Recent Advances in Particle Swarm Optimization Analysis and Understanding

AP Engelbrecht[1,2] and CW Cleghorn [3]

[1]Department of Industrial Engineering
[2]Division of Computer Science
Stellenbosch University
South Africa
engel@sun.ac.za

[3]Department of Computer Science
University of Pretoria
South Africa
ccleghorn@cs.up.ac.za

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07.
doi:10.1145/3319619.3323368

Presenter
Andries Engelbrecht

Received the Masters and PhD degrees in Computer Science from the University of Stellenbosch, South Africa, in 1994 and 1999 respectively. He is Voigt Chair in Data Science in the Department of Industrial Engineering, with a joint appointment as Professor in the Computer Science Division, Stellenbosch University. His research interests include swarm intelligence, evolutionary computation, artificial neural networks, artificial immune systems, and the application of these Computational Intelligence paradigms to data analytics, games, bioinformatics, finance, and difficult optimization problems. He is author of two books, *Computational Intelligence: An Introduction* and *Fundamentals of Computational Swarm Intelligence*.

Presenter
Christopher Cleghorn

Received his Masters and PhD degrees in Computer Science from the University of Pretoria, South Africa, in 2013 and 2017 respectively. He is senior lecturer in Computer Science at the University of Pretoria, and a member of the Computational Intelligence Research Group. His research interests include swarm intelligence, evolutionary computation, and machine learning, with a strong focus on theoretical research. Dr Cleghorn annually serves as a reviewer for numerous international journals and conferences in domains ranging from swarm intelligence and neural networks to mathematical optimization.

Presentation Outline

Introduction
Purpose

The main objectives of this tutorial are to:

1. Inform particle swarm optimization (PSO) practitioners of the many common misconceptions and falsehoods that are actively hindering a practitioner's successful use of PSO; i.e. to
   - separate fact from fiction with evidence
2. Highlight the existing PSO theory that will greatly improve your effectiveness with PSO
   - This knowledge will not only improve your results but also allow you to develop a better intuition for how PSO actually works.

Particle Swarm Optimization
Introduction

What is particle swarm optimization (PSO) [8, 13]?

- a simple, computationally efficient optimization method
- population-based, stochastic search
- individuals follow very simple behaviors:
  - emulate the success of neighboring individuals,
  - but also bias towards own experience of success
- emergent behavior: discovery of optimal regions within a high dimensional search space

Particle Swarm Optimization
Main Components

What are the main components?

- a swarm of particles
- each particle represents a candidate solution
- elements of a particle represent parameters to be optimized

The search process:

- Position updates

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1), \quad \mathbf{x}_{ij}(0) \sim U(x_{min,j}, x_{max,j})$$

- Velocity (step size)
  - drives the optimization process
  - reflects experiential knowledge of the particles and socially exchanged information about promising areas in the search space

Particle Swarm Optimization
Inertia Weight PSO

- used either the star (gbest PSO) or social (lbest PSO) topology
- velocity update per dimension [28]:

$$v_{ij}(t+1) = wv_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_{ij}(t) - x_{ij}(t)]$$

- $v_{ij}(0) = 0$ (preferred [11])
- w is the inertia weight
- c_1, c_2 are positive acceleration coefficients
- $r_{1j}(t), r_{2j}(t) \sim U(0,1)$
- note that a random number is sampled for each dimension

Particle Swarm Optimization

Inertia Weight PSO (cont)

- $\mathbf{y}_i(t)$ is the personal best position calculated as (assuming minimization)

$$\mathbf{y}_i(t+1) = \begin{cases} \mathbf{y}_i(t) & \text{if } f(\mathbf{x}_i(t+1)) \geq f(\mathbf{y}_i(t)) \\ \mathbf{x}_i(t+1) & \text{if } f(\mathbf{x}_i(t+1)) < f(\mathbf{y}_i(t)) \end{cases}$$

- $\hat{\mathbf{y}}_i(t)$ is the neighborhood best position calculated as the best personal best position in particle i's neighborhood

Particle Swarm Optimization

PSO Algorithm

Create and initialize an n_x-dimensional swarm, S;
repeat
  for *each particle* $i = 1, \ldots, S.n_s$ **do**
    if $f(S.\mathbf{x}_i) < f(S.\mathbf{y}_i)$ **then**
      $S.\mathbf{y}_i = S.\mathbf{x}_i$;
    end
    for *each particle* $\hat{i}$ *with particle* i *in its neighborhood* **do**
      if $f(S.\mathbf{y}_i) < f(S.\hat{\mathbf{y}}_{\hat{i}})$ **then**
        $S.\hat{\mathbf{y}}_{\hat{i}} = S.\mathbf{y}_i$;
      end
    end
  end
  for *each particle* $i = 1, \ldots, S.n_s$ **do**
    update the velocity and position;
  end
until *stopping condition is true*;

Neighborhood Topologies

Introduction

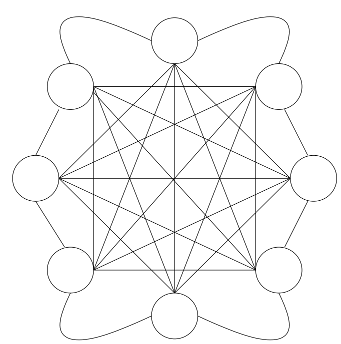
Neighborhood topologies are used to determine the best positions, or attractors, which guide the search trajectories of particles [15, 16]:

- topologies determine the extent of the search space used to determine best positions
- topologies regulate the speed at which information about best positions is transferred through the swarm
- neighborhoods are based on particle indices, not spatial information
- neighborhoods overlap to facilitate information exchange

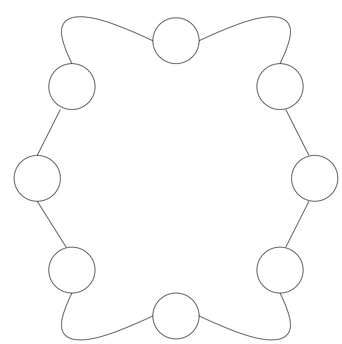
Neighborhood Topologies

Popular Topologies

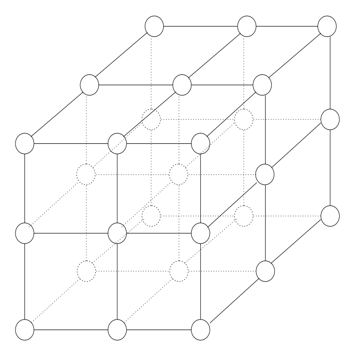
While many neighborhood topologies have been proposed, the most popular ones are



Star Topology (gbest PSO)



Ring Topology (lbest PSO)



Von Neumann Topology

gbest PSO versus lbest PSO

Problem Statement

Original PSO came in two versions, differing in the neighborhood topology used to exchange information about best found positions, i.e.

- **gbest PSO**, using a star neighborhood topology, and
- **lbest PSO**, using a ring neighborhood topology

A general opinion emerged from the PSO community that gbest PSO should not be used, and that lbest PSO should be used due to lbest PSO's [9]

- better exploration ability,
- diminished susceptibility of being trapped in local minima, and
- because it does not suffer from premature convergence.

These opinions are based on very limited empirical evidence and intuitive beliefs about particle behavior

gbest PSO versus lbest PSO

Two Topologies

gbest PSO and lbest PSO differ in the way that neighborhood best positions are updated:

- gbest PSO uses a star neighborhood topology
  - each particle has the entire swarm as its neighborhood
  - $\hat{\mathbf{y}}_i = \hat{\mathbf{y}}$ for all particles $i = 1, \ldots, n_s$
  - consequence: all particles are attracted to one global best position
- lbest PSO uses a ring topology
  - each particle's neighborhood consists of itself and its immediate two neighbours
  - neighborhoods overlap
  - consequence: each particle is attracted to a (initially) different neighborhood best position

gbest PSO versus lbest PSO

General Opinions

Much has been said about the advantages and disadvantages of these two topologies:

- gbest PSO should not be used due to premature convergence to local optima
- gbest PSO converges fast due to faster transfer of best positions throughout the swarm, therefore a strong attraction to one best position
- lbest PSO converges more slowly, and therefore explores more as it maintains diversity for longer
- gbest PSO is more susceptible to being trapped in local minima
- gbest PSO is best suited to unimodal problems and should not be used for multimodal problems
- gbest PSO does not perform well for non-separable problems
- lbest PSO is superior to gbest PSO in terms of solution accuracy for the majority of problems

gbest PSO versus lbest PSO

Empirical Analysis: Algorithm Implementation

Objective: To conduct an extensive empirical analysis to test these general opinions

Two algorithms were implemented to differ only in the neighborhood topology used:

- synchronous position updates
- memory-based personal best position update
- zero initial velocities
- no velocity clamping
- personal best positions updated only if they remain within bounds

Control parameter values:

- $w = 0.729844$
- $c_1 = c_2 = 1.49618$
- 30 particles
- 5000 iterations

gbest PSO versus lbest PSO
Empirical Analysis: Performance Measures

Performance was quantified over 50 independent runs using

- **Accuracy**:
  - average quality of best solution over 50 runs after 5000 iterations
- **Success Rate**:
  - percentage of the 50 independent runs that converged to specific accuracy levels
  - 1000 accuracy levels have been considered, from best obtained accuracy, logarithmically scaled to the worst obtained accuracy
- **Efficiency**:
  - average number of iterations to reach the different accuracy levels
- **Consistency**:
  - deviation from the average best value

gbest PSO versus lbest PSO
Empirical Analysis: Statistical Procedure

- **Accuracy**:
  - paired Mann-Whitney U tests at 0.05 significance level
  - wins and losses calculated per function class
- **Success rate**:
  - Mann-Whitney U test applied on success rates over all of the accuracy levels
  - indicates success rate profile, over all accuracy levels
  - a win indicates that the corresponding algorithm had the most successful runs for most of the accuracy levels
- **Efficiency**:
  - average number of iterations to reach accuracy levels over all accuracy levels
  - a win indicates that the corresponding algorithm converged faster to most accuracy levels

gbest PSO versus lbest PSO
Empirical Analysis: Benchmark Suite

59 boundary constrained problems, of the following types

- uni-modal
- multi-modal
- separable, rotated
- non-separable
- shifted
- noisy
- composition functions

gbest PSO versus lbest PSO
Empirical Analysis: Results (cont)

'$>$' indicates gbest better than lbest, '$<$' gbest worse than lbest, and '$=$' no statistically significant difference

Function Class		Number of Functions	Accuracy			Success Rate			Efficiency			Diversity		
			$>$	$=$	$<$	$>$	$=$	$<$	$>$	$=$	$<$	$>$	$=$	$<$
UM	S	7	5	0	2	6	0	1	2	0	5	5	0	2
	NS	3	2	1	0	2	1	0	2	1	0	2	0	1
	N	2	1	0	1	1	1	0	2	0	0	1	0	1
	Sh	5	2	3	0	2	3	0	2	3	0	1	0	4
	R	1	1	0	0	1	0	0	0	1	0	0	0	1
MM	S	6	1	2	3	2	2	2	3	1	2	6	0	0
	NS	9	4	1	4	3	4	2	4	3	2	1	0	8
	Sh	10	3	4	3	5	5	0	8	1	1	1	0	9
	R	4	0	3	1	1	2	1	2	1	1	0	0	4
	N	1	0	1	0	0	1	0	0	1	0	0	0	1
	C	11	1	2	8	0	4	7	1	5	5	0	0	11
Overall		59	20	17	22	23	23	13	26	17	16	11	0	48
Overall UM		18	11	4	3	12	5	1	8	5	5	9	0	9
Overall MM		41	9	13	19	11	18	12	18	12	11	2	0	39
Overall S		17	7	4	6	9	5	3	12	1	4	5	0	12
Overall NS		42	13	13	16	14	18	10	11	16	9	6	0	36

gbest PSO versus lbest PSO

Empirical Analysis: Consistency

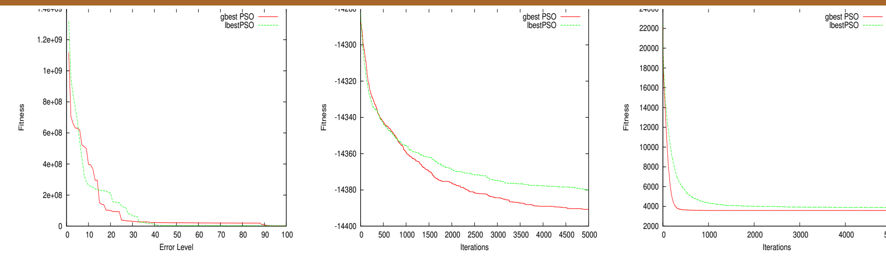
With reference to consistency:

- For 21.7% of the functions did gbest PSO have a significantly smaller deviation than lbest PSO
- For 31.6% of the functions did lbest PSO have a significantly smaller deviation than gbest PSO

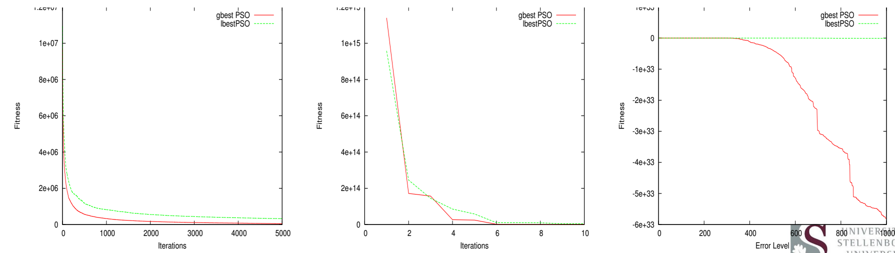
No one of the two topologies can be said to be more consistent than the other

gbest PSO versus lbest PSO

Empirical Analysis: Fitness Profiles



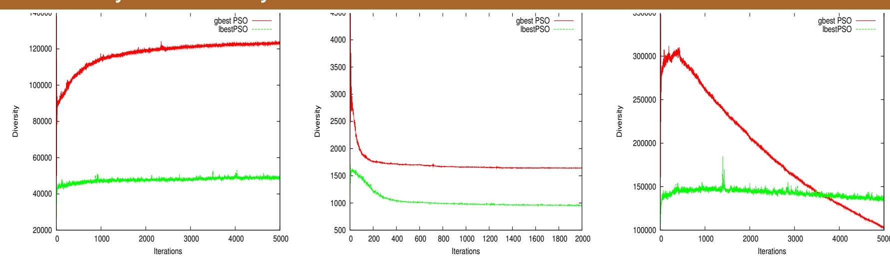
(a) Elliptic (b) Shifted Schaffer 6 (c) Rastrigin



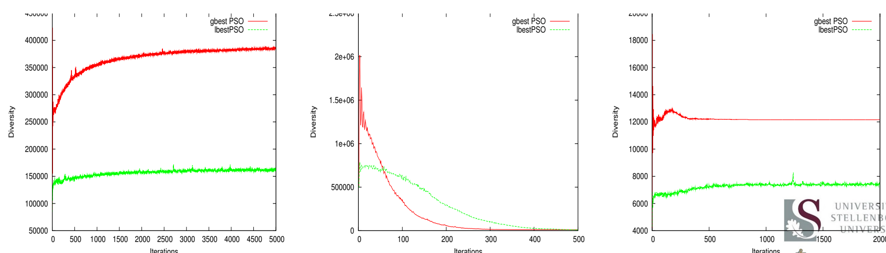
(d) Noisy Shifted Schwefel 1.2 (e) Schwefel 2.22 (f) Shubert

gbest PSO versus lbest PSO

Empirical Analysis: Diversity Profiles



(g) Shifted Rotated Ackley (h) De Jong F4 (i) Rotated Elliptic



(j) Shifted Schaffer 6 (k) Griewank (l) Rastrigin

gbest PSO versus lbest PSO

Observations

The following observations can be made over all the functions:

- gbest and lbest PSO performed very similar with respect to accuracy
- gbest slightly better than lbest with respect to success rate and efficiency
- lbest slightly better than gbest with respect to consistency
- lbest PSO did not maintain diversity for longer than PSO for all functions
- despite the fact that gbest converges faster, it is not at the cost of accuracy nor success rate
- both gbest PSO and lbest PSO sometimes prematurely converge

gbest PSO versus lbest PSO

Observations (cont)

Observations with respect to specific function classes:

- gbest and lbest are equally good at separable and non-separable functions with respect to accuracy
- gbest obtained better success rates than lbest PSO for separable and non-separable functions
- for most of the non-separable functions, there is no significant difference in convergence speed
- lbest was more accurate for a number of unimodal functions
- lbest more accurate for less than half of the multi-modal functions
- lbest did converge faster for a number of unimodal and multi-modal functions

gbest PSO versus lbest PSO

Observations (cont)

Which of gbest PSO or lbest PSO is best?

Based on an extensive empirical analysis, the main conclusions are that

- none of the two algorithms can be considered the preferred algorithm for any of the main function classes
- the best choice is very problem dependent

Velocity Initialization

The Opinions

Velocties have been initialized using any of the following [11]:

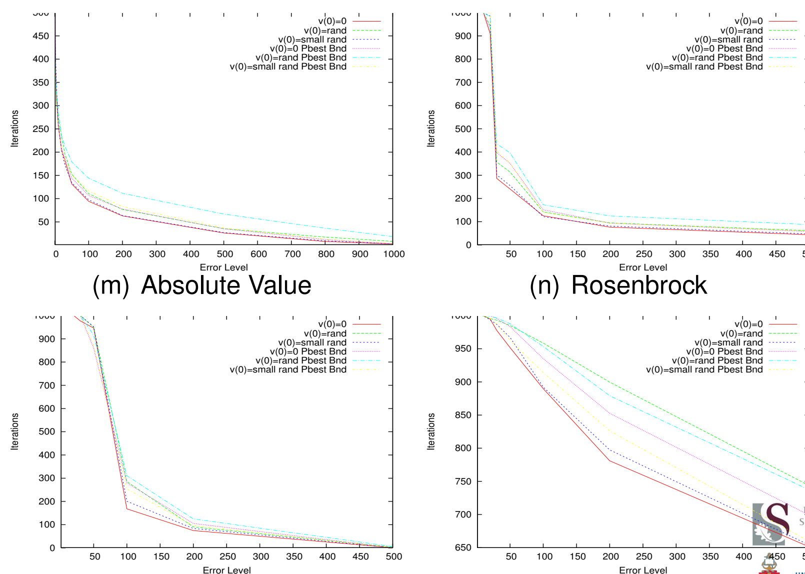
- $\mathbf{v}_i(0) = \mathbf{0}$
  - Critique: Limits exploration ability, therefore extent to which the search space is initially covered
  - Counter argument: Initial positions are uniformly distributed
  - Flocking analogy: Physical objects, in their initial state, do not have any momentum
- $\mathbf{v}_i(0) \sim U(-x_{min}, x_{max})^{n_x}$, where n_x is the problem dimension
  - Argument in favor: Initial random velocities help to improve exploration abilities of the swarm, therefore believed to obtain better solutions, faster
  - Argument against: large initial step sizes cause more particles to leave search boundaries and for longer:

$$\mathbf{v}_i(0) \sim U(-x_{min}, x_{max})^{n_x} \longrightarrow \mathbf{x}_i(1) \sim U(-2x_{min}, 2x_{max})^{n_x}$$

- Initialize to small random values

Velocity Initialization

Fitness Profiles



(m) Absolute Value (n) Rosenbrock

(o) Rastrigin (p) Quadric

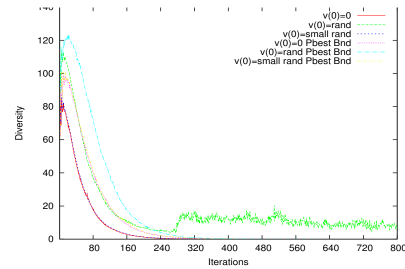
Velocity Initialization

Fitness After 1000 Iterations

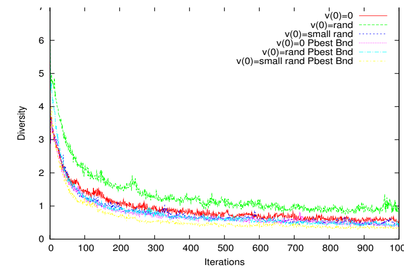
Function	Zero Init No Pbest Bound	Random Init No Pbest Bound
Absolute Value	3.53E-001±2.87E+000	2.46E-001±1.47E+000
Ackley	2.49E+000±1.35E+000	2.68E+000±2.67E+000
Bukin 6	6.20E-002±4.50E-002	6.65E-002±5.56E-002
Griewank	3.72E-002±5.26E-002	3.91E-002±5.57E-002
Quadric	9.04E+001±8.70E+001	1.80E+002±3.15E+002
Rastrigin	6.66E+001±1.71E+001	7.37E+001±2.16E+001
Rosenbrock	2.65E+001±1.53E+001	2.73E+001±1.66E+001

Velocity Initialization

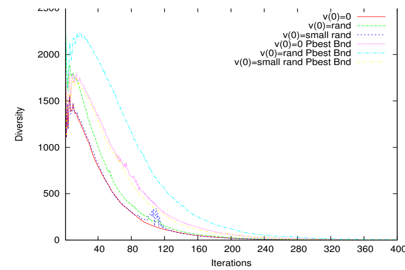
Diversity Profiles



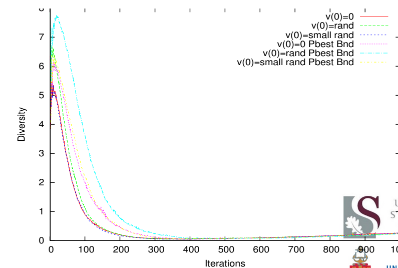
(a) Ackley



(b) Bukin 6



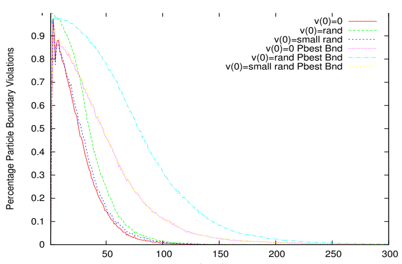
(c) Griewank



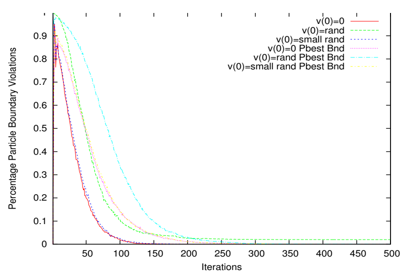
(d) Rosenbrock

Velocity Initialization

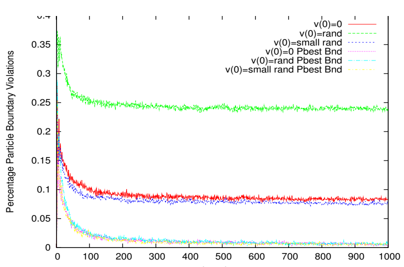
Roaming Behavior: Percentage of Infeasible Particles



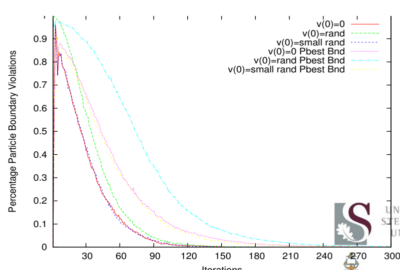
(a) Absolute Value



(b) Ackley



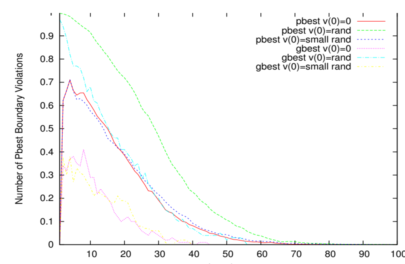
(c) Bukin 6



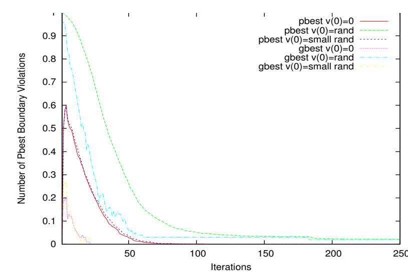
(d) Griewank

Velocity Initialization

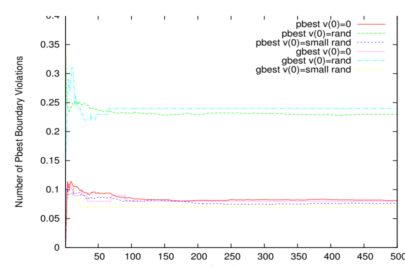
Roaming Behavior: Percentage of Infeasible Personal Bests



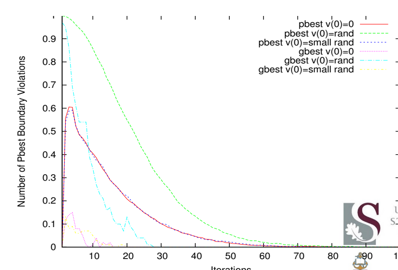
(a) Absolute Value



(b) Ackley



(c) Bukin 6



(d) Griewank

Velocity Initialization

Observations

The following general observations are made:

- Small random initialization and zero initialization have similar behaviors
- Random initialization
  - slower in improving the fitness of the best solution
  - resulted in larger diversity
  - had more roaming particles, roaming for longer
  - significantly more best positions left boundaries
  - took longer to reduce number of particle and best position violations
  - very slow in increasing number of converged dimensions
- Not much of a difference in final accuracies obtained for most of the problems, with random initialization performing poor for some functions

Iteration Strategies

Introduction

Two iteration strategies can be found for PSO [10]:

- Synchronous interation strategy
  - personal best and neighborhood bests updated separately from position and velocity vectors
  - slower feedback of new best positions
- Asynchronous iteration strategy
  - new best positions updated after each particle position update
  - immediate feedback of new best positions
  - lends itself well to parallel implementation

Iteration Strategies

Pseudocode

Synchronous Iteration Strategy

Create and initialize the swarm;
repeat
  for *each particle* **do**
    Evaluate particle's fitness;
    Update particle's personal best position;
    Update particle's neighborhood best position;
  end
  for *each particle* **do**
    Update particle's velocity;
    Update particle's position;
  end

Asynchronous Iteration Strategy

Create and initialize the swarm;
repeat
  for *each particle* **do**
    Update the particle's velocity;
    Update the particle's position;
    Evaluate particle's fitness;
    Update the particle's personal best position;
    Update the particle's neighborhood best position;
  end
until *stopping condition is true*;

Iteration Strategies

Another Debate

- Should a synchronous iteration strategy (SIS) or an asynchronous iteration strategy (AIS) be used?
- General opinions:
  - AIS is generally faster and less costly than SIS
  - AIS generally provides better results
  - AIS is better suited for lbest PSO, while SIS is better for gbest PSO
- Recently, it was shown that SIS generally yields better results than AIS, specifically unimodal functions, and equal to AIS or better for multimodal functions
- It was also recently stated that the choice of iteration strategy is very function dependent

Iteration Strategies
Accuracy Scores

Ranks based on Final Fitness Values

Function Class		Number of Functions	gbest PSO			lbest PSO			GCPSO			BBPSO		
			>	=	<	>	=	<	>	=	<	>	=	<
UM	Sep	7	0	0	7	0	1	6	0	0	7	0	1	6
	Non-sep	3	1	1	1	0	2	1	0	2	1	0	3	0
	Noisy	2	0	0	2	1	1	0	1	0	1	1	0	1
	Shifted	5	0	5	0	0	4	1	0	5	0	0	5	0
	Rotated	1	0	0	1	0	0	1	0	0	1	0	1	0
MM	Sep	6	0	5	1	0	6	0	0	4	2	0	6	0
	Non-sep	9	0	7	2	0	9	0	1	7	1	0	9	0
	Shifted	10	2	6	2	0	10	0	1	7	2	1	8	1
	Rotated	4	0	1	3	0	4	0	1	0	3	1	1	2
	Noisy	1	1	0	0	0	1	0	1	0	0	1	0	0
	Composition	11	7	4	0	0	11	0	7	3	1	10	0	1
Overall Total		59	11	29	19	1	49	9	12	28	19	14	34	11
Overall UM		18	1	6	11	1	8	9	1	7	10	1	10	7
Overall MM		41	10	23	8	0	41	0	11	21	9	13	24	4
Overall Sep		17	1	7	9	1	10	6	0	7	10	0	10	7
Overall Non-sep		42	10	23	9	0	39	3	12	21	9	13	25	4

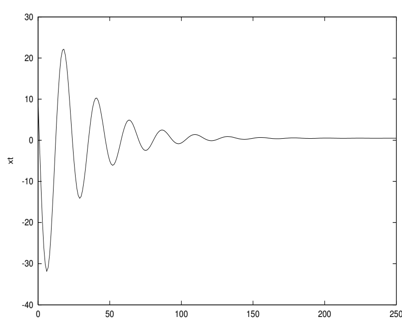
Iteration Strategies
Observations

- Unimodal functions: AIS had better accuracy for most functions
- Multimodal functions:
  - No significant difference for most of the functions
  - For the remainder of the functions, no clear winner
  - For lbest PSO no significant difference over all the functions – insensitive to iteration strategy
- Separable functions: SIS not the preferred strategy for most of the functions
- Non-separable:
  - AIS bad for BBPSO
  - For lbest PSO AIS slightly better than SIS
  - For gbest PSO, GCPSO, SIS slightly better
  - However, for most functions no significant difference

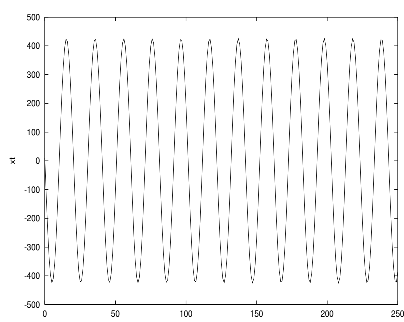
Control Parameters
Introduction

Performance of PSO has been shown to be very sensitive to values assigned to its control parameters

$w = 0.5$ and $c_1 = c_2 = 1.4$



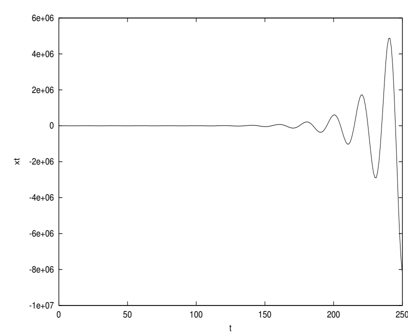
$w = 1.0$ and $c_1 = c_2 = 1.999$



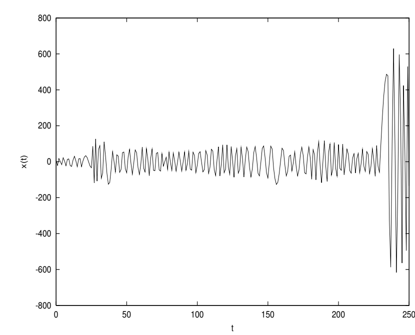
Movement in expectation

Control Parameters
Introduction (cont)

$w = 0.7$ and $c_1 = c_2 = 1.9$



$w = 1.0$ and $c_1 = c_2 = 2.0$



Movement in expectation

Control Parameters
Velocity Components

Performance of PSO has been shown to be very sensitive to values assigned to its control parameters. Where are these control parameters used?

- previous velocity, $w\mathbf{v}_i(t)$
  - inertia component
  - memory of previous flight direction
  - prevents particle from drastically changing direction
- cognitive component, $c_1\mathbf{r}_1(\mathbf{y}_i - \mathbf{x}_i)$
  - quantifies performance relative to past performances
  - memory of previous best position
  - nostalgia
- social component, $c_2\mathbf{r}_2(\hat{\mathbf{y}}_i - \mathbf{x}_i)$
  - quantifies performance relative to neighbors
  - envy

UNIVERSITEIT STELLENBOSCH UNIVERSITY
UNIVERSITEIT VAN PRETORIA UNIVERSITY OF PRETORIA YUNIBESITHI YA PRETORIA

Control Parameters
Inertia Weight, w

- Was introduced to control step sizes
- Can be used to balance exploration-exploration trade-off
  - large values – favor exploration
  - small values – promote exploitation
  - (depending on the values of c_1 and c_2)
- for $w \geq 1$
  - velocities increase over time
  - swarm diverges
  - particles fail to change direction towards more promising regions
- for $0 < w < 1$
  - particles decelerate
  - convergence also dependent on values of c_1 and c_2

UNIVERSITEIT STELLENBOSCH UNIVERSITY
UNIVERSITEIT VAN PRETORIA UNIVERSITY OF PRETORIA YUNIBESITHI YA PRETORIA

Control Parameters
Acceleration Coefficients, c_1, c_2

Weights the contributions of the cognitive and social components:

- $c_1 = c_2 = 0$?
- $c_1 > 0, c_2 = 0$:
  - particles are independent hill-climbers
  - local search by each particle
- $c_1 = 0, c_2 > 0$:
  - swarm is one stochastic hill-climber
- $c_1 = c_2 > 0$:
  - particles are attracted towards the average of $\mathbf{y}_i$ and $\hat{\mathbf{y}}_i$
- $c_2 > c_1$:
  - promotes exploitation
- $c_1 > c_2$:
  - promotes exploration

UNIVERSITEIT STELLENBOSCH UNIVERSITY
UNIVERSITEIT VAN PRETORIA UNIVERSITY OF PRETORIA YUNIBESITHI YA PRETORIA

Control Parameters
What are good parameters for your problem?

One big challenge with using an optimizer is picking which control parameters to use.

- We are now going to test the ability of the audience to guess reasonable control parameters.
  - Interactive demo using CEC2014 benchmark suite.

UNIVERSITEIT STELLENBOSCH UNIVERSITY
UNIVERSITEIT VAN PRETORIA UNIVERSITY OF PRETORIA YUNIBESITHI YA PRETORIA

Despite PSO having many emergent and chaotic properties there are still aspects of its behavior we can predict. We will focus on the following

- **The need for per-dimension stochasticity**
- **Stability of particles in the swarm** (stochastic convergence)
- **Particle movement patterns**
  - Influence of dimensionality and the desired movement pattern
- **Roaming behavior of particles**
  - Effect in low dimensional search spaces versus high dimensional search spaces

In PSO the source of stochasticity comes from the **vectors** $\mathbf{r}_1$ and $\mathbf{r}_2$, where each component is sampled from the uniform distribution $U(0, 1)$

- However, some practitioners have opted to replace them with **scalars**.
- This is a fundamentally poor idea, which will be made clear with a little use of linear algebra

For ease of explanation, consider the situation where velocities are initialized to **0**, and personal best information is derived from the initialized swarm. An unsimplified discussion can be found here [20]

- Let the swarm size be n_s and the dimensionality of the search space be n_x.
- If we use scalars r_1 and r_2 all position generated after the first iteration must be within $span(\mathcal{I})$, where $\mathcal{I} = \{\mathbf{x}_0(0), \mathbf{x}_1(0), \ldots, \mathbf{x}_{n_s}(0)\}$
  - Since all position will be a **linear** combination of

$$(\mathbf{y}_i(0) - \mathbf{x}_i(0)) \text{ and } (\hat{\mathbf{y}}_i(0) - \mathbf{x}_i(0))$$

and $\mathbf{y}_i(0)$ and $\hat{\mathbf{y}}_i(0)$ where derived from the initialized positions

- Furthermore, since all new positions are generated from the *span* of $\mathcal{I}$ we will forever search within $span(\mathcal{I})$
  - Why is the an issue?
  - Note that $span(\mathcal{I}) \subseteq \mathbb{R}^N$, where $N = min\{n_s, n_x\}$
  - If $n_s < n_x$ it implies we search within a **subspace** of our search space $\mathbb{R}^{n_x}$
    - Part of the search space is **unreachable**

Using Theory to Guide PSO Use
The Need for Per-dimension Stochasticity

- If $n_s \geq n_x$ the issue is a little more subtle
  - Firstly the maximum subspace size of $span(\mathcal{I})$ is n_s but we have no guarantee it will be that large.
  - We could get unlucky with the degree of orthogonality in our initial set $\mathcal{I}$ and still only search a subspace.
  - Even if we could guarantee that $span(\mathcal{I}) = \mathbb{R}^{n_x}$, it is possible to lose degrees of freedom,
    - Namely our set from which we can derive new positions loses a degree of orthogonality.
  - We cannot recover a lost degree of orthogonality with scalar r_1 and r_2.

All the above issue are avoided by simply using vector $\mathbf{r}_1$ and $\mathbf{r}_2$, where each component is sampled independently.

Using Theory to Guide PSO Use
Stability of Particles

From a theoretical perspective, the question of particle convergence is probably the most heavily analysed aspects of PSO behavior

- Yet is often misunderstood
- The cause of the confusion, is likely a result of very overloaded terminology

Specifically the word **convergence** is ambiguous in a stochastic context.

Using Theory to Guide PSO Use
Stability of Particles

In the early works on particle convergence of the inertia PSO by Van den Bergh [30], and Trelea [29]:

- The stochastic components were treated as constants
- As a result, the provided criteria of [29, 30] ensure the following type of particle convergence

Definition (Convergent sequence)

The sequence $(\boldsymbol{s}_t)$ in $\mathbb{R}^n$ is convergent if there exists an $\boldsymbol{s} \in \mathbb{R}^n$ such that

$$\lim_{t\to\infty} \boldsymbol{s}_t = \boldsymbol{s} \tag{1}$$

Using Theory to Guide PSO Use
Stability of Particles

However, if we wish to understand the actual PSO, the stochasticity cannot be ignored

- Which brings up the question of what do we mean by convergence in a stochastic context?
- The simplest type of stochastic convergence is in convergence expectation namely:

Definition (Order-1 stability)

The sequence $(\boldsymbol{s}_t)$ in $\mathbb{R}^n$ is order-1 stable if there exists an $\boldsymbol{s}_E \in \mathbb{R}^n$ such that

$$\lim_{t\to\infty} E[\boldsymbol{s}_t] = \boldsymbol{s}_E \tag{2}$$

where $E[\boldsymbol{s}_t]$ is the expectation of $\boldsymbol{s}_t$.

Using Theory to Guide PSO Use
Stability of Particles

While converge in expectation is informative, it leaves out part of the picture, as noted by Poli [24]:

- Even if the **expectation** of a stochastic sequence becomes constant, the **variance** may be increasing
- Consider the random sequence, defined as

$$(\lambda_t) \text{ where } \lambda_t \sim U(-t, t) \text{ for all } t. \tag{3}$$

  Now, the expectation of λ_t is zero for every t, which implies that the sequence (λ_t) is order-1 stable
  - However, the variance of the sequence (λ_t) is increasing over time
  - Clearly (λ_t) is not particularly stable

Using Theory to Guide PSO Use
Stability of Particles

It is for this reason that we need both order-1 and order-2 stability, defined as

Definition (Order-2 stability)

The sequence $(\boldsymbol{s}_t)$ in $\mathbb{R}^n$ is order-2 stable if there exists a $\boldsymbol{s}_V \in \mathbb{R}^n$ such that

$$\lim_{t\to\infty} V[\boldsymbol{s}_t] = \boldsymbol{s}_V \tag{4}$$

where $V[\boldsymbol{s}_t]$ is the variance of $\boldsymbol{s}_t$.

When $\mathbf{S}_v$ must equal zero we term this order-2* stability

- order-2* stability cannot be guaranteed for PSO [3]

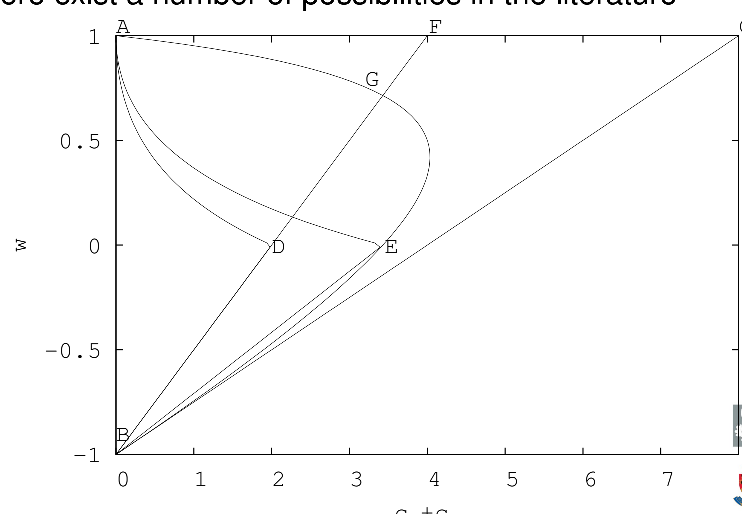
Using Theory to Guide PSO Use
Stability of Particles

In literature, some authors refer to the sequence of particle positions as convergent if it is both order-1 and order-2 stable

- However, the meaning of order-1 and order-2 stability is very different to that of traditional convergence,
- because particle that are order-1 and order-2 stable can **still move**
  - Just with a fixed expectation and variance
  - This can actually be seen as a positive outcome as the swarm can continue to search, provided that the fixed point of the order-2 moment is not 0
    - More on this variance later

Using Theory to Guide PSO Use
Stability of particles in the swarm

- So what are the criteria on control parameters to guarantee order-1 and order-2 stability?
- There exist a number of possibilities in the literature



Using Theory to Guide PSO Use

Stability of particles in the swarm

- The correct region is in fact the curved line segment, AGB
  - Originally derived by Poli and Bromhead [25] and Jiang [14] independently:

$$0 < c_1 + c_2 < \frac{24\left(1 - w^2\right)}{7 - 5w} \quad \text{and} \quad |w| < 1 \tag{5}$$

- The criteria above has also been empirically verified without the presence of simplifying assumptions [5]
- And re-derived recently using what can be shown to be the minimal necessary modeling assumptions by Cleghorn and Engelbrecht [4].

UNIVERSITEIT STELLENBOSCH UNIVERSITY

UNIVERSITEIT VAN PRETORIA UNIVERSITY OF PRETORIA YUNIBESITHI YA PRETORIA

Using Theory to Guide PSO Use

Stability of particles in the swarm

So why does stability matter?

- It tells you where to look for viable parameter configurations
- Specifically, it was shown that parameter configurations that resulted in particle **instability** almost always caused PSO to perform worse than **random search** [7]
  - A particle is unstable if it violates the criteria of equation (5)

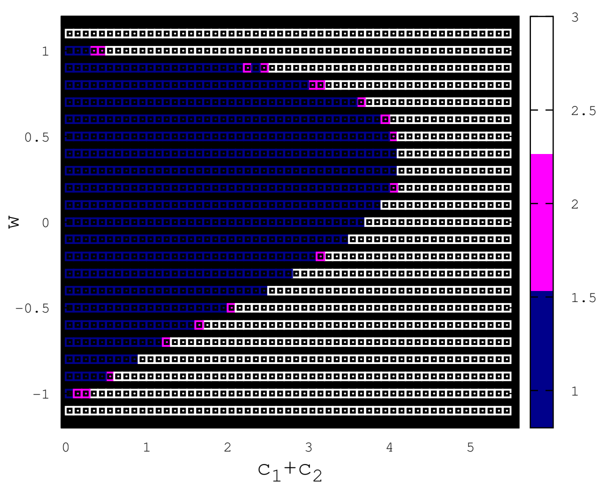
UNIVERSITEIT STELLENBOSCH UNIVERSITY

UNIVERSITEIT VAN PRETORIA UNIVERSITY OF PRETORIA YUNIBESITHI YA PRETORIA

Using Theory to Guide PSO Use

Stability of particles in the swarm

To illustrate the impact of stability on performance consider:



Michalewicz, 30-dimensions, 1000 iterations 1 = performed better than random search, 2 = no statistical difference, 3 = random search performed better

UNIVERSITEIT STELLENBOSCH UNIVERSITY

UNIVERSITEIT VAN PRETORIA UNIVERSITY OF PRETORIA YUNIBESITHI YA PRETORIA

Using Theory to Guide PSO Use

Stability of particles in the swarm

Often times people use a variant of PSO

- Most theory only applies to the inertia and constriction PSO
- However, using the theorem from [4] you can easily derive stability criteria for all variants that can be rewritten in the form

$$x_k(t+1) = x_k(t)\alpha + x_k(t-1)\beta + \gamma_t \tag{6}$$

  where k indicates the vector component, α and β are well defined random variables, and (γ_t) is a sequence of random variables
- Despite the simplicity of equation (6), it caters for a large number of PSOs, such as:
  - Fully informed PSO [17], unified PSO [22], fitness-distance-ratio PSO [23], and multi-guided PSO [26, 27]
  - Furthermore, the mentioned examples are catered for when using any arbitrary well defined distributions

UNIVERSITEIT STELLENBOSCH UNIVERSITY

UNIVERSITEIT VAN PRETORIA UNIVERSITY OF PRETORIA YUNIBESITHI YA PRETORIA

The theorem relies on the non-stagnate distribution assumption,

Definition (Non-stagnant distribution assumption on two informers)

It is assumed that both $\mathbf{y}_i(t)$ and $\hat{\mathbf{y}}_i(t)$ are random variables sampled from a time dependent distribution, such that both $\mathbf{y}_i(t)$ and $\hat{\mathbf{y}}_i(t)$ have well defined expectations and variances for each t and that $\lim_{t\to\infty} E[\mathbf{y}_i(t)], \lim_{t\to\infty} E[\hat{\mathbf{y}}_i(t)], \lim_{t\to\infty} V[\mathbf{y}_i(t)]$, and $\lim_{t\to\infty} V[\hat{\mathbf{y}}_i(t)]$ exist.

Shown to actually be a necessary condition for stability

The theorem has four parts:

Theorem

(1) *Assuming $(\mathbf{i}_t)$ converges, particle positions are order-1 stable for every initial condition if and only if $\rho(\mathbf{A}) < 1$, where*

$$\mathbf{A} = \begin{bmatrix} E[\alpha] & E[\beta] \\ 1 & 0 \end{bmatrix} \text{ and } \mathbf{i}_t = \begin{bmatrix} E[\gamma_t] \\ 0 \end{bmatrix} \quad (7)$$

$\rho(\mathbf{A})$ is the spectral radius of the matrix A, $\rho(\mathbf{A}) = \max_{\lambda\in\Sigma_{\mathbf{A}}} |\lambda|$, $\Sigma_{\mathbf{A}}$ is the set of eigenvalues of $\mathbf{A}$

Theorem

(2) *The particle positions are order-2 stable if $\rho(\mathbf{B}) < 1$ and $(\mathbf{j}_t)$ converges, where*

$$\mathbf{B} = \begin{bmatrix} E[\alpha] & E[\beta] & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & E[\alpha^2] & E[\beta^2] & 2E[\alpha\beta] \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & E[\alpha] & 0 & E[\beta] \end{bmatrix} \quad \text{and} \quad \mathbf{j}_t = \begin{bmatrix} E[\gamma_t] \\ 0 \\ E[\gamma_t^2] \\ 0 \\ 0 \end{bmatrix}$$

under the assumption that the limits of $(E[\gamma_t\alpha])$ and $(E[\gamma_t\beta])$ exist

Theorem

(3) *Assuming that $x(t)$ is order-1 stable, then the following is a necessary condition for order-2 stability:*

$$1 - E[\alpha] - E[\beta] \neq 0 \quad (8)$$

$$1 - E\left[\alpha^2\right] - E\left[\beta^2\right] - \left(\frac{2E[\alpha\beta]\,E[\alpha]}{1 - E[\beta]}\right) > 0 \quad (9)$$

(4) *The convergence of $(E[\gamma_t])$ is a necessary condition for order-1 stability, and the convergence of both $(E[\gamma_t])$ and $(E[\gamma_t^2])$ is a necessary condition for order-2 stability*

To illustrate the power of the presented theorem. Consider again the inertia PSO velocity update equation

$$\begin{aligned} v_i(t+1) &= w\mathbf{v}_i(t) \\ &+ c_1\mathbf{r}_1 \otimes (\mathbf{y}_i(t) - \mathbf{x}_i(t)) \\ &+ c_2\mathbf{r}_2 \otimes (\hat{\mathbf{y}}_i(t) - \mathbf{x}_i(t)) \end{aligned} \quad (10)$$

where $\otimes$ represents component-wise multiplication. However, now let

- $\boldsymbol{\theta}_1 = c_1\mathbf{r}_1$, $\boldsymbol{\theta}_2 = c_2\mathbf{r}_2$
- $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$, and w be **random variables** sampled from **arbitrary distribution**

By applying the presented theorem,

Order-1

$$-1 < E[w] < 1 \quad \text{and} \quad 0 < \frac{E[\theta_1] + E[\theta_1]}{E[w]+1} < 2 \quad (11)$$

Order-2

$$-1 < \frac{E[w]}{\sqrt{1 - V[w]}} < 1 \quad (12)$$

$$0 < E[\theta_1] + E[\theta_1] < \frac{-2(E[w]^2 + V[w] - 1)}{1 - E[w] + \frac{(V[\theta_1]+V[\theta_2])(1+E[w])}{(E[\theta_1]+E[\theta_2])^2}} \quad (13)$$

Let us consider actually doing such derivations. Consider the fully informed PSO:

- The velocity update equation of FIPS is defined as follows:

$$\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + \sum_{m=1}^{|\mathcal{N}_i|} \gamma_m(t) \otimes \frac{(\mathbf{y}_m(t) - \mathbf{x}_i(t))}{|\mathcal{N}_i|} \quad (14)$$

where $\mathcal{N}_i$ is set of particles in particle i's neighborhood, $|\mathcal{N}_i|$ is the cardinality of $\mathcal{N}_i$, and $\gamma_{m,k}(t) \sim U(0, c_1 + c_2)$ for $1 \leq k \leq d$

The approach taken to derive the order-1 stable region is to use theorem 1 (a). Specifically, for FIPS

$$\mathbf{A} = \begin{vmatrix} E[\alpha] & E[\beta] \\ 1 & 0 \end{vmatrix} \qquad \mathbf{i}_t = \begin{vmatrix} E[\gamma_t] \\ 1 \end{vmatrix} \quad (15)$$

where

$$E[\alpha] = (1+w) - \frac{1}{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{N}|} E[\theta_m] = -(1+w) + \frac{\check{c}}{2} \quad (16)$$

$$E[\beta] = -w \quad (17)$$

$$E[\gamma_t] = \frac{1}{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{N}|} E[\theta_m]E[y_m(t)] = \frac{\check{c}}{2|\mathcal{N}|}\sum_{m=1}^{|\mathcal{N}|} E[y_m(t)] \quad (18)$$

- By making the non-stagnant distribution assumption on all particle informers, it follows that $\mathbf{i}_t = \begin{vmatrix} E[\gamma_t] \\ 1 \end{vmatrix}$ converges, since a finite sum of convergent sequences is also convergent.
- Then, we need $\rho(\mathbf{A}) < 1$ to use part (1), which corresponds to the following necessary and sufficient criteria for order-1 stability:

$$|w| < 1 \text{ and } 0 < c_1 + c_2 < 4(w+1) \tag{19}$$

Equation (19) corresponds to the order-1 stable region of FIPS

In order to obtain the necessary conditions for order-2 stability, part (3) of the theorem can be used. Specifically,

Conditions for order-1 and order-2 stability of FIPS

$$|w| < 1 \tag{20}$$

$$0 < c_1 + c_2 < \frac{12|\mathcal{N}|\left(1 - w^2\right)}{3|\mathcal{N}| + 1 + w\left(1 - 3|\mathcal{N}|\right)} \tag{21}$$

In order to confirm that the criteria of equation (20) are in fact sufficient, we need to show that $\rho(\mathbf{B}) < 1$, where

$$\boldsymbol{B} = \begin{bmatrix} E[\alpha] & E[\beta] & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & E[\alpha^2] & E[\beta^2] & 2E[\alpha\beta] \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & E[\alpha] & 0 & E[\beta] \end{bmatrix} \tag{22}$$

Ideally, this should be done analytically but the Eigen values can become incredibly large (symbolic solvers are not great an inequality problems), so we

- Randomly select parameter configurations within the region of equations (20) and (21) (10^9 used)
- It was found that all of generated configurations satisfy $\rho(\mathbf{B}) < 1$. Which is strong evidence that the conditions are in fact sufficient as well
  - Nice research question is to prove when the equivalence between the necessary and sufficient conditions hold.

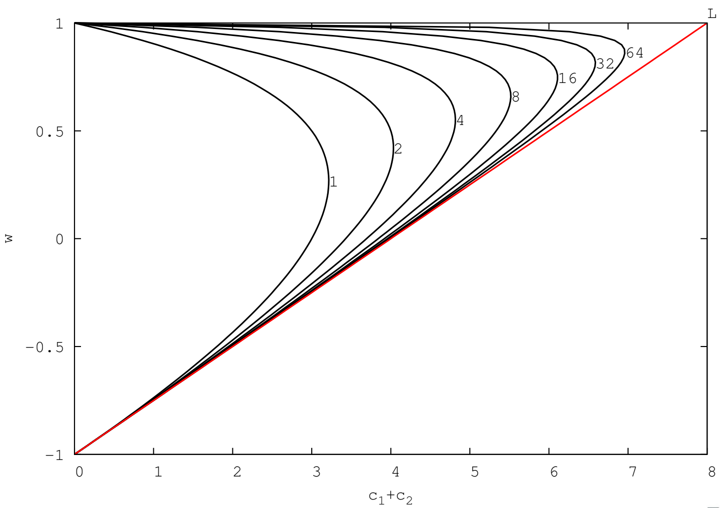
Lastly, we need to show that $\boldsymbol{j}_t$ converges,

$$\boldsymbol{j}_t = \begin{bmatrix} E[\gamma_t] \\ 0 \\ E[\gamma_t^2] \\ 0 \\ 0 \end{bmatrix} \tag{23}$$

The convergence follows directly from simple expansion and the use of the non-stagnant distribution assumption

Using Theory to Guide PSO Use

Utilization of Stability Theorem



Derived order-1 and order-2 stable regions for $|\mathcal{N}| = 1, 2, 4, 8, 16, 32, 64$, and the maximum convergence region

Using Theory to Guide PSO Use

Roaming Behavior of Particles

The problem of particle roaming is a well known issue of PSO

- A particle is said to be roaming if it is moving outside the feasible space.

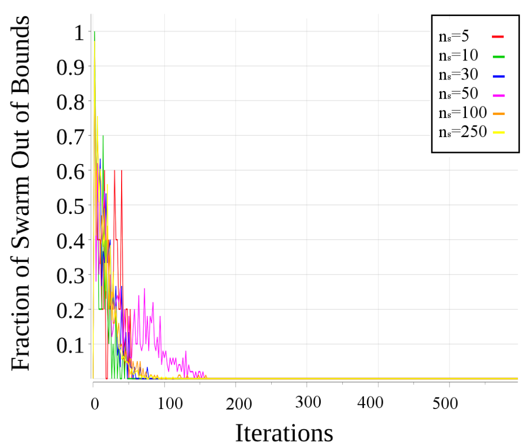
Why do particles roam?

- It was formally proved by Helwig and Wanka [12] that particles will leave the search space with overwhelming probability in the first iteration
  - when velocities are uniform initialized within $[-x_{min}, x_{max}]^{n_x}$ or initialized to **0**.

Using Theory to Guide PSO Use

Roaming Behavior of Particles

In low dimensional search spaces the roaming problem is not so severe. Under the "let them fly" approach [2], particles return to the search space

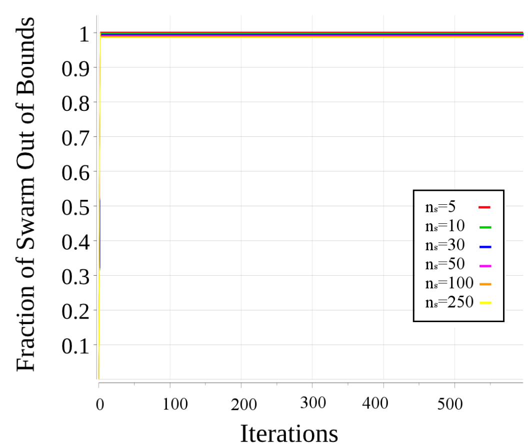


Fraction of swarm outside search space on F7 (CEC2010 large scale optimization benchmark) in 10 dimensions

Using Theory to Guide PSO Use

Roaming Behavior of Particles

However, in high dimensional search spaces the problem of roaming is highly significant



Fraction of swarm outside search space on F7 (CEC2010 large scale optimization benchmark) in 1000 dimensions

How do we handle the problem of particle roaming in high dimensions?

- **Particle variance restriction**:
  - Originally shown by Poli [24], the component-wise variance of the particle positions can be predicted as

$$V[x_i(t)] = \frac{c(5w+1)}{c(54-7) - 12w^2 + 12}(\hat{y}_{ij}(t-1) - y_{ij}(t-1))^2 \qquad (24)$$

    where $c = c_1 = c_2$
- If the variance is restricted, we decrease the likelihood of a boundary violation

How do we handle the problem of particle roaming in high dimensions?

- **Boundary constraint handling**:
  - While there exist many boundary constraint handling approaches they often interact poorly with the explosive PSO dynamics
    - Continuous reinitialization
    - Boundary bias, and often most of the swarm is stuck on the boundary in high dimensional spaces
    - Movement direction warping
- In high dimensions the current best approach is:
  - a per dimension hyperbolic boundary constraint handling mechanism [19]

A more complete exploration of approaches can be found in [18]

While there exists some early research papers on the manner in which particles move through the search space, they where derived in a deterministic context [29, 21]

- Informative when considering the trajectory of a particle in **expectation**, but it does not give us enough information

We can, however, characterize stochastic particle behavior based on the following two aspects:

- Range of motion: from equation (24) we extract the coefficient

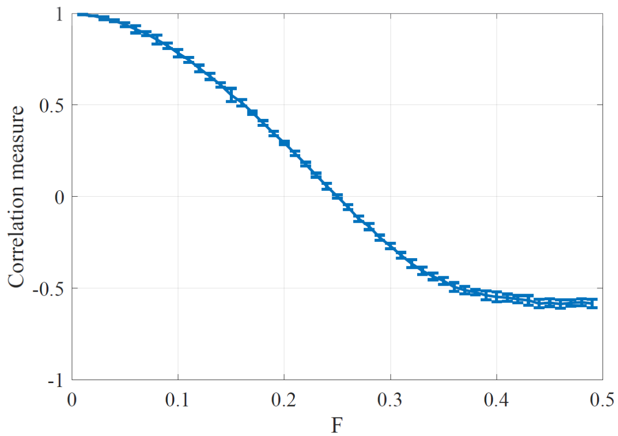
$$V_c = \frac{c(5w+1)}{c(54-7) - 12w^2 + 12} \qquad (25)$$

- Base frequency, F, is defined to be the largest amplitude among the Fourier series coefficients of the particle's positions throughout the search [1]:
  - Particles with small values for F typically exhibit smooth trajectories
  - Particles with large values for F are prone to more oscillations with large steps between positions
- For a given F and V_c the control coefficients can be derived

Using Theory to Guide PSO Use

Particle Movement Patterns

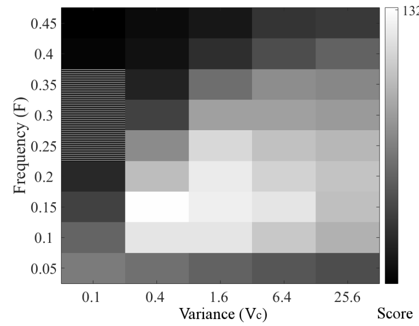
Relationship between base frequency, F, and correlation of particle positions



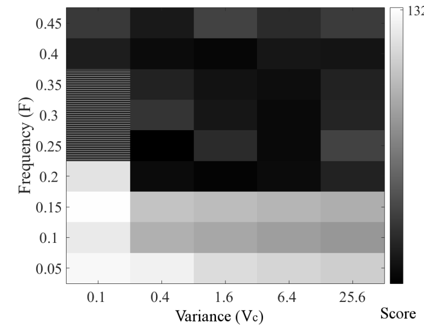
Using Theory to Guide PSO Use

Particle Movement Patterns

The ideal movement pattern is very different in low dimensional search spaces versus high dimensional search spaces.



Optimal frequency-variance combinations (n=10)



Optimal frequency-variance combinations (n=100)

The color of a block shows its score, with lighter indicating a better score across the CEC2010 large scale benchmark suite

Control Parameters

Self-Adaptive Particle Swarm Optimization

Approaches to find the best values for control parameters:

- Just use the values published in literature?
- Fine-tuned static values
- Dynamically changing values
- Self-adaptive control parameters

Many dynamic and self-adaptive approaches have recently been developed

But... more research is needed...

Control Parameters

Self-Adaptive Particle Swarm Optimization: Shortcomings

Issues with current self-adaptive approaches:

- Most, at some point in time, violate convergence conditions, and many do so for most of the search process
- Converge prematurely, with little exploration of control parameter space
- Introduce more control parameters
- Current empirical analysis shows that they do not really result in improved performance with reference to solution quality

Control Parameters

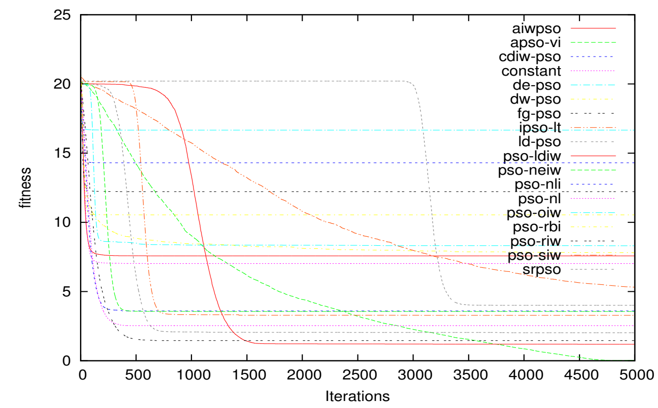
Self-Adaptive Particle Swarm Optimization: Approaches

Optimizer	Parameters Tuned	Net Change
PSO-TVIW (Shi and Eberhart, 1998, 1999)	ω	+1
PSO-AIWF (Liu et al, 2005)	ω	+1
DAPSO (Yang et al, 2007)	ω	+2
IPSO-LT (Li and Tan, 2008)	ω	+1
SAPSO-LFZ (Li et al, 2008)	ω	-1 (0)
SAPSO-DWCY (Dong et al, 2008)	ω	-1 (+2)
PSO-RBI (Panigrahi et al, 2008)	ω	+1
IPSO-CLL (Chen et al, 2009)	ω	-1
AIWPSO (Nickabadi et al, 2011)	ω	+1
APSO-VI (Xu, 2013)	ω	+2
SRPSO (Tanweer et al, 2015)	ω	+2
PSO-SAIC (Wu and Zhou, 2007)	ω, c_2	+2 (+4)
PSO-RAC	ω, c_1, c_2	-3
PSO-TVAC (Ratnaweera et al, 2004)	ω, c_1, c_2	+3
PSO-ICSA (Jun and Jian, 2009)	ω, c_1, c_2	+3 (+31)
APSO-ZZLC (Zhan et al, 2009)	ω, c_1, c_2	-3 (+35)
UAPSO-A (Hashemi and Meybodi, 2011)	ω, c_1, c_2	+6
GPSO (Leu and Yeh, 2012)	ω, c_1, c_2	$+3\ (+(n_d+3))$

Control Parameters

Self-Adaptive Particle Swarm Optimization: Ackley

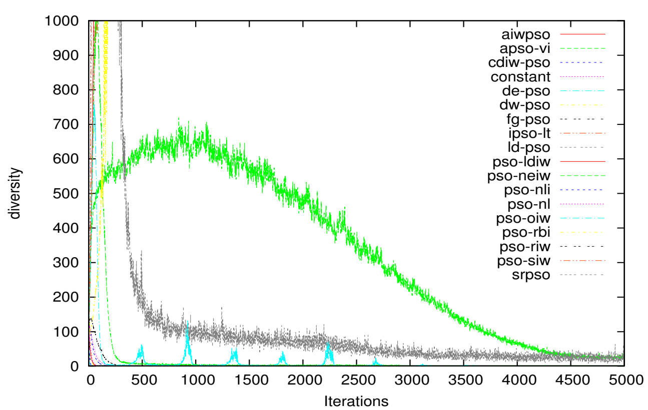
Average solution quality



Control Parameters

Self-Adaptive Particle Swarm Optimization: Ackley (cont)

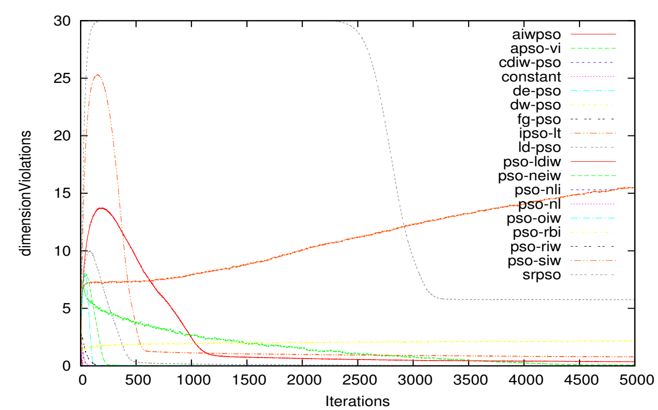
Average swarm diversity



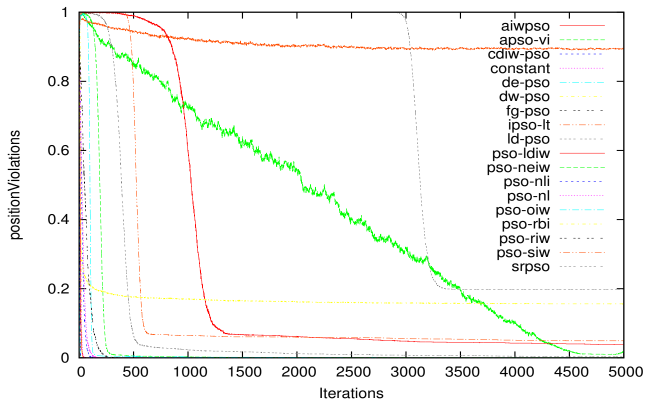
Control Parameters

Self-Adaptive Particle Swarm Optimization: Ackley (cont)

Average boundary violations per dimension



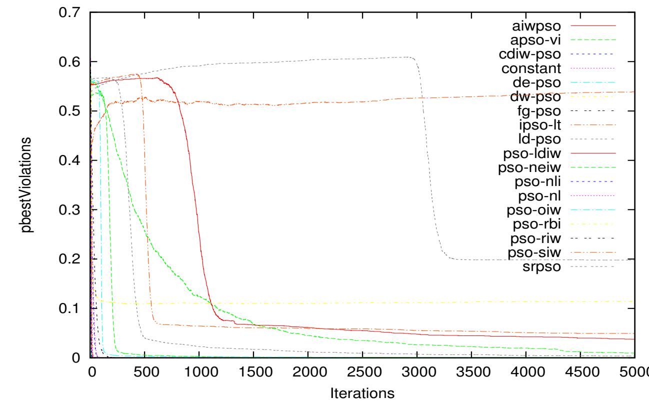
Average percentage particle position boundary violations



Control Parameters

Self-Adaptive Particle Swarm Optimization: Ackley (cont)

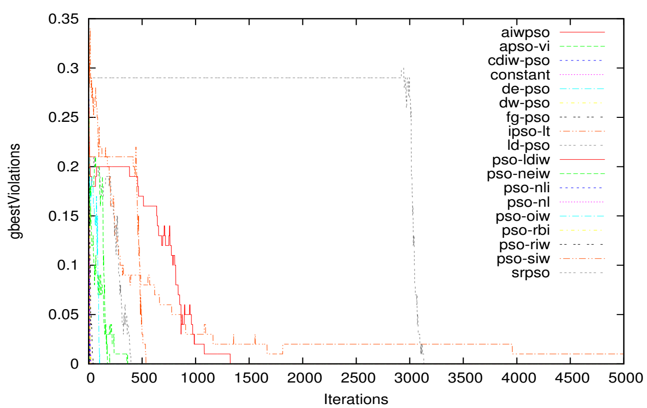
Average percentage personal best position boundary violations



Control Parameters

Self-Adaptive Particle Swarm Optimization: Ackley (cont)

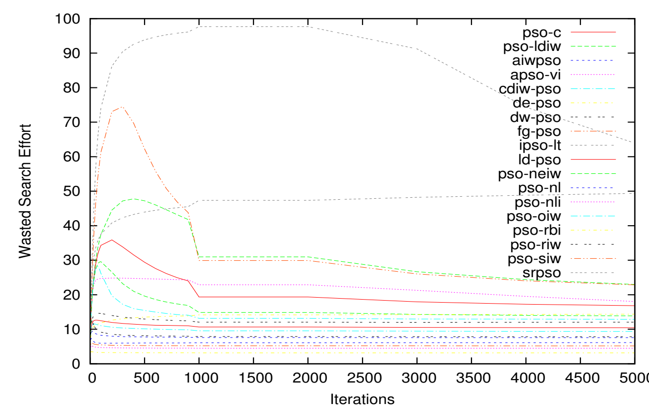
Average global best position boundary violations



Control Parameters

Self-Adaptive Particle Swarm Optimization: Ackley (cont)

Wasted search effort over 60 functions, in dimensions 30, 40, 50, 60, 80, 90, and 100



Uses the specially-formulated function to study convergence behavior [6]:

$$F(\mathbf{x}) \sim U(0, 2000)$$

such that

$$F(\mathbf{x}_1) = F(\mathbf{x}_2) \text{ if } \mathbf{x}_1 = \mathbf{x}_2$$

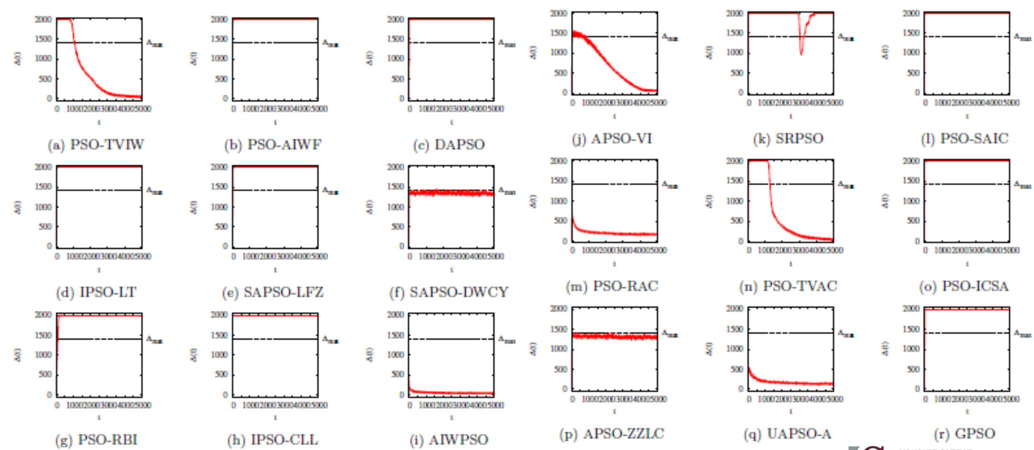
- the fitness value of each position in the search space is randomly sampled within the range $[0, 2000]$
- complete stagnation is highly unlikely
- provides a good benchmark function for studying convergence behavior

Performance measures:

- Average particle movement, Δ:
  - quantifies average particle step size
  - if value does not decrease, particles do not converge
- Percentage particles with convergent control parameters, CP:
  - measures algorithm's ability to generate convergent parameters
- Average parameter movement, Δ_p:
  - average step size in parameter space
  - quantifies stability of the control parameter values
- Percentage particles that violates boundaries, IP:
  - proportion of particles that violates boundary constraints in at least one dimension
  - quantification of wasted search effort

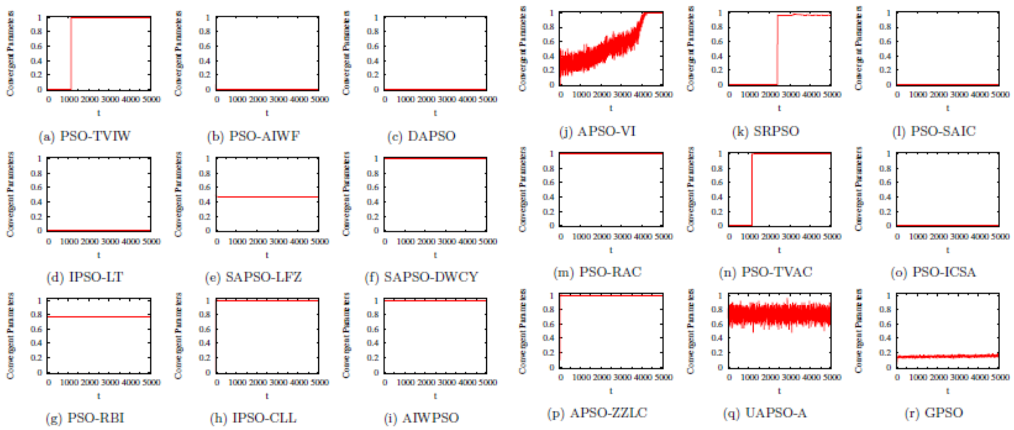
After 5000 interations:

Algorithm	Δ	CP	Δ_p	IP
PSO	415.125	100%	0.0	70.7%
PSO-TVIW	56.489	100%	1.00e-4	9.6%
PSO-AIWF	2000.000	0%	0.0	96.7%
DAPSO	2000.000	0%	NaN	96.9%
IPSO-LT	2000.000	0%	0.0	96.7%
SAPSO-LFZ	2000.000	47.2%	0.0	53.5%
SAPSO-DWCY	1324.322	100%	0.0	96.2%
PSO-RBI	2000.000	76.7%	6.01e-2	41.5%
IPSO-CLL	2000.000	100%	0.0	100%
AIWPSO	45.521	100%	0.0	3.3%
APSO-VI	55.940	100%	0.0	6.1%
SRPSO	2000.000	96.7%	0.0	3.3%
PSO-SAIC	2000.000	0%	NaN	96.7%
PSO-RAC	165.544	100%	1.60e+0	44.2%
PSO-TVAC	32.354	100%	5.74e-4	6.5%
PSO-ICSA	2000.000	0%	4.00e-4	96.7%
APSO-ZZLC	1318.307	100%	4.51e-5	96.1%
UAPSO-A	124.467	70%	8.47e-1	38.1%
GPSO	2000.000	16.7%	8.35e-2	96.7%



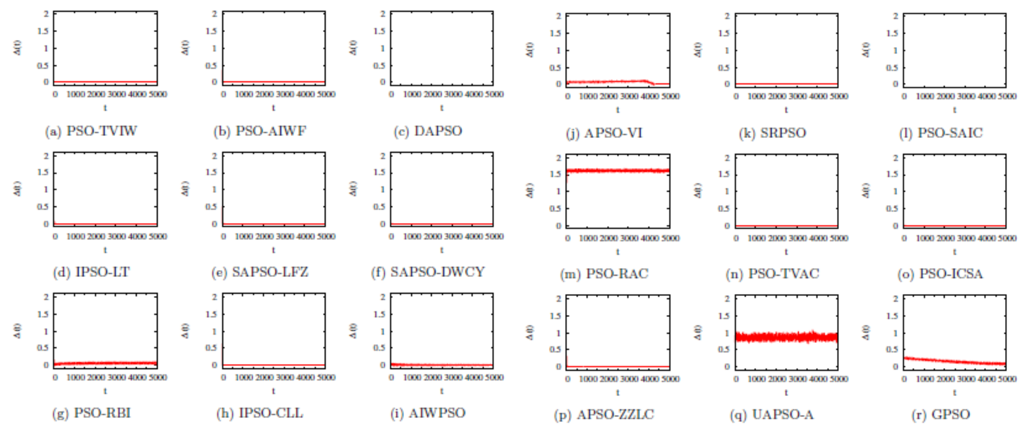
Control Parameters

Self-Adaptive Particle Swarm Optimization: %Convergent Parameters



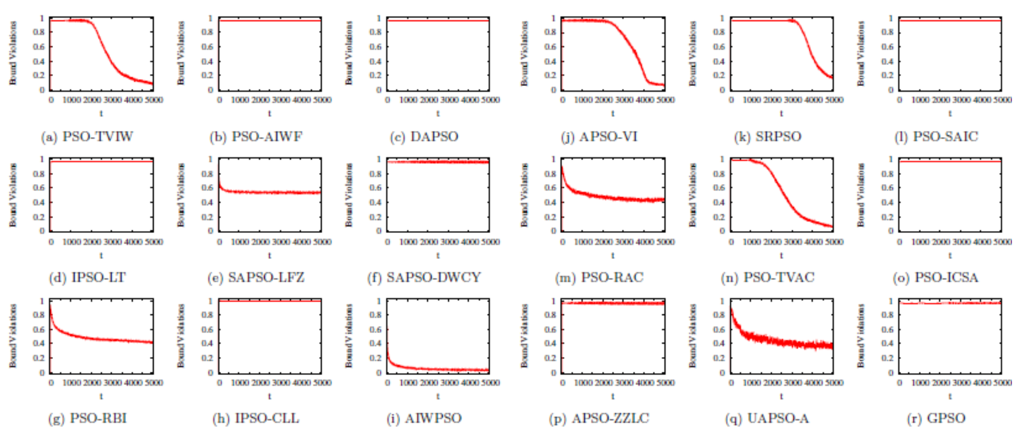
Control Parameters

Self-Adaptive Particle Swarm Optimization: Parameter Movement



Control Parameters

Self-Adaptive Particle Swarm Optimization: Boundary Violations



Concluding Rekmarks

- Yes, PSO has been very successfully applied to solve a wide range of optimization problems
- However, there are a number of aspects about PSO that are not well understood, and many opinions have been made without proper analysis
- This tutorial have identified a number of these misconceptions, and have provided guidance on how to optimally implement PSO, to even furth improve its performance and expands its applications

References I

[1] M. R. Bonyadi and Y. Michalewicz. Impacts of coefficients on movement patterns in the particle swarm optimization algorithm. *IEEE Transactions on Evolutionary Computation*, 21(3):378–390, 2017.

[2] D. Bratton and J. Kennedy. Defining a standard for particle swarm optimization. In *Proceedings of the IEEE Swarm Intelligence Symposium*, pages 120–127, Piscataway, NJ, 2007. IEEE Press.

[3] C. W. Cleghorn. Particle swarm optimization: Understanding order-2 stability guarantees. In *Proceedings of International Conference on the Applications of Evolutionary Computation*, pages 535–549, Switzerland, 2019. Springer International Publishing.

[4] C.W. Cleghorn and A. P. Engelbrecht. Particle swarm stability: a theoretical extension using the non-stagnate distribution assumption. *Swarm Intelligence*, 12(1):1–22, 2018.

References II

[5] C.W. Cleghorn and A.P. Engelbrecht. Particle swarm convergence: An empirical investigation. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 2524–2530, Piscataway, NJ, 2014. IEEE Press.

[6] C.W. Cleghorn and A.P. Engelbrecht. Particle swarm variants: Standardized convergence analysis. *Swarm Intelligence*, 9(2–3):177–203, 2015.

[7] C.W. Cleghorn and A.P. Engelbrecht. Particle swarm optimizer: The impact of unstable particles on performance. In *Proceedings of the IEEE Symposium Series on Swarm Intelligence*, pages 1–7, Piscataway, NJ, 2016. IEEE Press.

[8] R.C. Eberhart and J. Kennedy. A New Optimizer using Particle Swarm Theory. In *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, pages 39–43, 1995.

References III

[9] A.P. Engelbrecht. Particle swarm optimization: Global best or local best? In *Proceedings of the 1st BRICS Countries Congress on Computational Intelligence*, pages 124–135, Piscataway, NJ, 2013. IEEE Press.

[10] A.P. Engelbrecht. Particle swarm optimization: Iteration strategies revisted. In *Proceedings of the 1st BRICS Countries Congress on Computational Intelligence*, pages 119–123, Piscataway, NJ, 2013. IEEE Press.

[11] A.P. Engelbrehct. Particle Swarm Optimization: Velocity Initialization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 2012.

[12] S. Helwig and R. Wanka. Theoretical analysis of initial particle swarm behavior. In *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature*, volume 5199, pages 889–898, New York, 2008. Springer-Verlag.

References IV

[13] Kennedy. J. and R.C. Eberhart. Particle Swarm Optimization. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, pages 1942–1948. IEEE Press, 1995.

[14] M. Jiang, Y.P. Luo, and S.Y. Yang. Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. *Information Processing Letters*, 102(1):8–16, 2007.

[15] J. Kennedy. The Particle Swarm: Social Adaptation of Knowledge. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 303–308, April 1997.

[16] J. Kennedy. Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 3, pages 1931–1938, July 1999.

References V

[17] J. Kennedy and R. Mendes. Neighborhood topologies in fully-informed and best-of-neighborhood particle swarms. In *Proceedings of the IEEE International Workshop on Soft Computing in Industrial Applications*, pages 45–50, Piscataway, NJ, 2003. IEEE Press.

[18] E. T. Oldewage. *The Perils of Particle Swarm Optimisation in High Dimensional Problem Spaces*. PhD thesis, University of Pretoria, South Africa, 2018.

[19] E. T. Oldewage, A.P. Engelbrecht, and C.W. Cleghorn. Boundary constraint handling techniques for particle swarm optimization in high dimensional problem spaces. In *Proceedings of International Swarm Intelligence Conference (ANTS), Swarm Intelligence*, pages 331–341, Switzerland, 2018. Springer International Publishing.

References VI

[20] E. T. Oldewage, A.P. Engelbrecht, and C.W. Cleghorn. The importance of component-wise stochasticity in particle swarm optimization. In *Proceedings of International Swarm Intelligence Conference (ANTS), Swarm Intelligence*, pages 264–267, Switzerland, 2018. Springer International Publishing.

[21] E. Ozcan and C.K. Mohan. Particle swarm optimization: Surfing the waves. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 3, pages 1939–1944, Piscataway, NJ, July 1999. IEEE Press.

[22] K.E. Parsopoulos and M.N. Vrahatis. UPSO: A unified particle swarm optimization scheme. In *Proceedings of the International Conference on Computational Methods in Sciences and Engineering*, pages 868–873, Netherlands, 2004. VSP International Science Publishers.

References VII

[23] T. Peram, K. Veeramachaneni, and C.K Mohan. Fitness-distance-ratio based particle swarm optiniization. In *Proceedings of the IEEE Swarm Intelligence Symposium*, pages 174–181, Piscataway, NJ, 2003. IEEE Press.

[24] R. Poli. Mean and variance of the sampling distribution of particle swarm optimizers during stagnation. *IEEE Transactions on Evolutionary Computation*, 13(4):712–721, 2009.

[25] R. Poli and D. Broomhead. Exact analysis of the sampling distribution for the canonical particle swarm optimiser and its convergence during stagnation. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 134–141, New York, NY, 2007. ACM Press.

References VIII

[26] C. Scheepers. *Multi-guided Particle Swarm Optimization: A Multi-objective Particle Swarm Optimizer*. PhD thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2018.

[27] C. Scheepers and A. P. Engelbrecht. Multi-guide particle swarm optimization a multi-swarm multi-objective particle swarm optimizer. *Swarm Intelligence (under review)*, pages 1–22, 2018.

[28] Y. Shi and R.C. Eberhart. A Modified Particle Swarm Optimizer. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 69–73, May 1998.

[29] I.C Trelea. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Information Processing Letters*, 85(6):317–325, 2003.

[30] F. Van den Bergh and A.P. Engelbrecht. A study of particle swarm optimization particle trajectories. *Information Sciences*, 176(8):937–971, 2006.