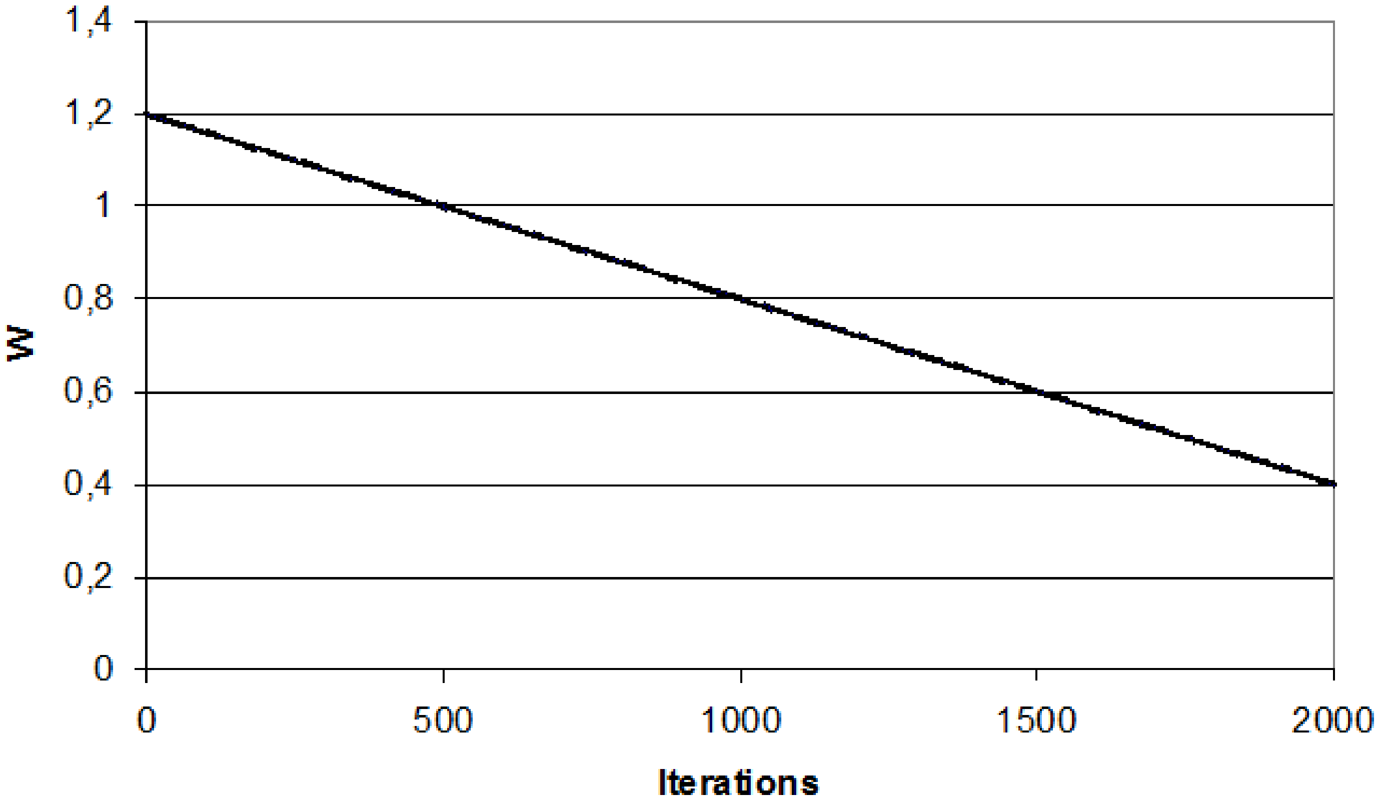
$$v_i(t+1) = \boxed{w} v_i(t) + c_1 r_1 (p_i^*(t) - x_i(t)) + c_2 r_2 (p_b^*(t) - x_i(t))$$



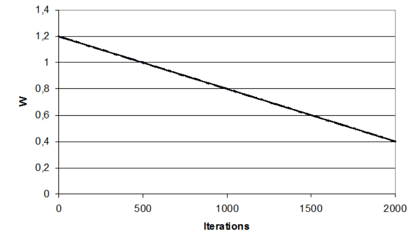


Particle Swarm Optimization

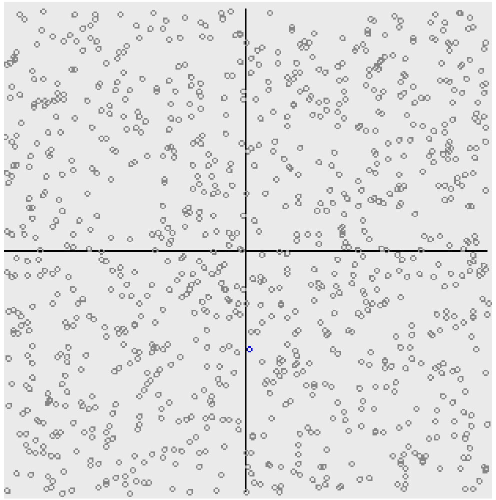
- In classical PSO, the *w* parameter has the following behavior:



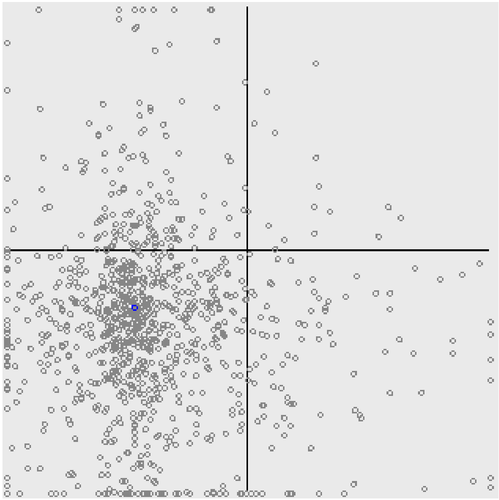
the swarm loses energy over time ...



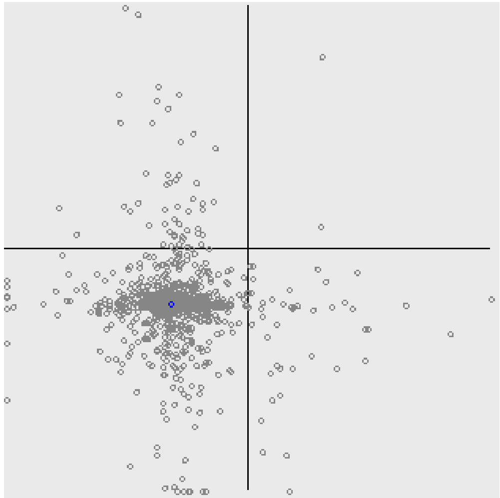
t_0



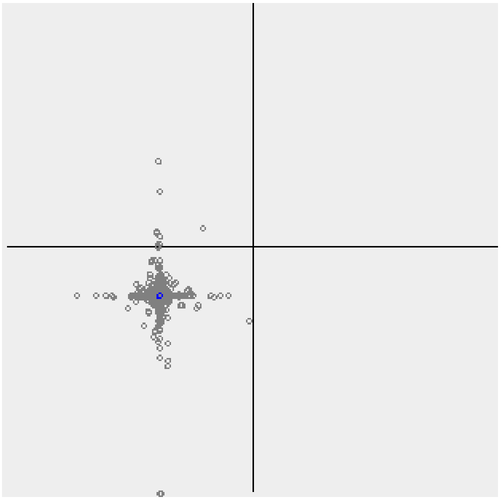
t_1



t_2

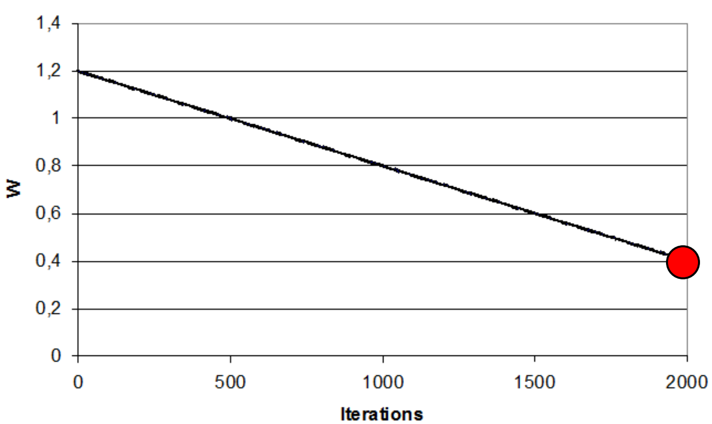


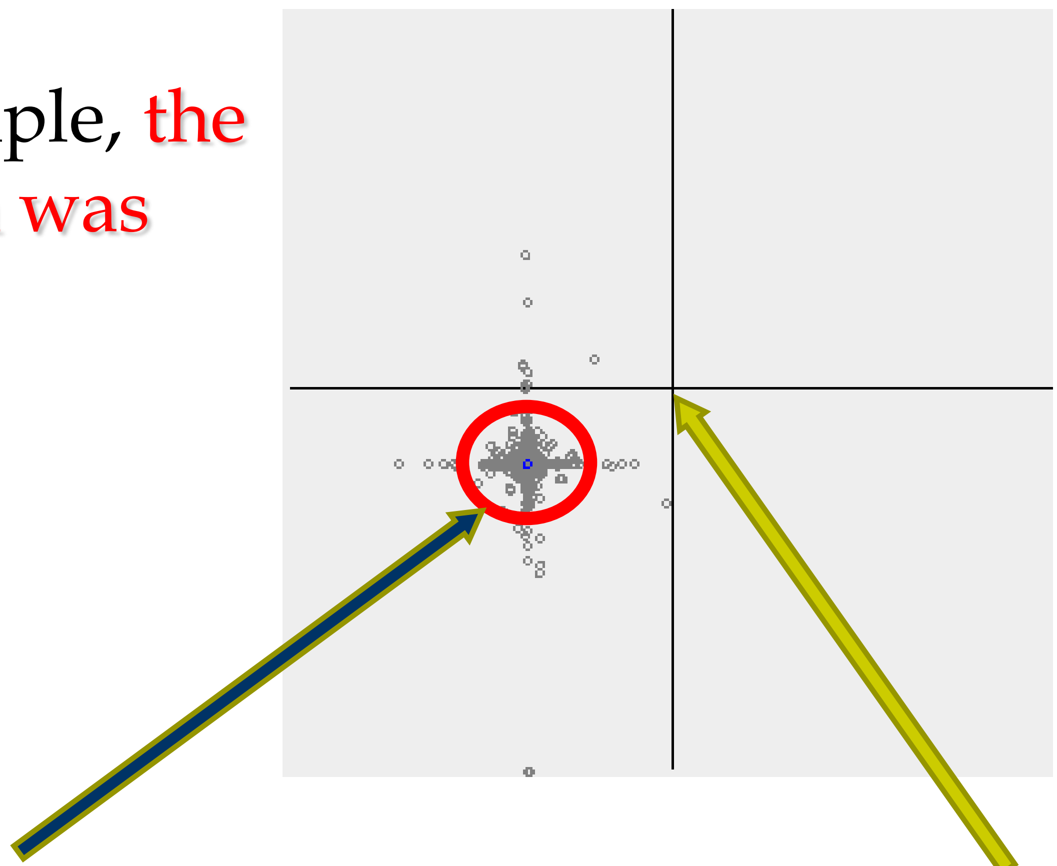
t_3



the swarm loses energy over time ...

- In this real example, the optimal solution was never found.





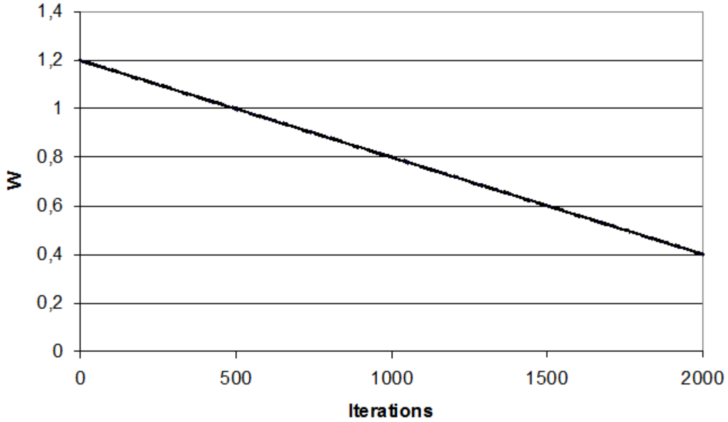
space of local search

best solution:
point (0,0)

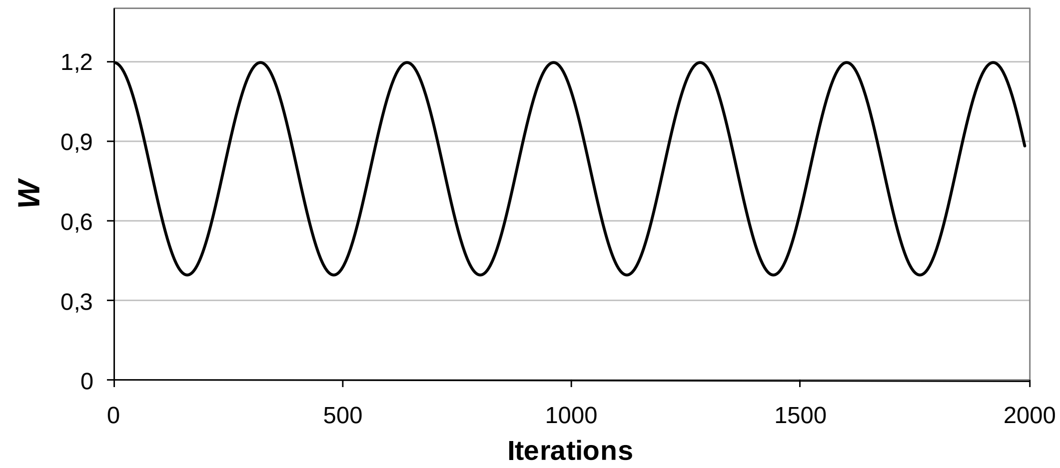
Proposal

- Faced with the problem, a mechanism is proposed to avoid stagnation in local minima;
- The central idea is the energy gain of the particles;
- For this, we use a non-monotonic control of w parameter.

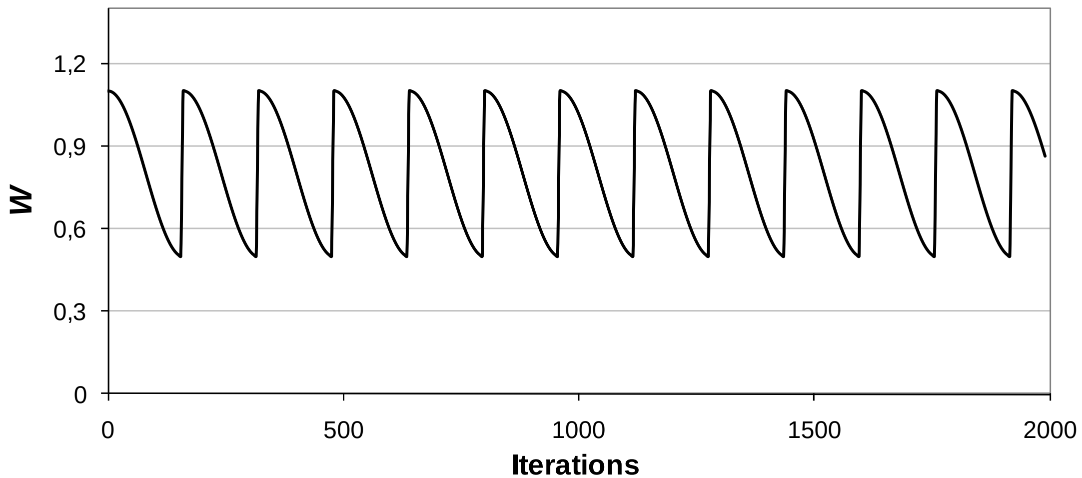
- Classical control



- Inertia control with Cosine Function



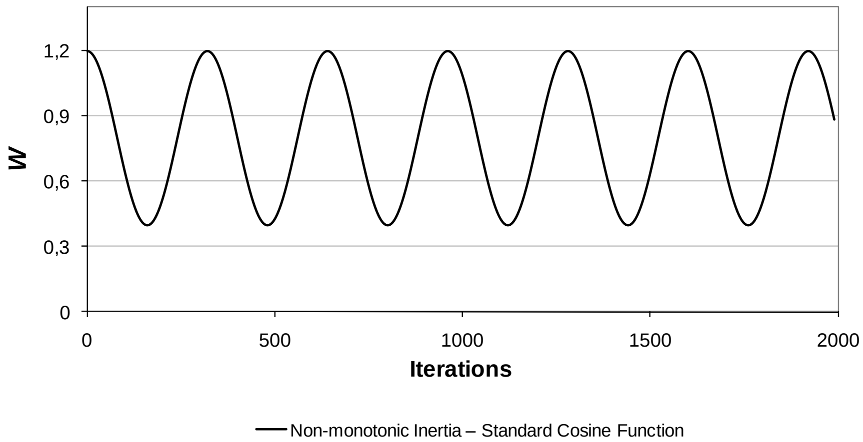
- Mirrored Cosine

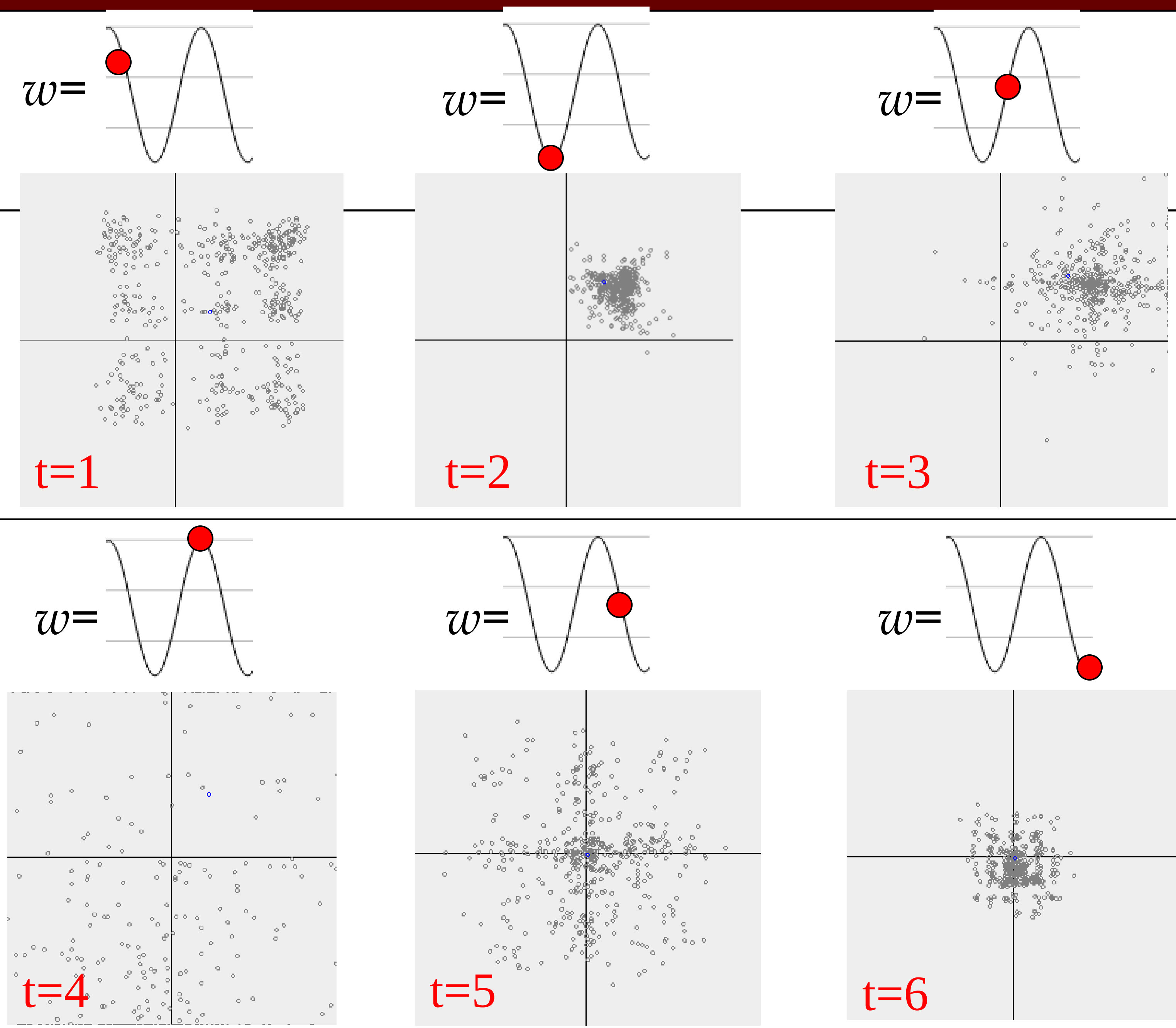


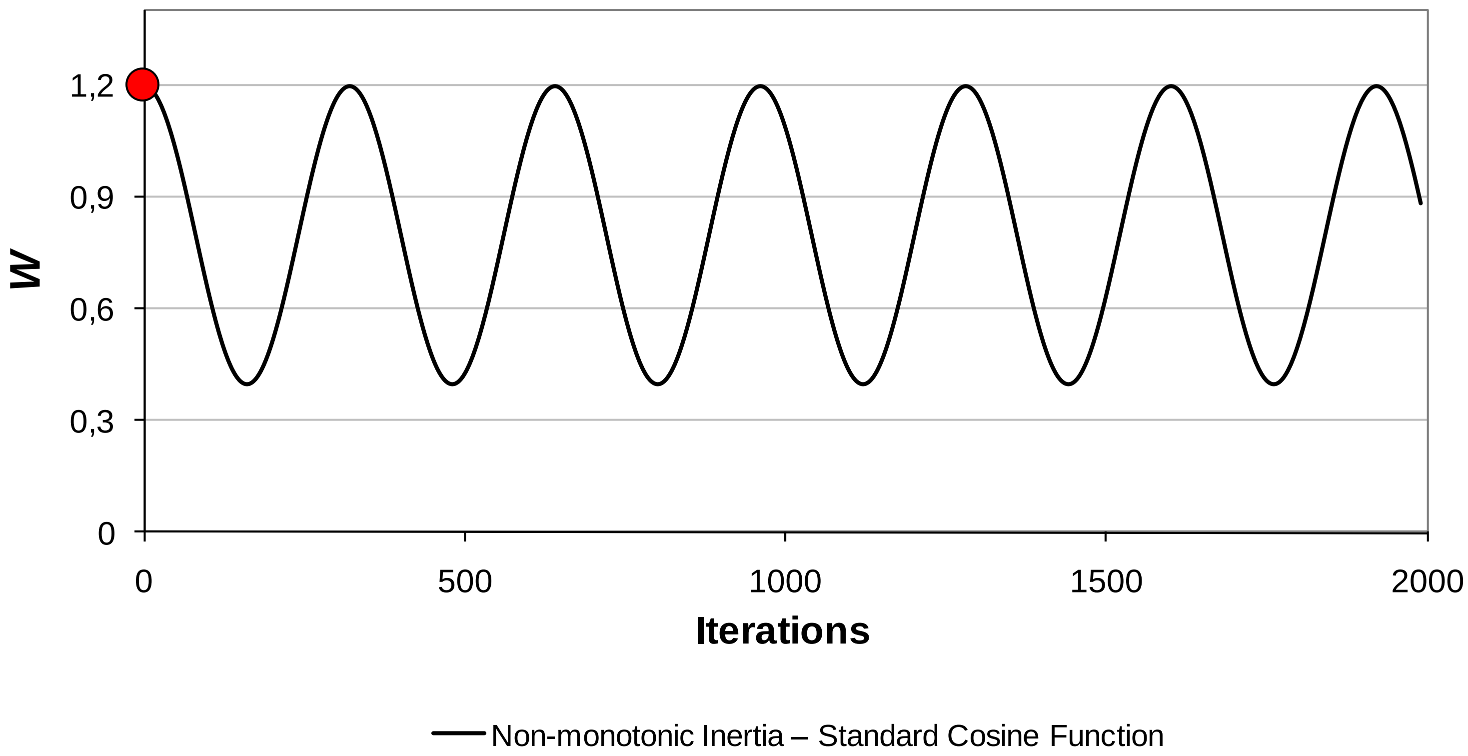
Non-monotonic control

$$v_i(t+1) = wv_i(t) + c_1 r_1 (p_i^*(t) - x_i(t)) + c_2 r_2 (p_b^*(t) - x_i(t))$$

- Advantage:
  - When w parameter receives values greater than 1, then the particle accelerates.
  - Over the iterations, the swarm energy is accumulated.




w=
t=1
w=
t=2
w=
t=3
w=
t=4
w=
t=5
w=
t=6



This process is repeated several times in our proposal

Experiments

Functions for tests

- 1st: *Sphere*
  Unimodal

$$f_1(x) = \sum_{i=1}^{n} x_i^2$$

- 2nd: *De Jong F4 – no noise*
  Unimodal

$$f_2(x) = \sum_{i=1}^{n} i \,.\, x_i^4$$

- 3nd: *Rastrigin*
  Multimodal

$$f_3(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$$

- 4nd: *Griewank*
  Multimodal

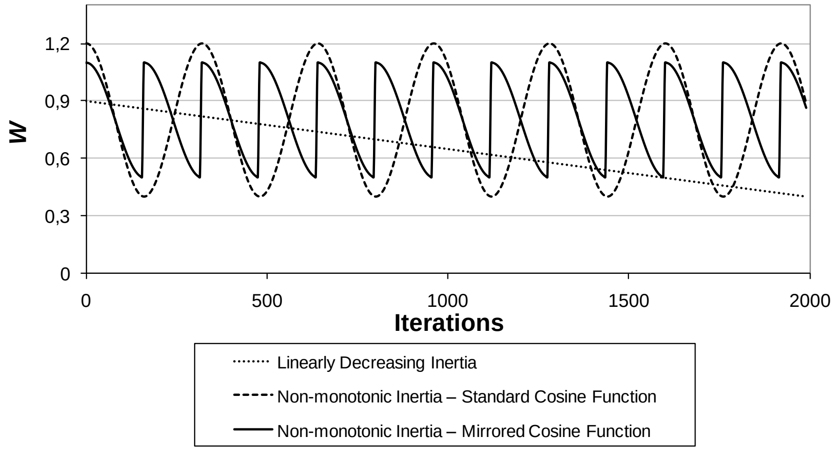
$$f_4(x) = \sum_{i=1}^{n} (x_i^2 / 4000) - \prod_{i=1}^{n} \cos(x_i / \sqrt{i}) + 1$$

- 5nd: *Ackley*
  Multimodal

$$f_5(x) = 20 + e - 20.\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi . x_i)\right)$$

Experimental Setup

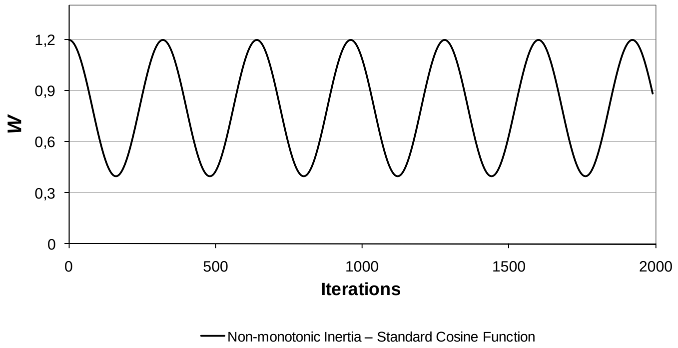
- Models tested:
  - Classical PSO;
  - Standard Cosine Function;
  - Mirrored Cosine Function



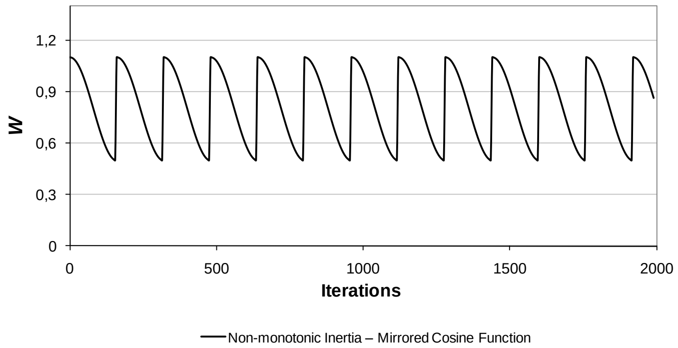
- Parameters for each model:
  - iterations = 2000;
  - swarm with 1000 particles;
  - c1 = c2 = 1.4;

Experimental Setup

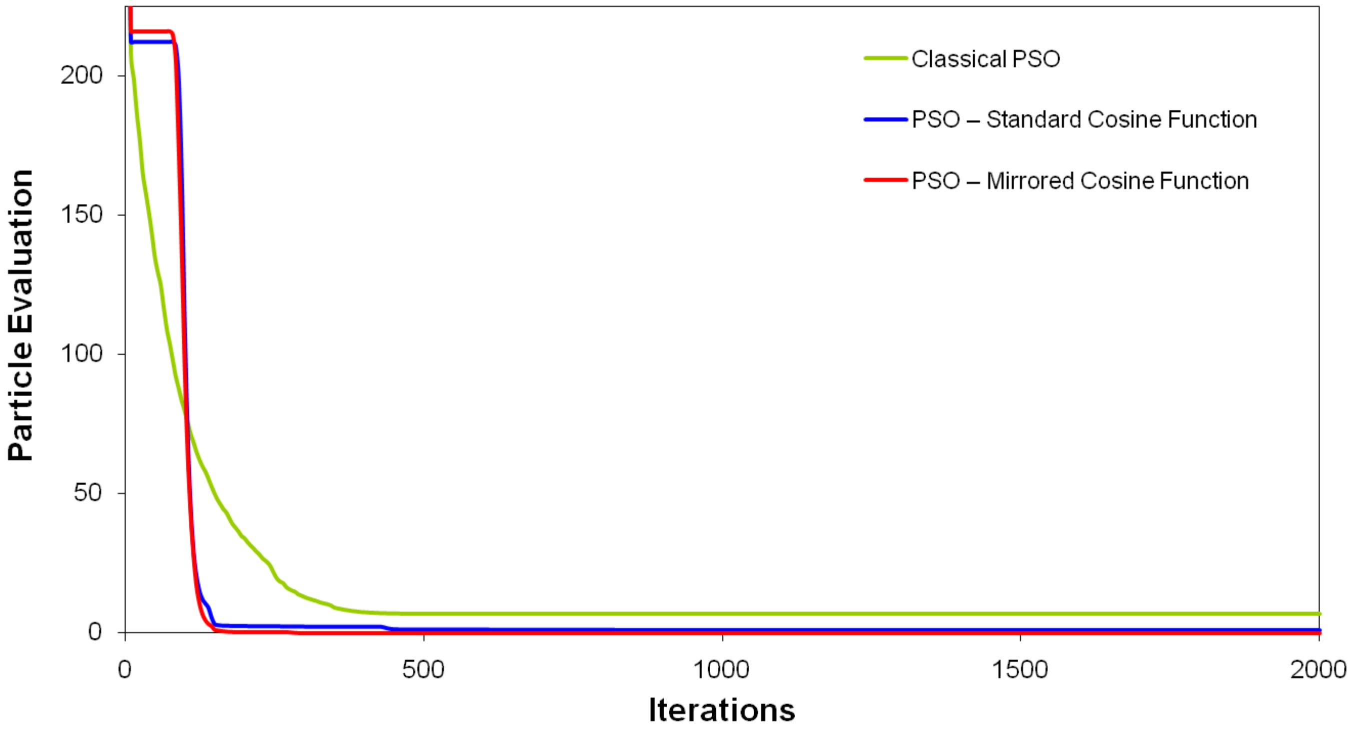
- For standard cosine function:
  - *w* max = 1.2;
  - *w* min = 0.4;
  - non-monotonic cycles are completed after *320* iterations;

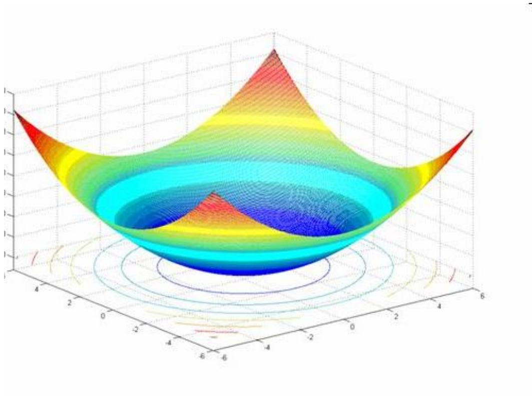


- For mirrored cosine function:
  - *w* max = 1.1;
  - *w* min = 0.5;
  - non-monotonic cycles are completed after *160* iterations;



Results



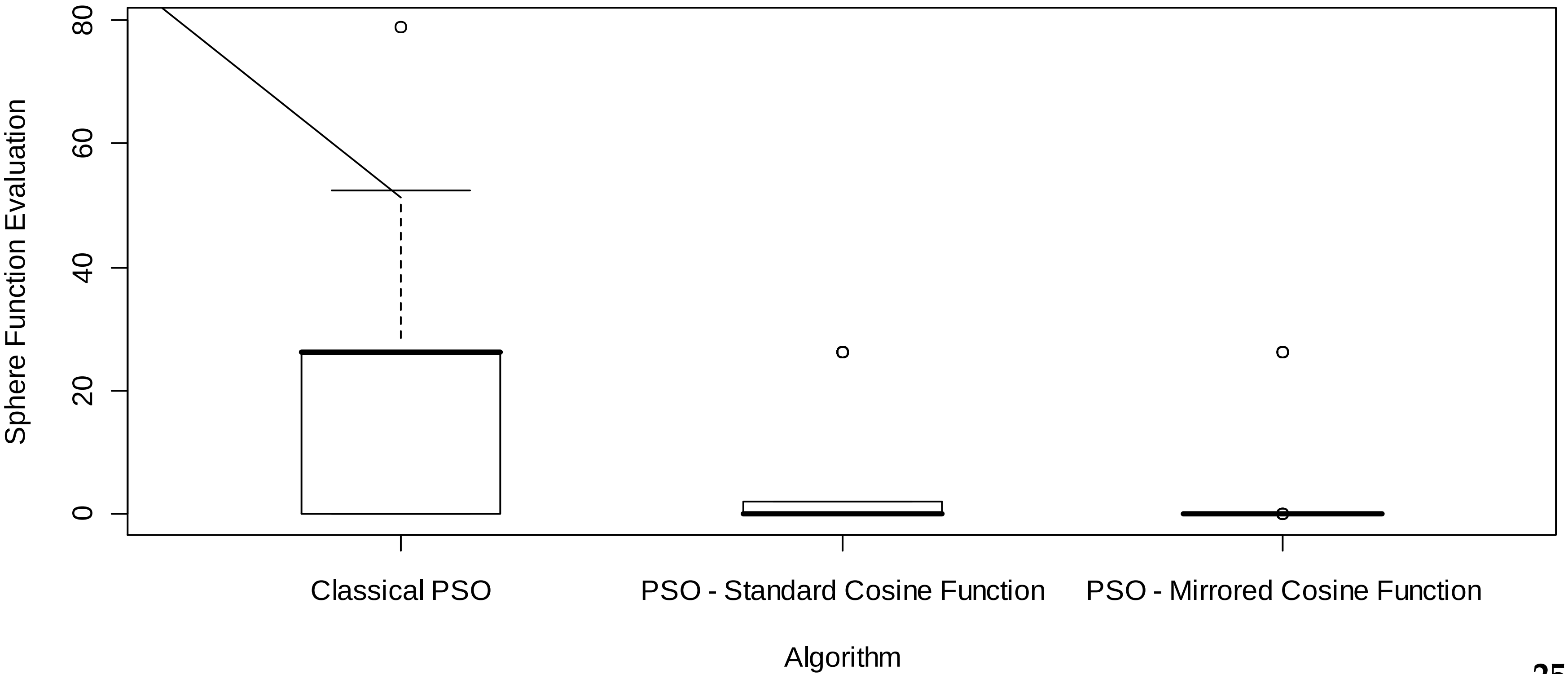


$$f_1(x) = \sum_{i=1}^{n} x_i^2$$

- *Sphere* function with dimension 60
  - Unimodal function

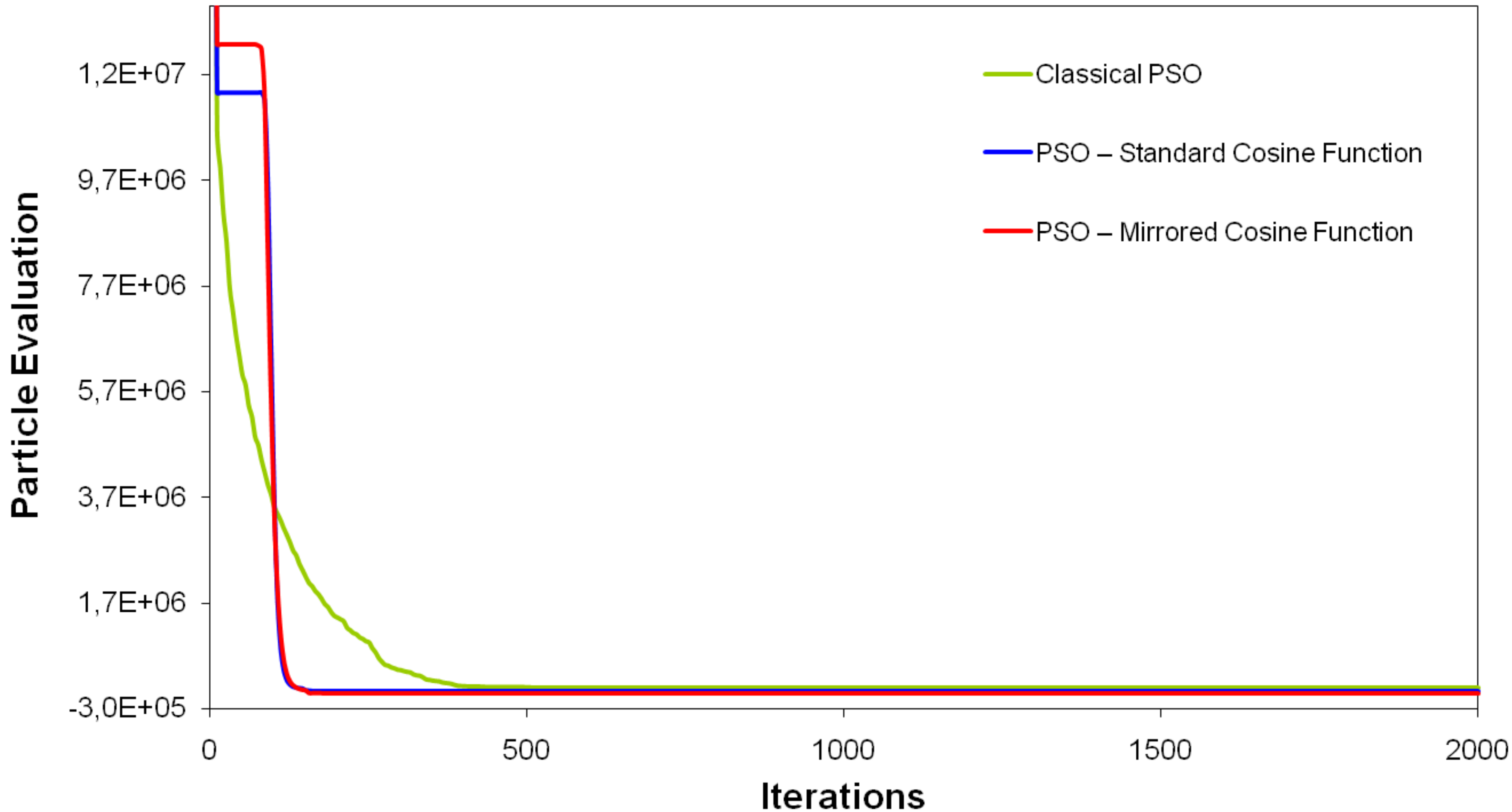
Results (*boxplot*)

- Sphere Function (unimodal)



Results

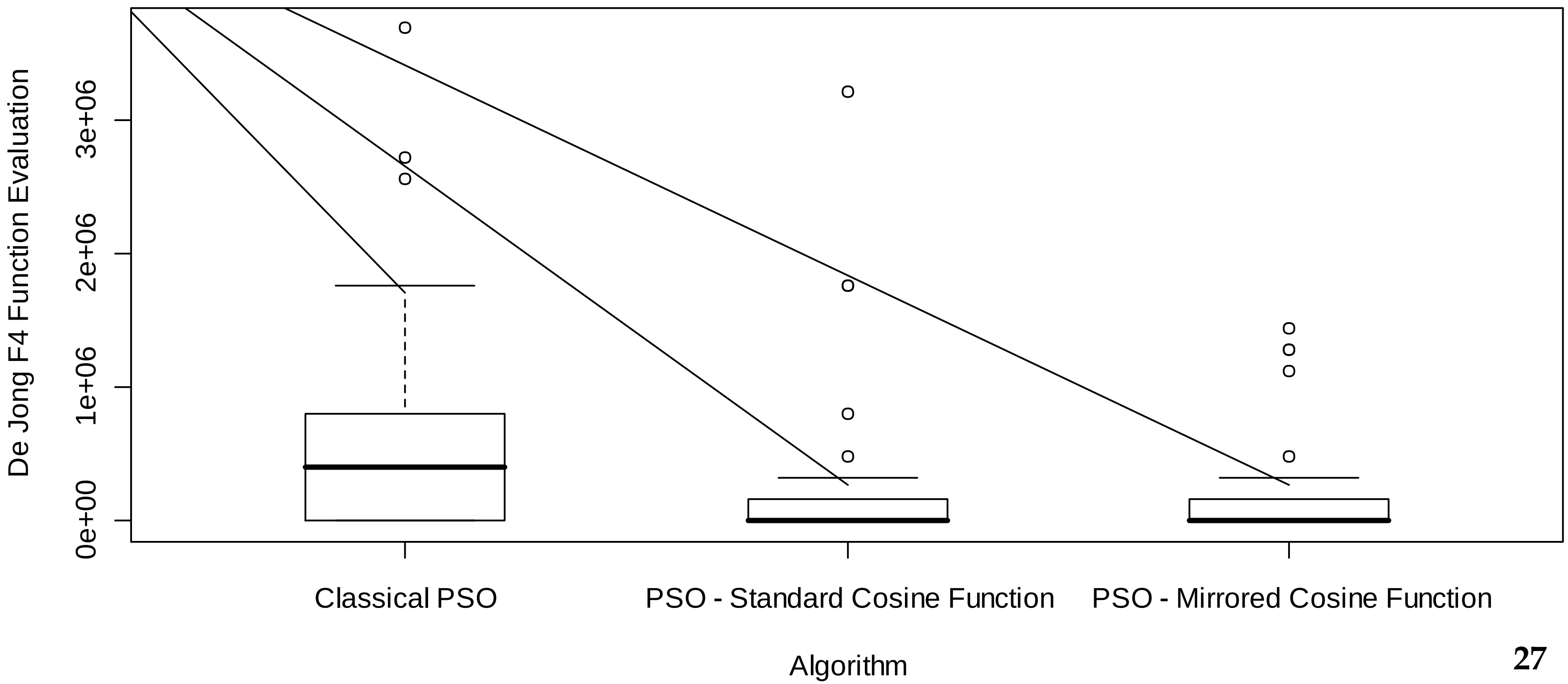
$$f_2(x) = \sum_{i=1}^{n} i \, . \, x_i^4$$



- *De Jong F4 function (no noise)* with dimension 60
  - Unimodal function

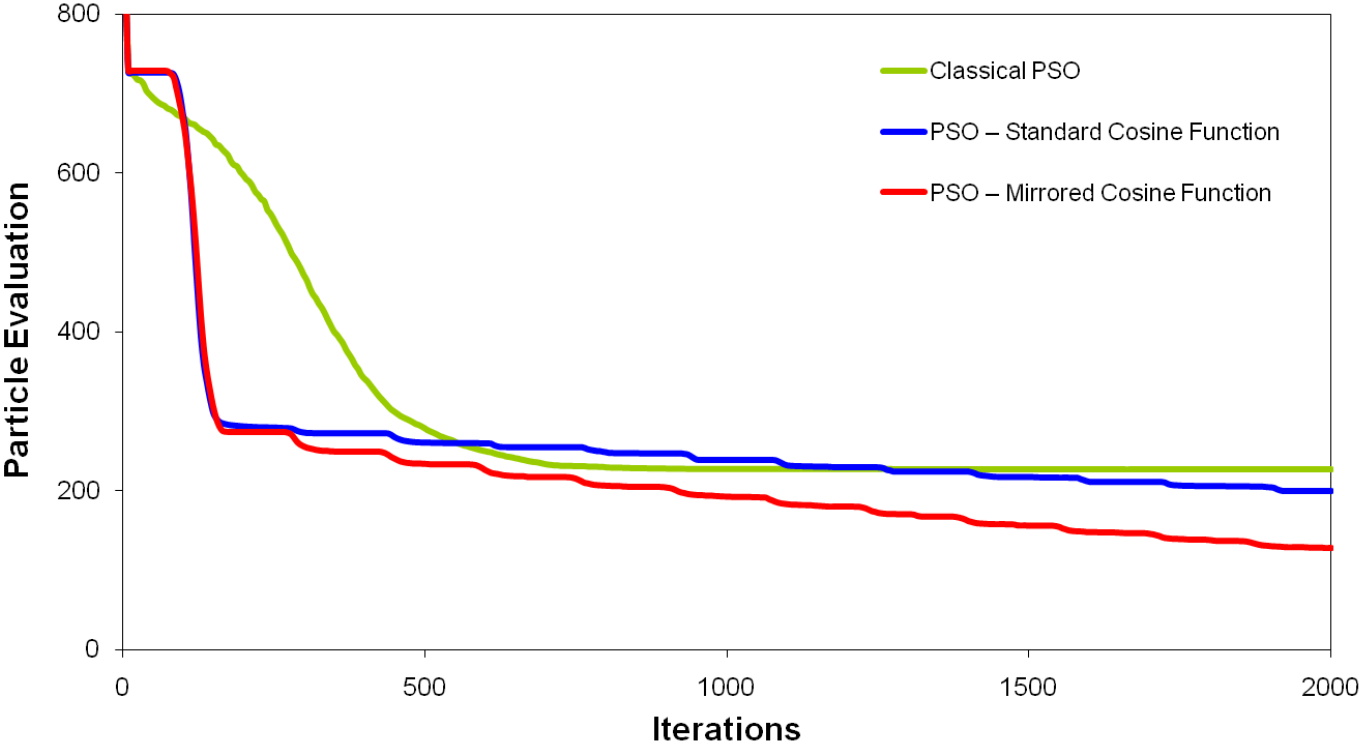
Results (*boxplot*)

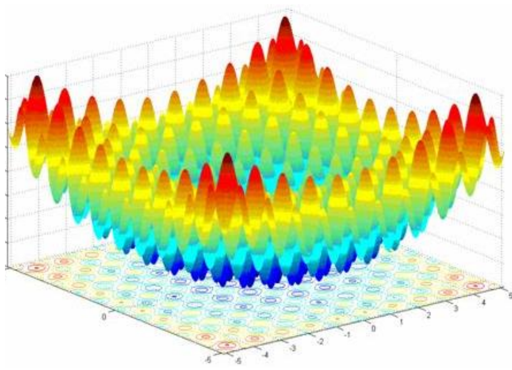
- De Jong F4 (unimodal)



Results

$$f_3(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$$

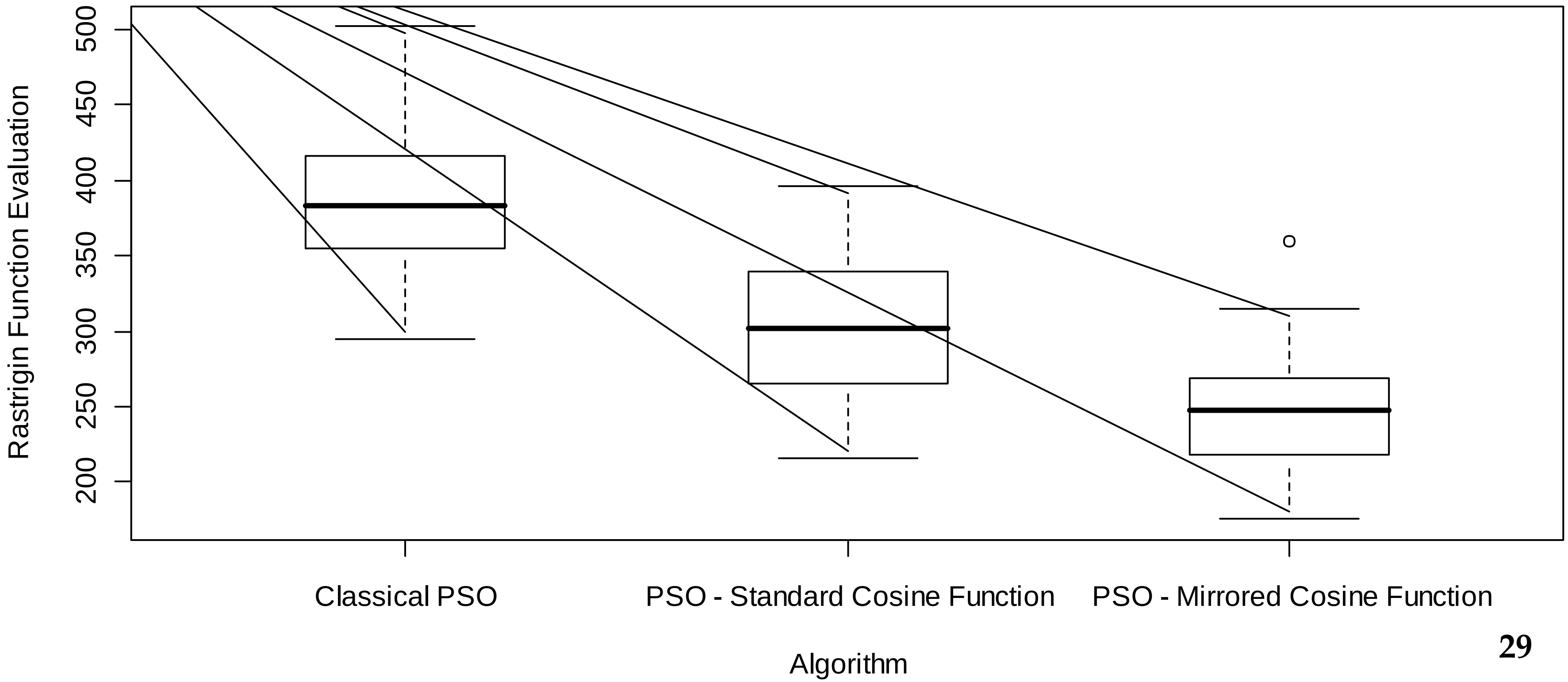




- *Rastrigin function* with dimension 60
  - Multimodal function

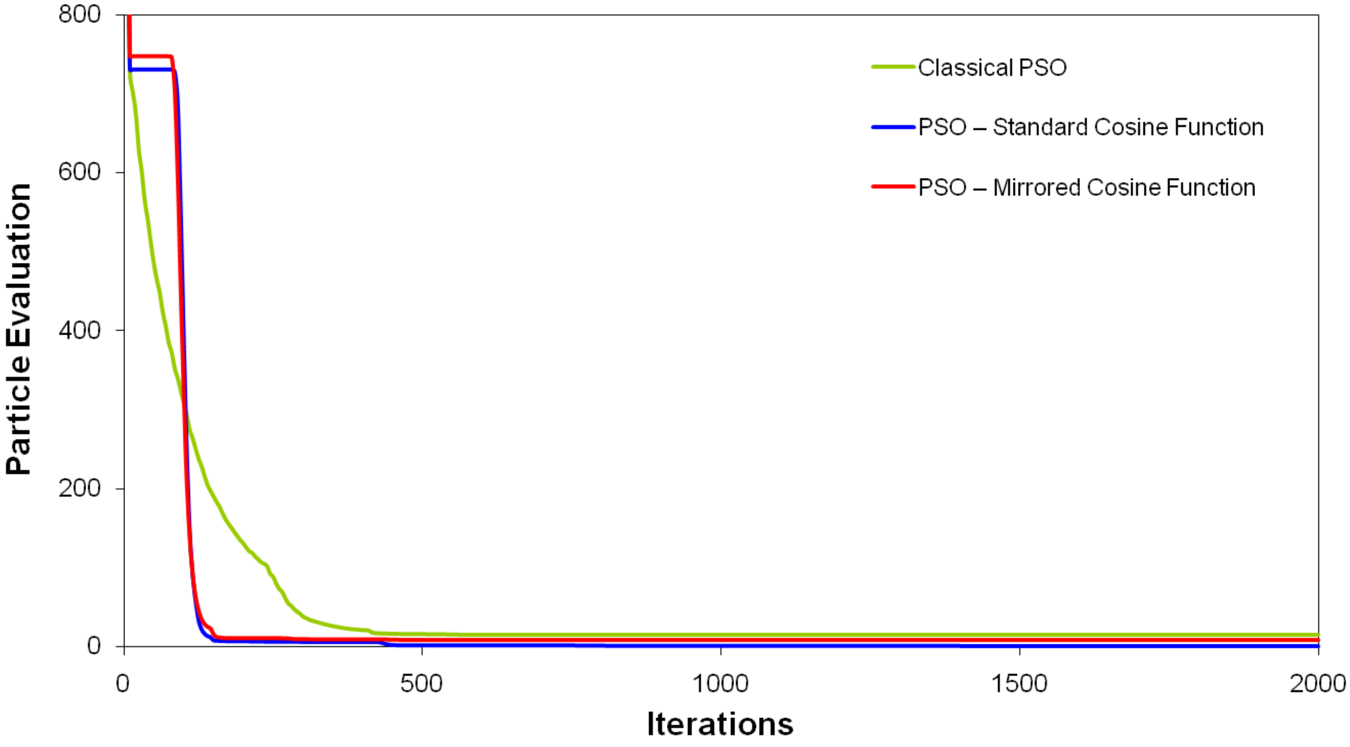
Results (*boxplot*)

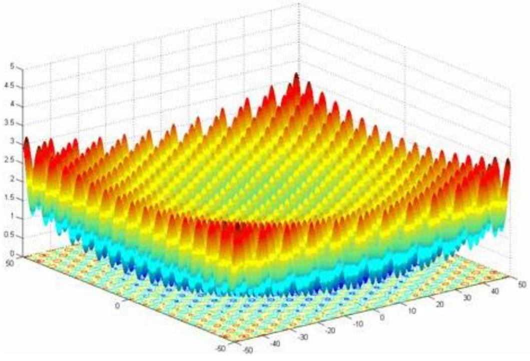
- *Rastringin* (multimodal)



Results

$$f_4(x) = \sum_{i=1}^{n}(x_i^2/4000) - \prod_{i=1}^{n}\cos(x_i/\sqrt{i}) + 1$$

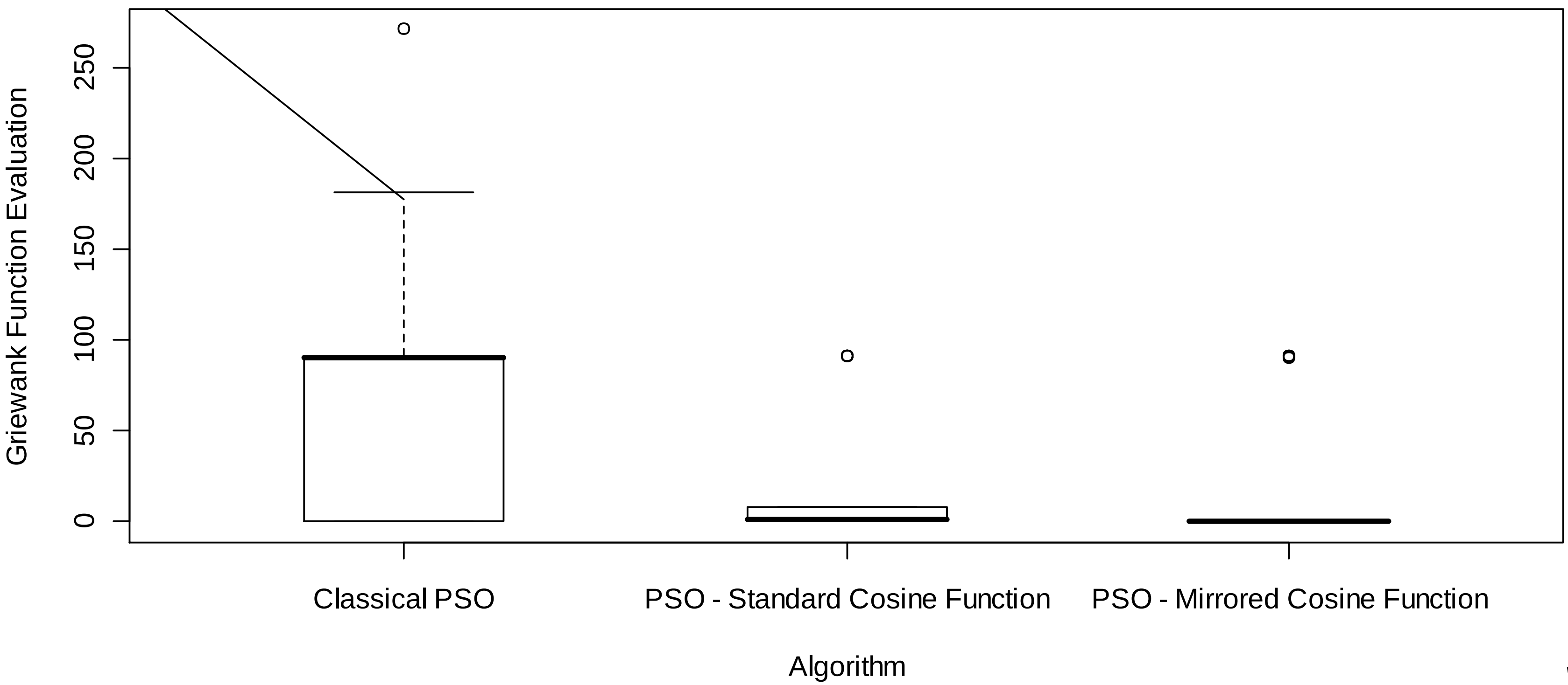




- *Griewank function* with dimension 60
  - Multimodal function

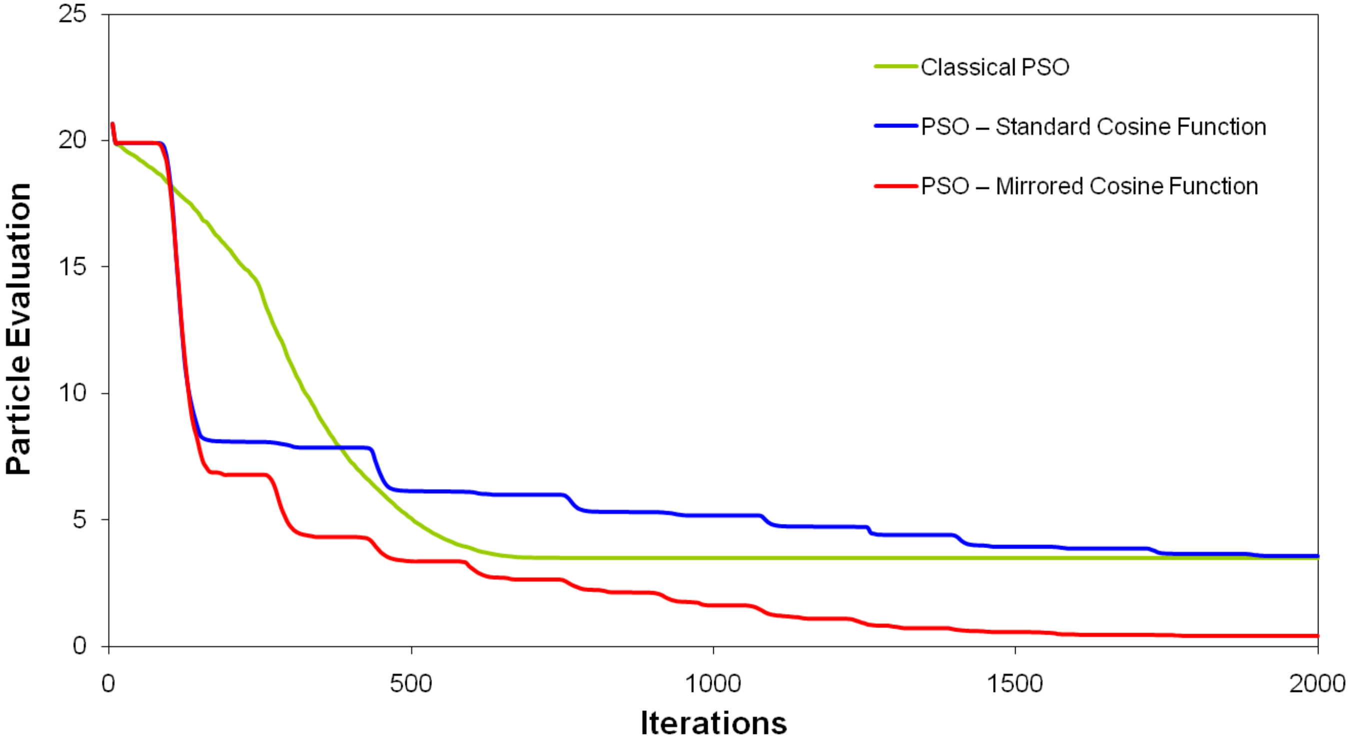
Results (*boxplot*)

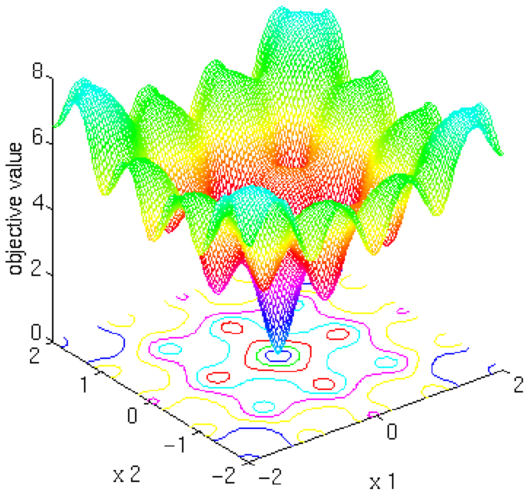
- *Griewank* (multimodal)



Results

$$f_5(x) = 20 + e - 20.\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi.x_i)\right)$$

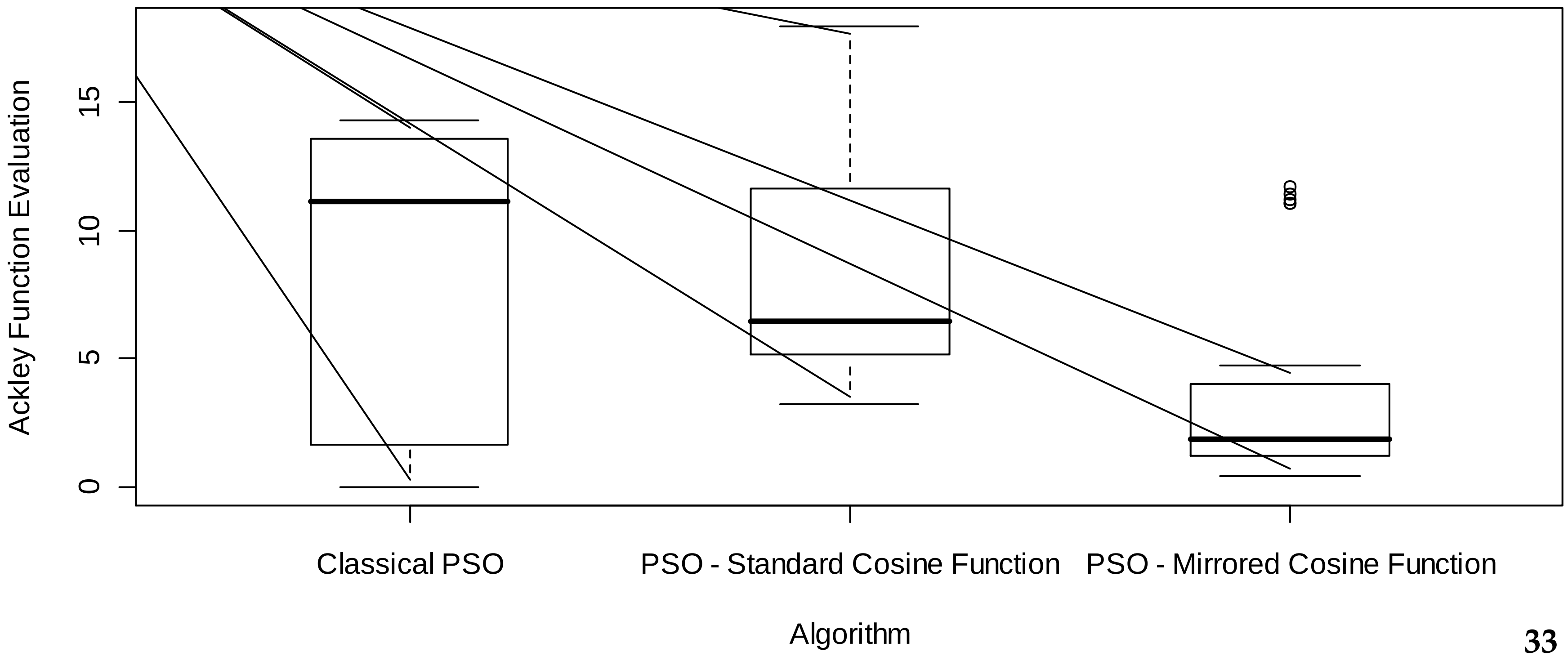




- *Ackley function* with dimension 60
  - Multimodal function

Results (*boxplot*)

- *Ackley* (multimodal)



Conclusions (1)

- A new way to control the inertia of the particle in the PSO;
- This new approach assigns non-monotonic values to w parameter;
- Thus, the swarm gains energy and spreads itself. After, with the loss of energy, the particles stay close again.

Conclusions (2)

- The new approaches were able to optimize the well known benchmark functions;

- The PSO with *w* parameter using cosine control is more adapted for multimodal functions, if compared with Classical PSO.

Cosine Function applied to the Inertia Control in the Particle Swarm Optimization

Tiago Silveira [1]
Humberto C. B. de Oliveira [1,2]
Ricardo Menezes Salgado [1]
Luiz Eduardo da Silva [1]
Geraldo Robson Mateus [2]

[1] *Federal University of Alfenas (UNIFAL) - Brazil*
[2] *Federal University of Minas Gerais (UFMG) - Brazil*