Analysis of the particle swarm optimization algorithm

by

Daniel N. Wilke

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Master of Engineering

in the Department of Mechanical and Aeronautical Engineering,
University of Pretoria

February 2005

Abstract

Title: Analysis of the particle swarm optimization algorithm

Author: Daniel Nicolas Wilke

Supervisors: Prof. A.A. Groenwold
Dr. S. Kok

Department: Department of Mechanical and Aeronautical Engineering

Degree: Master of Engineering

Keywords: Particle swarm optimization, diversity, trajectory collapse,
line searches, observer independence, invariance, shape optimization

Increasing prominence is given to the role of optimization in engineering. The global optimization problem is in particular frequently studied, since this difficult optimization problem is in general intractable. As a result, many a solution technique have been proposed for the global optimization problem, e.g. random searches, evolutionary computation algorithms, taboo searches, fractional programming, etc. This study is concerned with the recently proposed zero-order evolutionary computation algorithm known as the particle swarm optimization algorithm (PSOA). The following issues are addressed:

1. It is remarked that implementation subtleties due to ambiguous notation have resulted in two distinctly different implementations of the PSOA. While the behavior of the respective implementations is markedly different, they only differ in the formulation of the velocity updating rule.

   In this thesis, these two implementations are denoted by PSOAF1 and PSOAF2 respectively.

2. It is shown that PSOAF1 is observer independent, but the particle search trajectories collapse to line searches in n-dimensional space.

   In turn, for PSOAF2 it is shown that the particle trajectories are space filling in n-dimensional space, but this implementation suffers from observer dependence.

It is also shown that some popular heuristics are possibly of less importance than originally thought; their greatest contribution is to prevent the collapse of particle trajectories to line searches.

3. A novel PSOA formulation, denoted PSOAF1* is then introduced, in which the particle trajectories do not collapse to line searches, while observer independence is preserved. However, the observer independence is only satisfied in a stochastic sense, i.e. the mean objective function value over a large number of runs is independent of the reference frame.

   Objectivity and effectiveness of the three different formulations are quantified using a popular unimodal and multimodal test set, of which some of the multimodal functions are decomposable. However, the objective functions are evaluated in both the unrotated, decomposable reference frame, and an arbitrary rotated reference frame.

4. Finally, a practical engineering optimization problem is studied. The PSOA is used to find the optimal shape of a cantilever beam. The objective is to find the minimum vertical displacement at the edge point of the cantilever beam. In order to calculate the objective function the finite element method is used. The meshes needed for the linear elastic finite element analysis are generated using an unstructured remeshing strategy. The remeshing strategy is based on a truss structure analogy.

Opsomming

Titel:	Analise van die partikel swerm optimeringsalgoritme
Outeur:	Daniel Nicolas Wilke
Leier:	Prof. A.A. Groenwold Dr. S. Kok
Departement:	Departement Meganiese en Lugvaartkundige Ingenieurswese
Graad:	Meester van Ingenieurswese
Sleutelwoorde:	partikel swerm optimering, diversiteit, baan ineenstorting, lyn soektogte, waarnemer onafhanklikheid, invariansie, vorm optimering

Toenemende belangrikheid word aan die rol van optimering in ingenieurswese gegee. Veral die globale optimeringsprobleem word dikwels bestudeer, aangesien hierdie moelike optimeringsprobleem in die algemeen onoplosbaar is. Gevolglik is daar voorheen al verskeie oplossingstegnieke voorgestel vir die globale optimeringsprobleem, soos byvoorbeeld lukrake soektogte, evolusionêre berekeningsalgoritmes, taboe soektogte, fraksionele programmering, ens. Hierdie studie is vermoeid met die onlangs gepostuleerde nulde-orde evolusionêre berekeningsalgoritme wat bekend staan as die partikel swerm optimeringsalgoritme (PSOA). Die volgende kwessies word bespreek:

1. Daar word opgemerk dat twee verskillende formulerings van die PSOA bestaan, moontlik as gevolg van onduidelike notasie. Alhoewel die gedrag van die onderskeie implementerings dramaties verskil, verskil hulle slegs ten opsigte van die formulering van die snelheidsopdateringswet.

   In hierdie tesis word die onderskeie implementerings as PSOAF1 en PSOAF2 aangedui.

2. Verder word aangetoon dat PSOAF1 waarnemer onafhanklik is, maar dat die partikel bane in n-dimensionele ruimte na lyn soektogte ineenstort.

   Om die beurt, word daar vir PSOAF2 aangetoon dat die partikel bane ruimtevullend is in n-dimensionele ruimte, maar hierdie implementering is waarnemer afhanklik.

   Daar word ook gewys dat sommige gewilde heuristieke moontlik van minder belang is as

wat oorspronklik geag is. Daar word gewys dat hulle grootste bydrae waarskynlik is om partikel baan ineenstorting na lyn soektogte te voorkom.

3. ’n Nuwe PSOA formulering word dan voorgestel, naamlik PSOAF1*. Partikel trajekte stort nie na lyn soektogte ineen nie, terwyl waarnemer onafhanklikheid behou word. Waarnemer onafhanklikheid word egter slegs in ’n stogastiese sin bevredig, m.a.w. die gemiddelde doelwit funksie waarde is onafhanklik van ’n koördinaatstelsel, gesien oor ’n groot aantal verlope.

   Objektiwiteit en effektiwiteit van die drie formulerings word gekwantifiseer deur gebruik te maak van ’n gewilde unimodale en multimodale toets stel, waarvan die meerderheid multimodale funksies skeibaar is. Nietemin word die doelwit funksies geëvalueer in beide die ongeroteerde, skeibare, verwysingsraamwerk en ’n lukraak geroteerde verwysingsraamwerk.

4. Laastens word ’n praktiese ingenieurs optimeringsprobleem bestudeer. Die PSOA word aangewend om die optimale geometrie van ’n kantelbalk te vind. Die doelfunksie wat geminimeer word is die vertikale verplasing by die eindpunt van die kantelbalk. Die doelfunksies word bereken deur gebruik te maak van die eindige element metode. Die mase wat benodig word vir die linieêr elastiese eindige element analises word gegenereer deur van ’n ongestruktureerde hermasings-strategie gebruik te maak. Die hermasings-strategie is gebaseer op ’n vakwerk struktuur analoog.

Acknowledgments

Some have the pleasure of seeing distinguished men in their lifetime, some have the privilege of meeting them, but I had the honor to undertake a journey alongside them.
- D.N. Wilke
University of Pretoria, 2004.

I would like to dedicate this thesis to
my father and my mother

This section attempts to capture a mere snapshot of my thoughts and experiences over the last two years. This section is written in the form of an informal short story. To anyone who may feel affronted in any way by the writing style or content of this section, I offer my sincere apologies.

Conceit of the Absurd - A sailor's story

This story begins in the year 2002, as I reached my last year of formal enrollment as a sailor. My fellow sailor friends and I used to meet up at a local tap and talk about the adventures to come, and of possible treasures of gold.

As the year progressed I was adamant that I would set sail the following year for either Europe or the middle East, as I considered some lucrative offers of possible gold treasures and adventure in these distant and uncharted lands.

As the year came to an end, the sea tides turned.

I met Captain Groenwold and in the end I decided to trade adventures of Europe, and the middle East for adventures of another kind. The adventures that are about the journey and not the destination.

As the year 2003 dawned, Captain Groenwold and I embarked on an adventure. The adventure started calmly by sailing out of the harbor on a brig, affectionately referred to by the sailors as SORG. At the boat's command stood Captain Groenwold and on the deck stood I, a proud sailor.

We left the harbor and sailed into the open seas, that laid open for traveling and exploration. This was my first time out on the ocean. As the harbor disappeared on the distant horizon I did not quite

know where I was, but I knew I was out there, somewhere. Months past as we encountered some light breezes and stormy clouds here and there.

Then one day Captain Groenwold got word that he was needed in a distant land. We anchored the vessel at the nearest harbor and on board came co-Captain Kok to take command of the brig. Our only contact with Captain Groenwold being the infamous message bottle system.

The vessel sailed further under co-Captain Kok's command, over the calm seas and oceans. Every now and then we would spot land here and there, for me the excitement grew as the frequency of sightings of land increased. Some days the breezes became stronger than others as we set forth towards the land. Not before long by the middle of 2004, the adventure turned into an epic of Gulliver's travels as we finally reached land. The epic started by us nearly shipwrecking the preceding day after I misread the map. Fortunately, co-Captain Kok instantly realized my mistake and recovered our situation. Nevertheless we anchored and set forth our exploration.

For months we set sail and anchored to explore various places, each an exotic place in it's own right. On one of the stops we picked up Captain Groenwold after his return from the distant lands. The adventure continued, I was constantly fascinated by each place, and not before long it dawned. What seemed to be vast and distant lands was actually one big island. We attempted to map what we could of this beautiful island but as our supplies where running low, we had to set forth the journey back home.

By the beginning of 2005, the adventure came to an end as our ship sailed into the harbor. As I disembarked our ship and touched home soil I realized "*I embarked on the right adventure*".

Formally I would like to express my sincere gratitude towards the following persons:

- Prof. Groenwold for his discussions and philosophy that contributed to me starting this study, without whose guidance I never would have considered this study. Furthermore, for his guidance during this study, in particular for his patience with my writing.
- Dr. Kok for making this study a scientific voyage in the true sense of Gulliver's travels. Especially for the exploratory discussions and priceless hours of philosophy over plenty a cup of coffee in front of the Aula. For the profound effect he has had on my life, especially with respect to science.
- All my friends for keeping the ever vital balance between the academic and social spheres of my life, *ergo bibamus*.
- My father and my mother to whom I dedicate this thesis, I thank you for your continuous and steadfast support throughout my life.

Financial support granted by the National Research Foundation of South Africa (GUN:2059962) is gratefully acknowledged.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Global optimization

Increasing prominence is given to the role of optimization in engineering, both in academia and in industry. Computational power, combined with increased algorithm flexibility and simplicity, allows for a faster transition of algorithm development in academia to industry.

In convex optimization, the local minimizer is characterized by the Karush-Kuhn-Tucker conditions [1]. These conditions are necessary and *sufficient* to guarantee optimality.

In global optimization in general, no conditions are available to characterize the global optimizer, and the optimization problem is intractable.

The difficulty of the general global optimization problem is further aggravated by

1. the presence of numerical noise,
2. the presence of infeasible regions in the design domain,
3. the presence of discontinuities,
4. a large number of design variables, and
5. the computational cost of evaluating an expensive objective function (simulation).

The large variety of solution techniques in global optimization is therefore not surprising. Recent developments includes evolutionary computational algorithms, taboo searches, fractional programming, dynamical searches, etc. [2].

This study is concerned with the evolutionary particle swarm optimization algorithm (PSOA), introduced by Kennedy and Eberhart [3, 4]. Some advantages of the PSOA are that it is

1. gradient-free,
2. easy to parallelize, and
3. simple and easy to implement.

1.2 Motivation

PSOA investigations are predominantly based on empirical test functions [5, 6, 7, 8, 9]. In these investigations, the complexities of the algorithm are not broken into digestible units. An understanding of the fundamentals of the algorithm is therefore not easily achieved. In these studies, judgment is largely passed based purely on the final objective function values that a particular algorithm obtains.

Some researchers conduct empirical investigations to quantify the sequence effect of the algorithm by observing the particle positions in the design domain at selected iterations for a single run [10]. Alternatively, the trace of positions of the swarm over a single run is observed [11, 12]. Although some insight is gained into the behavior and convergence of the swarm, the stochastic nature of the algorithm is neglected.

Ozcan and Mohan [13] conducted a deterministic trajectory analysis of a single particle by neglecting the stochasticity of the algorithm. Clerc and Kennedy [14] extended the analysis to the stochastic dynamic behavior of an individual particle in complex space, with the main focus on constriction coefficients and achieving desirable dynamic characteristics for a particle. Trelea [15] averaged the stochasticity in the PSOA, in an investigation based on dynamic system theory, to increase the understanding of the dynamics of an individual particle. Zheng et al. [16] extended the study of Ozcan and Mohan [13] and concluded that the inertia weight should increase over time.

Although extensive research on the PSOA has been conducted, a basic understanding of the algorithm still seems lacking. It is the aim of this study to gain fundamental insight into the PSOA. It will be shown that there exist two different formulations of the PSOA. These formulations only differ in the formulation of the velocity updating rule.

1.3 Objectives

The four objectives of this study are

1. to discern between the two *different* formulations of the PSOA,
2. to investigate *observer independence* of the PSOA,
3. to introduce a *novel* observer independent PSOA with diverse, space filling particle trajectories, and
4. to utilize the PSOA in *shape optimization*.

1.4 Approach

In attaining each objective, a different approach is followed:

1. In order to discern between the different formulations, elementary linear algebra is used. In order to quantify the differences, a numerical study is conducted.

2. The investigation of observer independence is conducted using Monte Carlo simulations. For observer independence both translational and rotational invariance have to apply. Objectivity is quantified with a numerical study in both the unrotated, and rotated reference frames.

3. Knowledge developed in 1. and 2. above is then used to develop a novel PSOA formulation.

4. In order to utilize the PSOA in shape optimization, an unstructured remeshing strategy is implemented. This allows for large variations in designs, prominent in the initial phases of the PSOA searches.

1.5 Thesis overview

The chapters in this thesis are self contained.

In Chapter 2, the problem formulation under consideration in Chapters 3 and 4 is presented, as well as a very brief overview of the formulation of the PSOA, to allow for a brief historical overview of the PSOA.

In Chapter 3, a detailed analysis of the particle swarm optimization algorithm (PSOA) is presented. It is shown that implementation subtleties due to ambiguous notation have resulted in two distinctly different implementations of the PSOA, which have been used indiscriminately and unwittingly within the optimization community. However, discerning between these two implementations is shown to be of crucial importance. While the behavior of the respective implementations is markedly different, they only differ in the formulation of the velocity updating rule. In fact, the differences are merely due to subtle differences in the introduction of randomness into the algorithm. For a population of p particles, it is shown that for the first implementation, the particle trajectories collapse to p line searches. The second implementation does not suffer this drawback. Instead, diverse stochastic search trajectories are retained. It is then shown that some popular heuristics like maximum velocity limit, position restriction, craziness and high initial velocities are possibly of less importance than originally thought; their greatest contribution is that they prevent the collapse of particle trajectories to lines. Finally, it is emphasized that the determination of optimal values for parameters like inertia, velocity limit, etc. has to be performed within the context of the formulation used. To this extent, a proposed list of parameters and implementational issues that should be reported when 'tuning' the PSOA is given.

In Chapter 4, the ability of the particle swarm optimization algorithm (PSOA) to satisfy objectivity, also called observer independence or frame indifference, is investigated. In Chapter 3 it was shown that implementation subtleties have resulted in two distinctly different implementations of the PSOA. The first implementation is now shown to be observer independent. In turn, the second implementation of the PSOA is shown to suffer from observer dependence. A novel formulation of the PSOA, in which the particle trajectories do not collapse to line searches, while observer independence is preserved, is then introduced. However, the observer independence is only satisfied in a stochastic sense, i.e. the mean objective function value over a large number of runs is independent of the reference frame. Objectivity and effectiveness of the three different formulations

are quantified using a popular test set. The objective functions are evaluated in both the unrotated reference frame, and an arbitrary rotated reference frame.

In Chapter 5, the PSOA is combined with an unstructured remeshing shape optimization environment. The remeshing strategy creates unstructured meshes from triangular elements, based on the truss structure analogy proposed by Persson and Strang [17]. The PSOA is then used to search for optimal shapes. Results for a popular beam problem are presented.

Finally, conclusions and recommendations are offered in Chapter 6.

Chapter 2

Problem formulation and background

In this chapter, the problem formulation under consideration in Chapters 3 and 4 is presented, as well as a very brief overview of the formulation of the PSOA, to allow for a brief historical overview of the PSOA.

2.1 Global optimization problem formulation

For the sake of brevity, we restrict ourselves to the unconstrained or bounded constrained multi-modal global optimization problem which we will define as follows:

Find the global minimum value $f(\boldsymbol{x}^*)$ of a given real-valued function $f : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$, such that

$$f^* = f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}),\ \forall\, \boldsymbol{x} \in D, \tag{2.1}$$

where D is the allowable (bounded) design domain. Since this problem is intractable, the aim is usually to find a suitably low approximation $\bar{f}$ to f^*.

2.2 Basic formulation of the PSOA

Consider a swarm of p particles in an n-dimensional design space. The position vector $\boldsymbol{x}_k^i$ of each particle i is updated by

$$\boldsymbol{x}_{k+1}^i = \boldsymbol{x}_k^i + \boldsymbol{v}_{k+1}^i, \tag{2.2}$$

where k is a unit pseudo time increment (iteration number). $\boldsymbol{v}_{k+1}^i$ represents the velocity vector that is obtained from the velocity rule, given by

$$\boldsymbol{v}_{k+1}^i = w\boldsymbol{v}_k^i + \boldsymbol{\nu}_k^i, \tag{2.3}$$

where the inertia factor w [8] is a real number, typically between 0.4 and 0.9, and $\boldsymbol{\nu}_k^i$ is the stochastic 'velocity' vector.

In turn, the term $\boldsymbol{\nu}_k^i$ consists of the summation of the terms $c_1(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $c_2(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$, which are however randomly scaled in a to be specified way. $\boldsymbol{p}_k^i$ represents the best position vector of

particle i, while $\boldsymbol{p}_k^g$ represents the global best position vector of the complete swarm, up to time step k. The cognitive and social scaling factors c_1 and c_2 are real numbers; both are usually equal to 2. By selecting $c_1 = c_2$, the cognitive and social contributions are weighed equally.

The vectors $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ utilize the magnitudes and directions from a given particle's current position $\boldsymbol{x}_k^i$ to the particle's best position $\boldsymbol{p}_k^i$, and to the global best position $\boldsymbol{p}_k^g$.

2.3 Brief history of PSOA

The particle swarm optimization algorithm (PSOA) was introduced by Kennedy and Eberhart [3, 4] in 1995. Since then, the PSOA has been applied to optimization problems in a variety of disciplines. To name but a few, neural network training [3, 4, 5, 6], biochemistry [18], manufacturing [10], electromagnetism [11, 19, 20], electrical power [21, 22], optics [12] and structural optimization [23, 24].

The initial implementations of the PSOA contained two parameters, namely the maximum velocity limit [4] and the acceleration constant. The acceleration constant consists of two parts referred to as the cognitive and social constants c_1 and c_2 [25]. The cognitive and social constants are usually implemented statically, although dynamic implementations have also been studied [26]. Kennedy and Eberhart [3] initially proposed that $c_1 = c_2 = 2$.

Shi and Eberhart [8] introduced an additional parameter, referred to as the inertia weight w. Static and dynamic implementations of the inertia weight exist. Three dynamic inertia implementations are the frequently encountered reducing inertia [8, 26, 27], increasing inertia [16] and unstructured inertial adjustment [28]. Recently, Clerc [14, 29] introduced a new formulation, referred to as constriction.

Particle position limits can be implemented to ensure that the particle searches are confined to the defined design domain. The limits are usually implemented as constraints on each design variable. There are various strategies to accommodate the boundary constraints or to enforce position limitation. A complete discussion is given by Robinson and Rahmat-Samii [30].

The evaluation of the objective function value at a given iteration can occur either synchronously or asynchronously [31]. In the synchronous update method, the swarm's positions are updated before the objective function evaluations occur at the updated positions. Hence the particle best and global best positions can only be updated at the end of each iteration. In contrast, in the asynchronous implementation, the objective function is evaluated directly after a position update for a particle occurs. Therefore, the updates of the particle best position and the global best position occur after each particle updates its position. In the synchronous implementation all the particles move at once, whereas in the asynchronous update implementation, the particles move one after the other.

The PSOA methodology was extended through various hybrid PSOA implementations. Hybrid PSOA's can change the applicability of the algorithm, by incorporating gradient information [32, 33]. Hybrid PSOA's alter the empirical performance of the algorithm by various strategies such as reducing the stochastic terms in the velocity rule [3, 6, 29] or increasing the stochastic terms in the velocity rule [34, 35].

Chapter 3

Diversity in the PSOA

3.1 Introduction

The aim of evolutionary strategies is to improve on the performance of random searches in optimization [36]. This is accomplished by identifying regions containing good designs. It is then assumed that searches in the region of good designs may uncover further improved designs. Various strategies exist to identify and explore regions with a high probability of improved candidate solutions. These strategies distinguish between different evolutionary computation algorithms; the algorithms include genetic programming [37], genetic algorithms [38], evolutionary strategies [39], differential evolution [40] and the particle swarm optimization algorithm (PSOA) [3, 4].

The PSOA was introduced by Kennedy and Eberhart [3] as a gradient free stochastic optimization algorithm. The fundamental principle behind the PSOA is the evolutionary advantages that the sharing of information offers. This is often known as 'collaborative searching'.

The PSOA is quite simple: At first, a swarm of p particles is randomly deployed in an n-dimensional design domain. The particles then update their positions in the design domain over unit time increments using a simple stochastic rule, known as the 'velocity rule'.

The quality of each particle's position at each iteration is then evaluated using the objective or cost function. Each particle's cognitive memory allows it to remember it's own best cost function value, with associated position, over time. Importantly, the social interaction and awareness of the particles allows them to also remember 'fit' or 'good' cost function values the swarm itself found over time. The particles' movement can be over a continuous domain, a discrete domain [6, 41] or a mixed discrete-continuous domain.

The original formulation of Kennedy and Eberhart [3] is repeated here verbatim:

$$\begin{aligned} &vx[\,][\,] = vx[\,][\,] \\ &+2 * \text{rand}() * (pbest[\,][\,] - presentx[\,][\,]) \\ &+2 * \text{rand}() * (pbestx[\,][gbest] - presentx[\,][\,]). \end{aligned}$$

The bracket pair $[\,][\,]$ represents an $n \times p$ matrix, while rand() supposedly represents a scalar uniform random number. The rightmost terms between round brackets $(\cdot)$ in the second and third lines are denoted the cognitive and social components of learning, respectively.

The widespread use of, and the interest in, the PSOA, attests to the brilliance of the idea of Kennedy and Eberhart. However, their notation is not unambiguous: It is unclear whether the random numbers are scalar numbers which simply scale the *magnitude* of the cognitive and social components of learning, or whether the random numbers are vectors that scale *each component* of the cognitive and social components of learning. Ambiguous notation persists in the literature and the *reported* (i.e. notation) implementation is frequently different to the *actual* implementation. Examples of the *magnitude* scaling notation are [41, 42, 43, 44, 45, 46]. Examples of the *component* scaling notation are [3, 12, 13, 25, 35].

While the randomness issue outlined here may seem of minor importance, the implications are quite severe, as will be demonstrated in sections to come.

In fact, it will be demonstrated that for the *magnitude* scaling implementation, the particle trajectories collapse to p n-dimensional line searches. The *component* scaling implementation does not suffer this drawback. Instead, diverse stochastic search trajectories are retained. (In this chapter, the term 'diversity' implies the opposite of the collapse of a trajectory to a line.)

Also shown is that some popular heuristics like maximum velocity limit, position restriction, craziness and high initial velocities are possibly of less importance than originally thought; their greatest contribution is that they prevent the collapse of particle trajectories to lines.

3.2 Notes on the PSOA formulation

The investigation of the PSOA is started by defining the *instantaneous search domain* of a particle, viz. the domain to which the search of particle i at iteration k is restricted. Also introduced is the term *limit behavior*, viz. the limiting behavior of the complete swarm when no improvement in objective function value is experienced by any individual particle (and hence the swarm itself).

From Eqs. (2.2) and (2.3), observe that the instantaneous search domain depends on two independent contributions, namely the deterministic contribution due to the term $(\boldsymbol{x}_k^i + w\boldsymbol{v}_k^i)$, and the stochastic contribution due to the term $(\boldsymbol{\nu}_k^i \in \boldsymbol{\chi}_k^i)$. The situation is depicted in Figure 3.1, where we use the notation $x_k^i(j)$ to indicate the j-th component of vector $\boldsymbol{x}_k^i$.

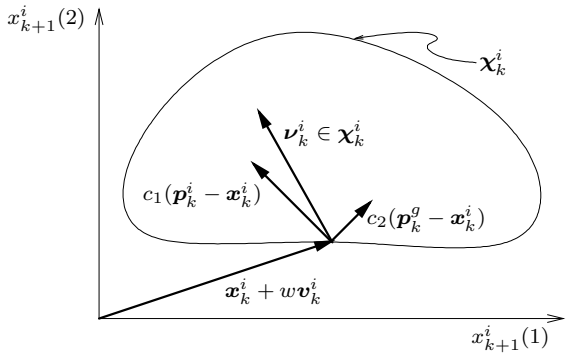


Figure 3.1: The position vector $\boldsymbol{x}_{k+1}^i$, partitioned into a deterministic contribution $(\boldsymbol{x}_k^i + w\boldsymbol{v}_k^i)$ and a stochastic contribution $(\boldsymbol{\nu}_k^i \in \boldsymbol{\chi}_k^i)$.

The instantaneous search domain depends on the stochastic domain $\boldsymbol{\chi}_k^i$, the position vector $\boldsymbol{x}_k^i$, and the inertia term $w\boldsymbol{v}_k^i$. In turn, the stochastic domain $\boldsymbol{\chi}_k^i$ depends on the particle position $\boldsymbol{x}_k^i$, the particle best position $\boldsymbol{p}_k^i$, the best global position $\boldsymbol{p}_k^g$, and the cognitive and social scaling factors c_1 and c_2.

The stochastic domain $\boldsymbol{\chi}_k^i$ is bounded, and has an associated probability distribution, due to the random scaling of finite scalars.

The position rule described by Eq. (2.2) may be viewed as being constructed in 2 separate steps: 1) stochastically generate a point $\boldsymbol{\nu}_k^i$ in the stochastic domain $\boldsymbol{\chi}_k^i$, and 2) deterministically translate this point by $\boldsymbol{x}_k^i + w\boldsymbol{v}_k^i$.

Let us now proceed with an analysis of the two different formulations of the PSOA that have been used in the literature.

3.3 Implementation subtleties: Formulation 1 (PSOAF1)

For the first formulation of the PSOA, which is denoted PSOAF1 here, the stochastic vector $\boldsymbol{\nu}_k^i$ is given by

$$\boldsymbol{\nu}_k^i = c_1 r_{1k}^i(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i) + c_2 r_{2k}^i(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i), \tag{3.1}$$

where r_{1k}^i and r_{2k}^i represent two uniform real random scalar numbers between 0 and 1. r_{1k}^i and r_{2k}^i are updated at every iteration k, and for each particle i in the swarm. Hence r_{1k}^i and r_{2k}^i simply scale the magnitudes of the cognitive and social vectors $c_1(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $c_2(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$.

For the sake of clarity, PSOAF1 is also described by the following pseudo code fragment:

```
for I = 1 to number of particles do
  R1 = uniform random number
  R2 = uniform random number
  for J = 1 to number of dimensions do
    V[I][J]=w*V[I][J]
           +C1*R1*(P[I][J]-X[I][J])
           +C2*R2*[G[I][J]-X[I][J])
  enddo
  X[I][J] = X[I][J]+V[I][J]
enddo
```

Let us now study the stochastic contribution $\boldsymbol{\nu}_k^i$ to the composition of the instantaneous search domain given by Eq. (3.1). The cognitive vector $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and the social vector $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ consist of the directions and distances from the current position $\boldsymbol{x}_k^i$ to the best particle position $\boldsymbol{p}_k^i$, and the best global position $\boldsymbol{p}_k^g$; the cognitive and social vectors can be anything from normal to parallel w.r.t. each other.

When the cognitive vector $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and the social vector $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ are not parallel, Eq. (3.1) may be interpreted as the vector equation of a bounded plane $\boldsymbol{\mathcal{P}}_k^i$ in n-dimensional space. The plane is bounded, since the length of the cognitive and social vectors are scaled independently by the finite scalars $c_1 r_{1k}^i$ and $c_2 r_{2k}^i$. The bounded plane $\boldsymbol{\mathcal{P}}_k^i$ is then translated in n-dimensional space by the addition of $\boldsymbol{x}_k^i$ and $w\boldsymbol{v}_k^i$, as depicted in Figure 3.2.

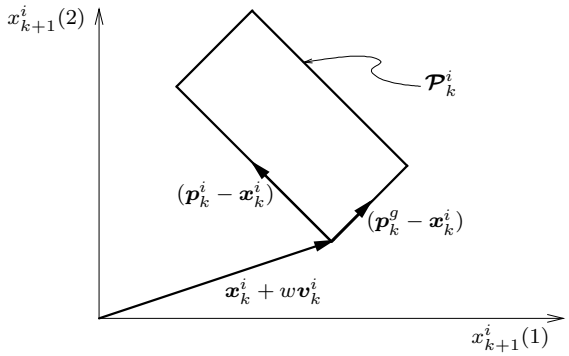


Figure 3.2: Partitioning the position vector $\boldsymbol{x}^i_{k+1}$ into a deterministic contribution $(\boldsymbol{x}^i_k + w\boldsymbol{v}^i_k)$, and a stochastic contribution $(\boldsymbol{\nu}^i_k \in \boldsymbol{\mathcal{P}}^i_k)$, for $c_1 = c_2 = 2$.

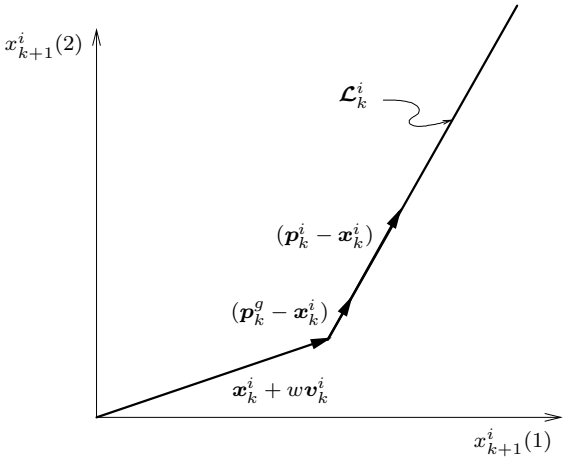


Figure 3.3: Partitioning the position vector $\boldsymbol{x}^i_{k+1}$ into a deterministic contribution $(\boldsymbol{x}^i_k + w\boldsymbol{v}^i_k)$, and a stochastic contribution $(\boldsymbol{\nu}^i_k \in \boldsymbol{\mathcal{L}}^i_k)$, for $c_1 = c_2 = 2$.

If the cognitive vector $(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and the social vector $(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ are parallel, Eq. (3.1) may be interpreted as the vector equation of a bounded line $\boldsymbol{\mathcal{L}}^i_k$ in n-dimensional space. Again, the line is translated in the n-dimensional space by the addition of $\boldsymbol{x}^i_k$ and $w\boldsymbol{v}^i_k$, as depicted in Figure 3.3.

3.3.1 Investigation of the limit behavior of PSOAF1

Let us study the 3-dimensional dynamic limiting behavior of a particle. For the sake of simplicity, it is assumed that the best particle position $\boldsymbol{p}^i_k$, and the best global position $\boldsymbol{p}^g_k$ remain unchanged for 25 consecutive iterations.

Randomly generate the best particle best position $\boldsymbol{p}^i_0$, the best global position $\boldsymbol{p}^g_0$, and the initial position vector $\boldsymbol{x}^i_0$ between -2 and 2 over each dimension. The initial velocity vector $\boldsymbol{v}^i_0$ is assumed to equal $\mathbf{0}$. No velocity or position restriction is implemented. The study is conducted with $c_1 = c_2 = 2$ and $w = 0.8$, being values which have frequently been used by others in combination with constant inertia [8, 47, 45].

The sequence of 25 consecutive iterations, for a single particle, is depicted in Figure 3.4. The figure

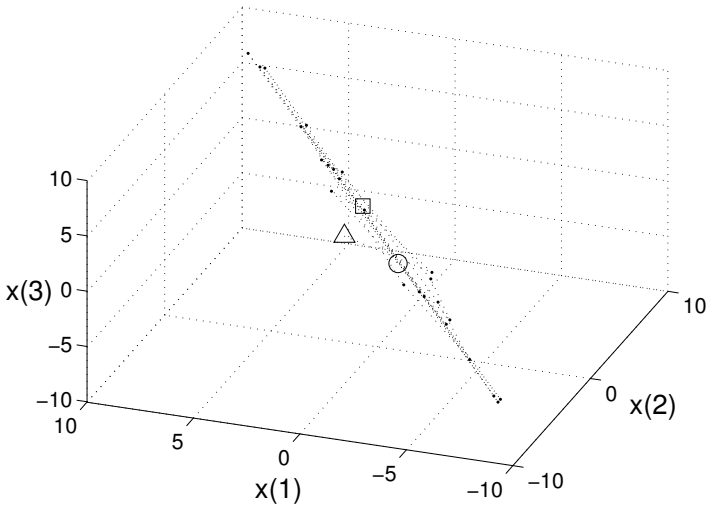


Figure 3.4: PSOAF1: Position vectors $\boldsymbol{x}_{k+1}^i$ generated for 25 consecutive iterations. The best particle position $\boldsymbol{p}_k^i$ and the best global position $\boldsymbol{p}_k^g$ are kept constant. Representing $\boldsymbol{x}_0^i$ is $\triangle$, $\boldsymbol{p}_k^i$ is $\bigcirc$, and $\boldsymbol{p}_k^g$ is $\square$.

illustrates that the particle trajectory collapse to a straight line as the iteration counter increases.

Let us consider this (undesirable) phenomenon in more detail: The angle $\bar{\theta}$ between the cognitive vector $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and social vector $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ may be determined using

$$\bar{\theta} = \cos^{-1}\Big(\frac{|(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i) \cdot (\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)|}{||(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)||\ ||(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)||}\Big). \tag{3.2}$$

If $\bar{\theta} = 0°$, the vectors $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ are parallel, when $\bar{\theta} = 90°$, the vectors $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ are perpendicular.

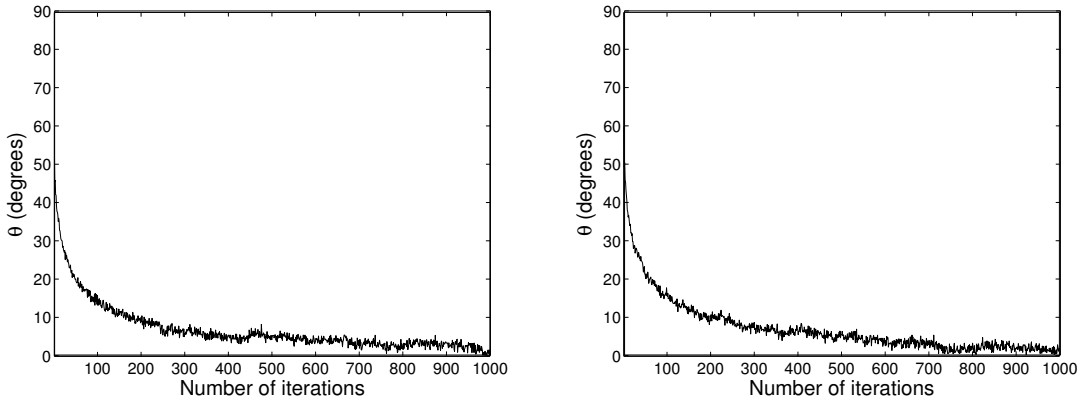
The experiment performed in constructing Figure 3.4 is repeated. However, this time the algorithm runs for 1000 iterations, and the average angle θ is determined from 100 independent runs. The study is conducted for $n = 3$ and $n = 30$, to determine any dependency on dimensionality.

The results for a dimensionality of $n = 3$ and $n = 30$ are respectively depicted in Figures 3.5(a) and 3.5(b). The figures illustrate that as the iterations progress, the average angle θ between the cognitive vector $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and the social vector $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ sharply decreases for both $n = 3$ and $n = 30$. The initial decrease is rapid; after some 150 iterations, the average angle θ is less than $10°$.

The foregoing implies that for a given particle, the position vector $\boldsymbol{x}_{k+1}^i$ is updated in a 'long narrow bounded plane'. This is reminiscent of multiple line searches in a largely restricted domain.

It should be noted that for low values of the inertia w, complete collapse to line searches ($\theta = 0°$) is demonstrated within 1000 iterations. Furthermore: for a given particle, it is impossible to escape from this long narrow bounded plane until *another* particle's best position $\boldsymbol{p}_k^g$ is updated, since each particle's best positions can only be updated in its own long narrow bounded plane.

For a swarm of p particles, this implies that if the best global position vector $\boldsymbol{p}_k^g$ is not updated, the



(a) (b)

Figure 3.5: PSOAF1: The average angle θ between $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ versus iteration number, for a dimensionality of a) $n = 3$, and b) $n = 30$. For the 1000 iterations, the best particle position $\boldsymbol{p}_k^i$ and the best global position $\boldsymbol{p}_k^g$ are stationary.

swarm conducts p line searches in n-dimensional space for possibly a high number of consecutive iterations. Worse: the line searches all *intersect in the same point* in the design domain, being the best global position $\boldsymbol{p}_k^g$. Again: if a particle updates its own best position vector $\boldsymbol{p}_k^i$, then it remains searching in a line. Only when the global best position vector is updated do all other particles again (briefly) search in planes.

For obvious reasons, the foregoing may have severe implications on algorithm performance.

3.4 Implementation subtleties: Formulation 2 (PSOAF2)

In the alternative implementation of the velocity rule, each component of $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ is scaled independently. The vector directions are then no longer preserved.

In order to scale each component independently, the scalar random numbers r_{1k}^i and r_{2k}^i in the stochastic vector in Eq. (3.1) are replaced by two random diagonal matrices $\boldsymbol{R}_{1k}^i$ and $\boldsymbol{R}_{2k}^i$ as follows:

$$\boldsymbol{\nu}_k^i = c_1 \boldsymbol{R}_{1k}^i (\boldsymbol{p}_k^i - \boldsymbol{x}_k^i) + c_2 \boldsymbol{R}_{2k}^i (\boldsymbol{p}_k^g - \boldsymbol{x}_k^i). \tag{3.3}$$

The $\boldsymbol{R}_{mk}^i$ random diagonal matrices are explicitly given as

$$\boldsymbol{R}_{mk}^i = \begin{bmatrix} \rho_{11k}^i & 0 & \cdots & 0 \\ 0 & \rho_{22k}^i & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & \rho_{nnk}^i \end{bmatrix}, \quad m = 1, 2, \tag{3.4}$$

with $0 < \rho_{jjk}^i < 1, \quad j = 1, \ldots, n$ a uniform random number [12, 48].

The above form is used since it has previously been introduced by others. However, for future use, the following equivalent form is proposed

$$\boldsymbol{\nu}_k^i = c_1 \boldsymbol{r}_{1k}^i \circ (\boldsymbol{p}_k^i - \boldsymbol{x}_k^i) + c_2 \boldsymbol{r}_{2k}^i \circ (\boldsymbol{p}_k^g - \boldsymbol{x}_k^i), \tag{3.5}$$

where the $\circ$ operator indicates element by element multiplication. Hence the random vectors $\boldsymbol{r}_{mk}^i$ are given as

$$\boldsymbol{r}_{mk}^i = \left(\rho_{1k}^i,\ \rho_{2k}^i,\ \cdots,\ \rho_{nk}^i\right), \quad m = 1, 2. \tag{3.6}$$

Eq. (3.5) is no longer a vector representation of a bounded plane $\boldsymbol{\mathcal{P}}_k^i$, since the non-zero components of the cognitive vector $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and the social vector $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ are independently scaled. As a result, possible stochastic updates can occur in n-dimensional space.

The pseudo code for PSOAF2 is

```
for I = 1 to number of particles do
  for J=1 to number of dimensions do
    R1=uniform random number
    R2=uniform random number
    V[I][J]=w*V[I][J]
           +C1*R1*(P[I][J]-X[I][J])
           +C2*R2*[G[I][J]-X[I][J])
  enddo
  X[I][J] = X[I][J]+V[I][J]
enddo
```

The difference with the pseudo code given for PSOAF1 is subtle; the only difference is that the random numbers are updated inside the *for-loop* that runs over the design dimensions $(1, \ldots, n)$. However, the implications are severe.

3.4.1 Investigation of the limit behavior of PSOAF2

As with PSOAF1, the 3-dimensional trajectory of a particle when the best particle position $\boldsymbol{p}_k^i$, and the best global position $\boldsymbol{p}_k^g$ are kept constant for 25 consecutive iterations is studied. In performing the experiment, the same settings as for PSOAF1 are used, except for using a lower value of inertia, namely $w = 0.6$, since PSOAF2 is unstable at high values of w.

Figure 3.6 suggests that the particle trajectories now do not collapse to line searches; instead, 'diversity' is retained. To verify this, the average angle θ between the cognitive vector $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and social vector $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ is calculated. A similar approach as for PSOAF1 is used; again the study is conducted for $n = 3$ and $n = 30$ to determine the sensitivity to problem dimensionality.

As opposed to PSOAF1, the trajectories of PSOAF2 do not collapse to line searches. As depicted in Figure 3.7, the angle θ is roughly 47° and 37° for $n = 3$ and $n = 30$ respectively, over the entire range of iterations studied.

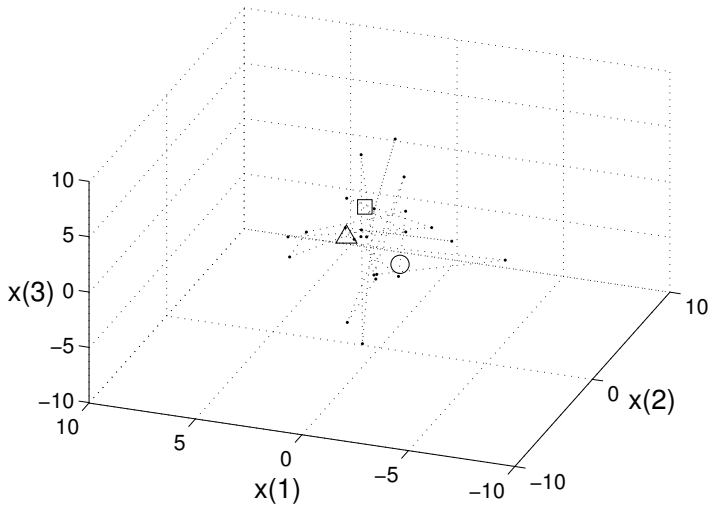
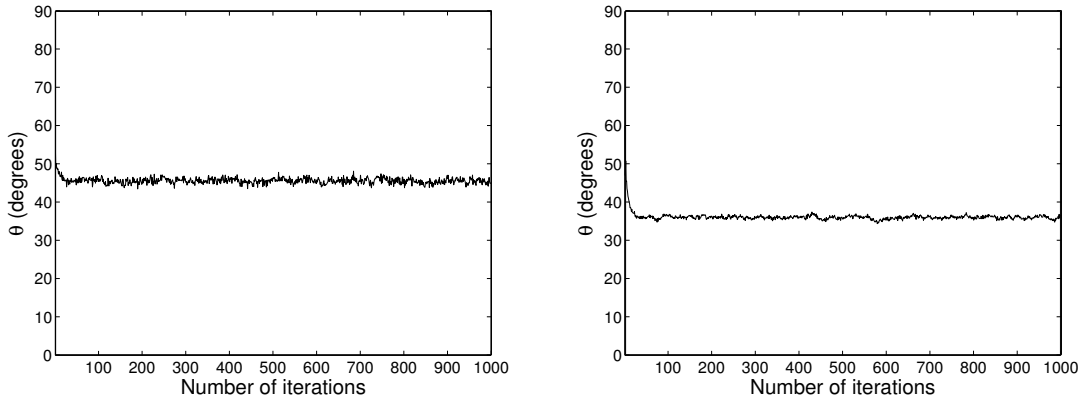


Figure 3.6: PSOAF2: Position vectors $\boldsymbol{x}_{k+1}^i$ generated over 25 iterations without updating the particle best position vector $\boldsymbol{p}_k^i$ and the global best position vector $\boldsymbol{p}_k^g$. No restriction is imposed on the velocity vector.



(a) (b)

Figure 3.7: PSOAF2: The average angle θ between $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ versus iteration number, for a dimensionality of a) $n = 3$, and b) $n = 30$. For the 1000 iterations, the best particle position $\boldsymbol{p}_k^i$ and the best global position $\boldsymbol{p}_k^g$ are stationary.

3.5 Numerical experiments

Numerical experiments to evaluate the difference in performance between PSOAF1 and PSOAF2 are now conducted.

The aim is not an exhaustive determination of optimal algorithmic parameters, but to illustrate the effects of the different velocity updating rules in the two formulations. Hence a basic PSOA is implemented, without additional heuristics such as maximum velocity restriction, position restriction, craziness or dynamic inertia reduction or increase. (These are however discussed in sections to come.)

Initial velocities are set equal to $\mathbf{0}$. A simple synchronous updating scheme [31] is used. Real variables are implemented using *double-precision floating-point* arithmetic.

For this study the algorithm parameters are $c_1 = c_2 = 2$, the swarm size is $p = 20$ particles and the computations are performed for various constant inertia factors w. Each run is terminated after 10000 iterations, and the reported results are average values obtained from 100 independent runs.

In this study the following five test functions are used:
i) The Rosenbrock function (unimodal, f_0):

$$f_0(\boldsymbol{x}) = \sum_{i=1}^{\frac{n}{2}} \Big(100\big(x_{2i} - x_{2i-1}^2\big)^2 + \big(1 - x_{2i-1}\big)^2\Big).$$

ii) The Quadric function (unimodal, f_1):

$$f_1(\boldsymbol{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2.$$

iii) The Ackley function (multimodal, f_2):

$$\begin{aligned} f_2(\boldsymbol{x}) = \; & -20 \exp\left(-0.2\sqrt{\tfrac{1}{n}\textstyle\sum_{i=1}^{n} x_i^2} \right) \\ & - \exp\left(\tfrac{1}{n}\textstyle\sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e. \end{aligned}$$

iv) The generalized Rastrigin function (multimodal, f_3):

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

v) Finally, the generalized Griewank function (multimodal, f_4):

$$f_4(\boldsymbol{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1.$$

The parameters used in the study are given in Table 3.1. The *domain* column represents the range of each dimension of the design variables; the test function domains are symmetrical about 0.0 in all dimensions.

Table 3.1: Test function parameters.

Function	n	domain
f_0	30	$\pm$ 2.048
f_1	30	$\pm$ 100.0
f_2	30	$\pm$ 30.0
f_3	30	$\pm$ 5.12
f_4	30	$\pm$ 600.0

3.6 Discussion of numerical results

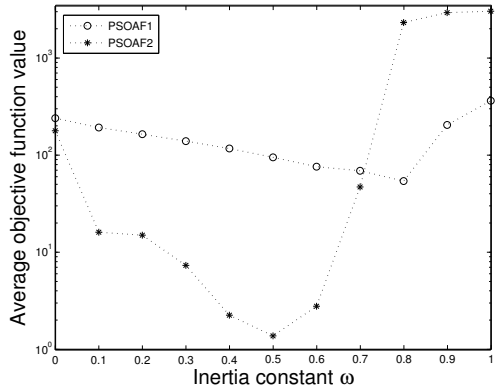


Figure 3.8: Average function value after 2×10^5 function evaluations (10000 iterations) over 100 runs on the Rosenbrock test function (f_0).

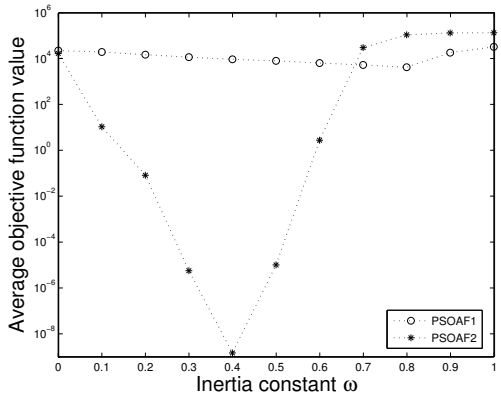


Figure 3.9: Average function value after 2×10^5 function evaluations (10000 iterations) over 100 runs on the Quadric test function (f_1).

Figures 3.8, 3.9, 3.10, 3.11 and 3.12 depict the average objective function values after 2×10^5 function evaluations (10000 iterations) for the 5 test functions under consideration. A summary

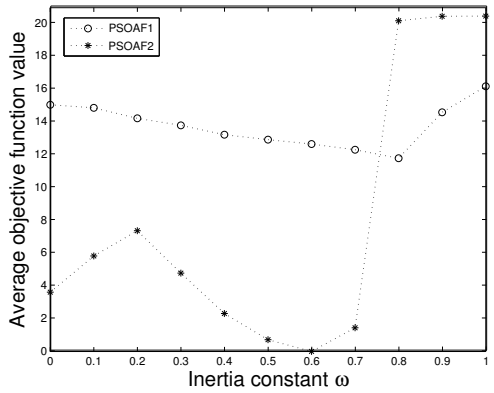


Figure 3.10: Average function value after 2×10^5 function evaluations (10000 iterations) over 100 runs on the Ackley test function (f_2).

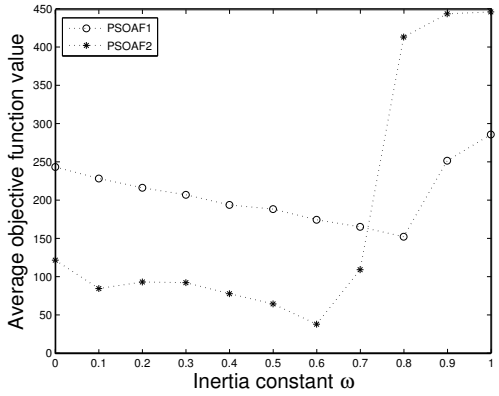


Figure 3.11: Average function value after 2×10^5 function evaluations (10000 iterations) over 100 runs on the Rastrigin test function (f_3).

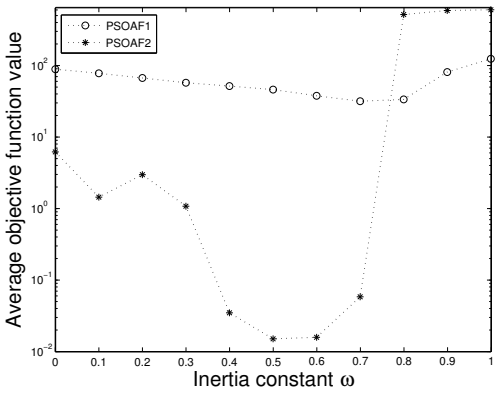


Figure 3.12: Average function value after 2×10^5 function evaluations (10000 iterations) over 100 runs on the Griewank test function (f_4).

Table 3.2: Constant inertia factor at which the best average objective function value is obtained.

	PSOAF1		PSOAF2	
	f_{ave}^{best}	w	f_{ave}^{best}	w
f_0	54.845	0.8	1.393	0.5
f_1	4395.919	0.8	1.5×10^{-9}	0.4
f_2	11.804	0.8	9×10^{-15}	0.6
f_3	152.988	0.8	38.425	0.6
f_4	32.078	0.7	1.5×10^{-2}	0.5

of these results may be found in Table 3.2, which tabulates the best function values obtained with PSOAF1 and PSOAF2 after the 10000 iterations, together with the corresponding inertia factor.

The figures and the table clearly illustrate the vast difference in performance between PSOAF1 and PSOAF2, with PSOAF2 superior to PSOAF1 for all the problems considered.

The performance of PSOAF1 improves as the inertia factor w is increased, up to an inertia factor of $w = 0.8$. For higher inertia factors ($w \geq 0.9$) there is a rapid decline in the performance of the algorithm, due to instability. (Instability occurs due to excessive particle velocities, since there is no limit on the maximum value of velocity.) For low values of inertia, performance is hampered, since the collapse of a particle trajectory to a line search occurs earlier than at high values of w.

PSOAF2 performs well for $w \leq 0.6$, but becomes unstable for $w \geq 0.8$. For the test set and conditions used, the optimal performance for PSOAF2 is obtained with $0.4 \leq w \leq 0.6$. For PSOAF1, higher values of w are suitable.

The average convergence history for Ackley's test function, for PSOAF1 and PSOAF2, is depicted in Figures 3.13(a) and 3.13(b) respectively. The two figures are drawn on the same scale; PSOAF2 is clearly superior to PSOAF1, both in terms of average convergence rate and in terms of the quality of the solution found (for reasonable values of w).

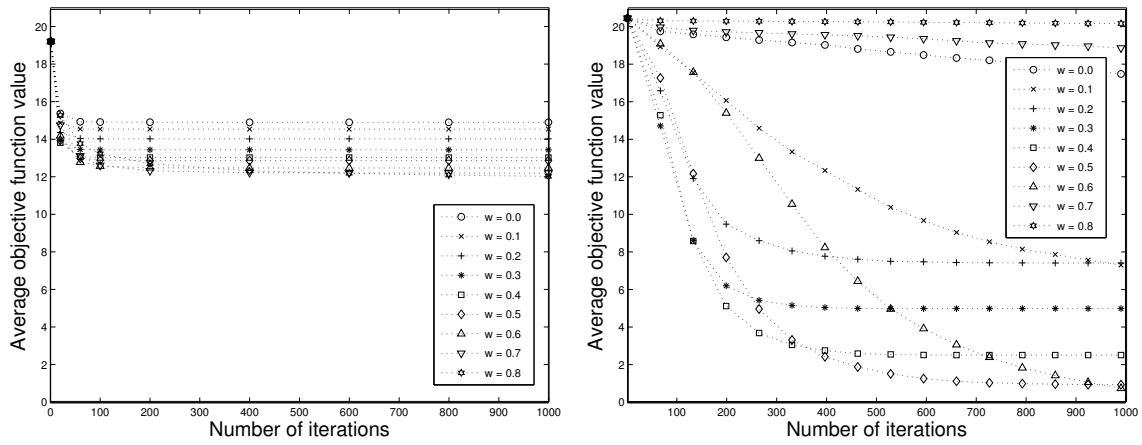
3.7 Notes on some heuristics of the PSOA

An explanation of the effects of some popular heuristics that are widely considered to improve the performance of the PSOA follows. (For obvious reasons, it should now be clear that the gain in performance for PSOAF1 can be expected to be far higher than for PSOAF2. Indeed, most 'successful' heuristics merely prevent or delay the collapse of particle trajectories to lines in PSOAF1.)

3.7.1 Local best neighborhood

Local neighborhoods [49] are used in an attempt to introduce independent social groups into the swarm; information between these groups is then propagated back into the swarm in some structured fashion.

Local neighborhoods are beneficial since their introduction results in clusters of line searches



(a) (b)

Figure 3.13: Average convergence history for Ackley's test function, for a) PSOAF1, and b) PSOAF2.

which intersect in the locally best point of each neighborhood, instead of only intersecting in the best global point of the complete swarm. Hence local neighborhoods increase the diversity of the algorithm. In addition, the communication of this information between neighborhoods results in an instantaneous increase in diversity.

3.7.2 Non-zero initial velocities

The introduction of non-zero initial velocities delays line searches in PSOAF1, since the term $w\boldsymbol{v}_k^i$ translates the instantaneous search domains (planes) in the design domain, which increases diversity. Note however that this only helps during initial iterations; these contributions damp out over time.

3.7.3 Maximum velocity restriction

Maximum allowed velocity [4] is a well known heuristic, frequently used in the literature. It is used to stabilize the algorithm.

There are two fundamentally different ways to implement maximum velocity restriction. Firstly, the restriction can be placed on each component of the vector $\boldsymbol{v}_{k+1}^i$, as shown in Figure 3.14(a). Alternatively, the length of $\boldsymbol{v}_{k+1}^i$ can be restricted, as shown in Figure 3.14(b).

The advantage of restricting each component is the ease of implementation. When restricting the components of the velocity vector, the magnitude of the velocity vector depends on the magnitudes of the velocity components. The maximum length that the velocity vector can obtain is

$$\boldsymbol{v}_{k+1}^{max} = \sqrt{\sum_{j=1}^{n} \left(v^{max}(j)\right)^2},$$

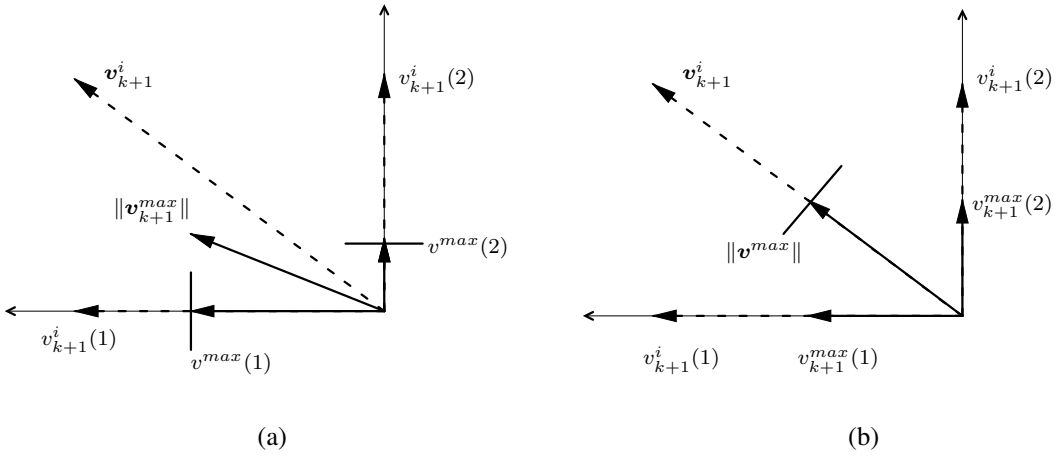


Figure 3.14: The velocity restriction implemented by a) restricting the component values of the velocity vector a) restricting the magnitude of the velocity vector.

when all the vector components are restricted. In addition, by restricting each component of the velocity vector, (Figure 3.14(a)), the direction of the velocity vector is not preserved. The result is that the velocity directions are not limited to the line $\mathcal{L}^i_k$.

On the other hand, restriction of the velocity magnitude (Figure 3.14(b)), does not alter the velocity direction between consecutive iterations.

In summary, this implies that the results of restriction of the maximum velocity component on the PSOA are two-fold. Firstly, this stabilizes the algorithm, by limiting the maximum component value in each dimension. Secondly, this increases diversity through the generation of position vectors, which by their very nature are not confined to the line $\mathcal{L}^i_k$.

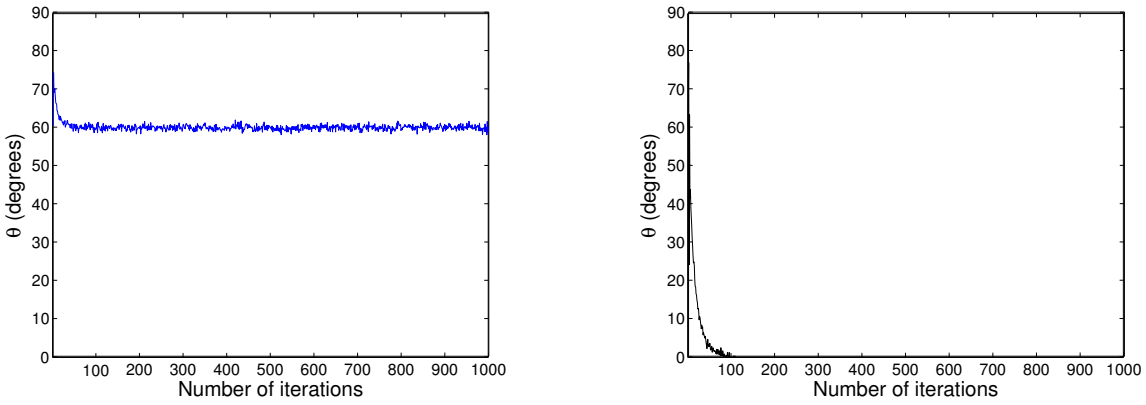
To illustrate the foregoing, the average angle θ between the vectors $(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ is again studied. As before, randomly generate the best particle position vector $\boldsymbol{p}^i_0$, the best global position vector $\boldsymbol{p}^g_0$, initial positions $\boldsymbol{x}^i_0$, and the initial velocities $\boldsymbol{v}^i_0$ with each vector component between -2 and 2.

Velocity restriction on both the component and the magnitude is implemented. Again, $c_1 = c_2 = 2$ and $w = 0.8$. The algorithm is again terminated after 1000 consecutive iterations, with θ averaged over 100 runs. As before, the best particle position $\boldsymbol{p}^i_k$ and the best global position $\boldsymbol{p}^g_k$ are assumed stationary. The study is conducted for $n = 30$.

When applying restriction to the *components* of velocity, Figure 3.15(a) depicts that the average angle θ between the cognitive vectors $(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and the and social vectors $(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ is some 60°; the trajectories do not collapse.

However, when the velocity *magnitude* is restricted, the angle θ quickly collapses to 0° (Figure 3.15(b)). (Here, a restriction of $||\boldsymbol{v}^i_{k+1}|| \leq 4$ is used.)

Depicted in Figure 3.16 is the average function value after 2×10^5 function evaluations or 10000 iterations for the Griewank test function f_4. As expected, velocity restriction significantly improves algorithm performance for high values of inertia $(w \geq 0.8)$, since instability at high inertia factors



(a) (b)

Figure 3.15: Velocity restriction on a) the components, and b) the magnitude of velocity. Depicted is the average angle θ between $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ versus iteration number, for a dimensionality of $n = 30$. For the 1000 iterations, the best particle position $\boldsymbol{p}_k^i$ and the best global position $\boldsymbol{p}_k^g$ are stationary.

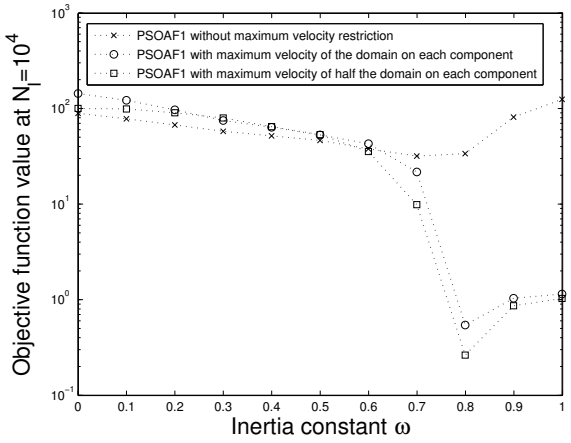


Figure 3.16: Average function value for the Griewank test function (f_4) after 1000 iterations, averaged over 100 runs.

is prevented.

Clearly, a restriction on each vector component of maximum velocity avoids the collapse of search trajectories to lines. This observation was also made by Carlisle and Dozier [31], who demonstrated that lower values of maximum velocity improves algorithm performance. The mechanism for this is now easily understood: lower values of maximum velocity trigger the introduction of diversity more often than higher values of maximum velocity. (Although low values of course decrease the instantaneous search domain.)

3.7.4 Minimum velocity restriction

Velocity may also be restricted on the lower end of the scale, as a mechanism to prevent premature convergence. Again, either the components or the magnitude of velocity may be restricted.

However, firstly demonstrated is the remarkable sensitivity of the algorithm to arithmetic precision, which may be viewed as a special case of minimum velocity restriction: PSOAF1 is implemented using both *single-precision floating-point* arithmetic and *double-precision floating-point* arithmetic. Otherwise, the two implementations are *identical*. Also provided is a minimum velocity limit on each component of the velocity vector of the code implemented with *double-precision floating-point* arithmetic. Using pseudo-code, it is done as follows:

```
for I = 1 to number of particles do
  for J = 1 to number of dimensions do
    calculate V[I][J]
    if abs(V[I][J]) less than Vmin then
      V[I][J] = sign(V[I][J]) * Vmin
    endif
  enddo
enddo
```

Values of $V_{min} = 10^{-2}$ and $V_{min} = 10^{-3}$ are used, again for runs consisting of 10000 iterations, and reported are the average values for 100 runs. Results for only the Griewank test function (f_4) are presented, but the results for the other test functions are similar. The same initial positions, velocities and random number sequences are used in the three different implementations.

Figure 3.17 depicts the average objective function value. The *single-precision floating-point* implementation is slightly superior to the *double-precision floating-point* implementation, simply because the lower precision results in angles which are not quite zero. For the *double-precision floating-point* implementation, minimum velocity increases the performance of the algorithm dramatically. (Although not shown in the figure, values of V_{min} equal to the precision attainable in single precision, renders the two implementations almost identical.)

As with maximum velocity restriction, the implications of minimum velocity restriction are twofold: Firstly, premature convergence may be overcome, and the collapse to the line $\mathcal{L}_k^i$ may be delayed.

(The very low values above are for illustrative purposes only, in practice higher values may be desirable.)

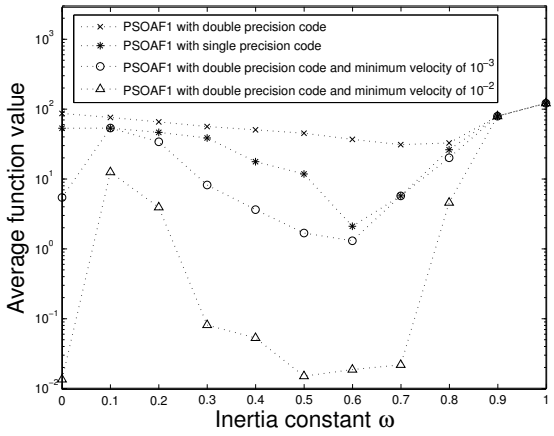


Figure 3.17: Average function value after 2×10^5 function evaluations (10000 iterations) over 100 runs on the Griewank test function (f_4).

3.7.5 Position restriction

There are various ways to implement position restriction, as Robinson and Rahmat-Samii [30] for example point out in their study. Position restriction may be seen as an alternative implementation of velocity restriction, albeit more complicated, since every vector $\boldsymbol{x}_{k+1}^i$ may equivalently be represented as the vector $\boldsymbol{x}_k^i + \boldsymbol{v}_{k+1}^i$.

While the implementation of position restriction has to a large extent been neglected in the literature, note that the desirability and implementation should be judged against the background of collapsed line searches.

3.7.6 Craziness

Craziness [3] is a heuristic which randomly places particles in the design domain. Normally performed at low probability, it obviously increases diversity, and prevents the collapse to line searches. (At high probability, craziness results in (ineffective) pure random search.) The craziness operator is somewhat reminiscent of mutation in the genetic algorithm (GA).

Continuous monitoring of θ may effectively be combined with craziness to prevent premature convergence of the PSOA. In addition, 'small perturbations', reminiscent of creep mutation in the GA, may be more effective than the equivalent of jump mutation in the GA.

A number of authors have previously reported that craziness is ineffective. However, it is a simple matter to demonstrate that craziness is indeed effective for PSOAF1, and in particular for problems of high dimensionality.

Consider the very simple unconstrained n-dimensional (convex) quadratic test function

$$f_5(\boldsymbol{x}) = \sum (x_i - i)^2, \quad i = 1, 2, 3, \cdots, n,$$

subject to the bounds

$$-100 \leq x_i \leq 100, \quad i = 1, 2, 3, \cdots, n.$$

Table 3.3: Effect of probability of craziness P_{cr} on $f_5(\boldsymbol{x})$

n	P_{cr}	PSOAF1	PSOAF2
10	0.00	1.158E+02	6.310E-32
	0.05	1.719E-05	1.650E-09
	0.10	1.165E-02	1.434E-03
30	0.00	1.627E+04	4.469E-28
	0.05	2.787E-02	5.597E-04
	0.10	2.528E+00	1.073E+01

Tabulated numerical results are presented in Table 3.3; the optimum solution $f^* = 0.0$. Again, PSOAF1 and PSOAF2 are used without any heuristic whatsoever. Both algorithms are terminated after 10000 iterations, and the reported results are averaged over 100 independent runs. Again $p = 20$ is used. For PSOAF1, select $w = 0.7$, for PSOAF2 select $w = 0.5$. Many implementations of craziness are possible; in this case, P_{cr} indicates the probability of a given particle to become crazy.

Table 3.3 illustrates that craziness is highly beneficial to PSOAF1, while it impairs the performance of PSOAF2. The reasons are obvious: for PSOAF1, craziness increases diversity, and prevents the collapse of trajectories to line searches. (Note that this does not imply that craziness is indispensable for PSOAF1; any of the other heuristics which increases diversity will of course improve the performance of PSOAF1.) The impairment of PSOAF2 is also easily explained: this algorithm is diverse 'enough'. (In the limit, an increase in craziness of course results in pure random search in both algorithms.)

3.7.7 Increasing social awareness

Again, in this approach [34, 35], a number of different implementations are possible. In a simple implementation, the best position of each particle in the swarm is considered when constructing the velocity rule.

It is now clear why this works: diversity is increased; the collapse of trajectories to line searches is delayed.

3.7.8 Inertia factor

Zheng et. al. [16] proposed to increase the inertia w as the iterations progress. They increased w from 0.4 to 0.9 over the prescribed number of iterations. Clearly, this assists in prolonging the time before the trajectories collapse to line searches, due to translation of the instantaneous search domains (planes) in D.

Very high inertia factors (w larger than unity) may of course also be used to prolong diverse searches. The amount $1 - w$ may be viewed as the equivalent damping term in a spring-mass-damper system [19]. Hence $w > 1.0$ implies negative damping, or the introduction of energy into

the system as the iterations progress. However, this is only recommended in terminal phases of the PSOA; during initial phases, this may result in pure random search. In addition, $w > 1.0$ should be used in combination with heuristics like position restriction.

3.7.9 Using a single random number

Some authors have attempted to replace the random numbers r^i_{1k} and r^i_{2k} in Eq. (3.1) by a single random number. While this is acceptable, the implications should be understood.

This issue is addressed by asking the following question: *What is the range and the distribution of the sum of two random numbers* r_1 *and* r_2 *that are both uniformly distributed between* 0 *and* 1*?* (All three authors initially guessed that the range would lie between 0 and 2 and that the probability distribution would remain uniform.)

However, a simple Monte Carlo simulation based on 10^6 instances reveals that the solution is the bi-linear probability distribution. The reason is of course obvious [50]: In order to generate a number close to zero both r_1 and r_2 need to be close to zero. In contrast, to generate a number close to 1, a large number of possibilities exist e.g. $(0.1+0.9)$, $(0.7+0.3)$, $(0.6+0.4)$, $(0.5+0.5)$, etc.

The sum of two random variables $r^i_{3k} = r^i_{1k} + r^i_{2k}$ should not be considered a uniform random number with a range between 0 and 2, e.g. see [16]. Instead it should be chosen from a bi-linear distribution.

3.8 On tuning of PSOA parameters (finding universal optimal parameter values)

In the foregoing, it is demonstrated that it is essential to distinguish between implementations of randomness into the PSOA when reporting results. 'Optimal settings' for the parameters of PSOAF1 are simply not optimal for PSOAF2.

However, the same argument applies for many other implementational issues. Without further elaboration, a check list of implementation issues is given that should in my opinion be reported when 'optimal setting' for the PSOA are published, since the optimality of the parameters strongly depends on the implementation.

The implementational issues to be reported are as follows:

1. General:
   (a) the selected velocity rule (using unambiguous notation),
   (b) the updating strategy (whether synchronous or asynchronous), and
   (c) the stopping criteria used.
2. The heuristics used (with a detailed description thereof):
   (a) minimum velocity restriction (and arithmetic precision),

(b) maximum velocity restriction,

(c) the implemented inertia strategy,

(d) position restriction,

(e) etc.

3. The parameters used:

(a) the population size p,

(b) the initial inertia and velocity values, and

(c) the social and cognitive scaling factors c_1 and c_2.

Finally, optimal parameters should of course be understood within the context of the 'no free lunch algorithms [51, 52].

3.9 Closure

Implementation subtleties due to ambiguous notation that resulted in two distinctly different implementations of the PSOA have been pointed out. Discerning between these two implementations is of crucial importance.

While the behavior of the two different implementations is markedly different, they only differ in the formulation of the velocity updating rule. In fact, the differences are merely due to subtle differences in the introduction of randomness into the algorithm.

A scrutiny of PSOA codes has revealed that the reported implementation is often different to the actual computer implementation. Against the background of this chapter, it is now also possible to identify papers for which this discrepancy holds.

As shown for the first implementation, the particle trajectories collapse to ineffective line searches. The second implementation does not suffer this drawback. Instead, diverse stochastic search trajectories are retained.

Also shown is that some popular heuristics like maximum velocity limit, position restriction, craziness and high initial velocities are not of overwhelming importance in their own right; they merely prevent collapse of the particle trajectories to lines.

Finally, it is emphasized that the determination of optimal values for parameters like inertia, velocity limit, etc. has to be performed within the context of the formulation used. A list of parameters and implementational issues that should be included when reports on PSOA parameter settings are written is proposed.

Chapter 4

Objectivity of the PSOA

4.1 Introduction

In science, most physical phenomena are invariant. It is fundamental that mathematical representation of these phenomena reflects this invariance. This fundamental requirement is known as objectivity, frame-indifference or observer independence, and is well known in classical mechanics [53]. For objectivity, the description of some quantity has to be invariant under pure translations, as well as pure rotations.

Objectivity or observer independence is also highly desirable (almost essential) in optimization procedures, to reflect the invariance of the physical processes that are optimized. Robust optimization procedures and algorithms should definitely be invariant.

In classical gradient based optimization, the gradient vector (or some conjugate direction to the gradient), indicates some direction of improvement, even if this direction is not optimal. This accounts for the reference frame; classical optimization is (usually) frame invariant.

In modern (stochastic) optimization procedures, the requirement of observer independence is equally essential. These algorithms include genetic programming [37], genetic algorithms [38], evolutionary strategies [39], differential evolution [40] and the particle swarm optimization algorithm (PSOA) [3, 4].

For the genetic algorithm (GA), Salomon [54, 55] demonstrated the lack of rotational invariance of the algorithm. He showed that the GA's performance at low mutation rates is significantly influenced by the frame of reference used to pose a problem.

The PSOA was introduced by Kennedy and Eberhart [3] as a gradient free stochastic optimization algorithm. The fundamental principle behind the PSOA is the evolutionary advantages that the sharing of information offers. This is often known as 'collaborative searching'.

The PSOA is quite simple: At first, a swarm of p particles is randomly deployed in an n-dimensional design domain. The particles then update their positions in the design domain over unit time increments using a simple stochastic rule, known as the 'velocity rule'.

The quality of each particle's position at each iteration is then evaluated using the objective or cost function. Each particle's cognitive memory allows it to remember it's own best cost function

value, with associated position, over time. Importantly, the social interaction and awareness of the particles allow them to also remember the 'best' cost function value the swarm itself found over time.

In Chapter 3 it was shown that implementation subtleties due to ambiguous notation have resulted in two distinctly different implementations of the PSOA. While this does not repute negatively on the ingenuity of the idea of Kennedy and Eberhart, discerning between these two implementations is of crucial importance. The behavior of the respective implementations is markedly different, although they only differ in the formulation of the velocity updating rule. In fact, the differences are merely due to subtle differences in the introduction of randomness into the algorithm.

In this chapter, the objectivity of the PSOA is investigated. It is shown that the first formulation PSOAF1 is objective, combined with the *disadvantage* that the particle trajectories collapse to line searches. It is then show that the second formulation PSOAF2 is not objective, although it has the *advantage* that the particle trajectories are n-dimensional space filling. A novel formulation that is *both* objective and diverse, i.e. the algorithm generates particle trajectories that are space filling, is then presented.

4.2 Notes on the investigation

The investigation into the objectivity of the PSOA is started by defining the instantaneous search domain of a particle, viz. the domain to which the search of a particle i at iteration k is restricted as discussed in Chapter 3.

From Eqs. (2.2) and (2.3), it is observed that the instantaneous search domain is composed from a deterministic contribution given by $(\boldsymbol{x}_k^i + w\boldsymbol{v}_k^i)$, and a stochastic contribution due to $\boldsymbol{\nu}_k^i$.

The stochastic domain is bounded, and has an associated probability distribution, due to the random scaling with finite scalars. In order to investigate the objectivity of the stochastic contribution $\boldsymbol{\nu}_k^i$ of the instantaneous search domain, Monte Carlo simulations [56] are used. These are conducted for different values of $\boldsymbol{p}_k^i$, $\boldsymbol{p}_k^g$ and $\boldsymbol{x}_k^i$. Scatter plots are constructed to define the domain of possible stochastic vectors $\boldsymbol{\nu}_k^i$ by generating 10^4 instances of $\boldsymbol{\nu}_k^i$. In all investigations $c_1 = c_2 = 2$.

4.3 Formulation 1 (PSOAF1)

For PSOAF1, the stochastic vector $\boldsymbol{\nu}_k^i$ is given by

$$\boldsymbol{\nu}_k^i = c_1 r_{1k}^i(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i) + c_2 r_{2k}^i(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i), \tag{4.1}$$

where r_{1k}^i and r_{2k}^i represent two uniform real random scalars between 0 and 1, which are updated at every iteration k, and for each particle i in the swarm. The random numbers r_{1k}^i and r_{2k}^i independently scale *only* the magnitudes of the cognitive and social vectors, respectively given by $c_1(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $c_2(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$. The cognitive vector $c_1(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and the social vector $c_2(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ can be anything from normal to parallel w.r.t. each other.

When the cognitive and social vectors are not parallel, Eq. (4.1) may be interpreted as the vector equation of a bounded plane $\boldsymbol{\mathcal{P}}_k^i$ in n-dimensional space. The bounded plane is then translated in

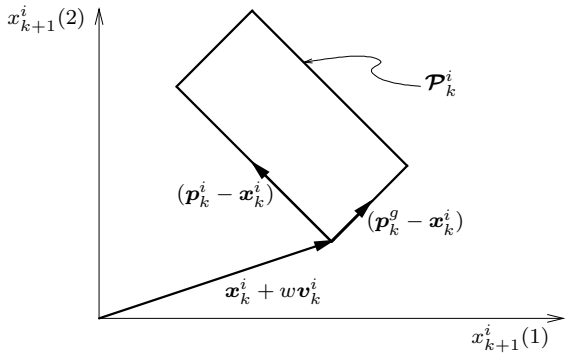


Figure 4.1: Partitioning the position vector $\boldsymbol{x}^i_{k+1}$ into a deterministic contribution $(\boldsymbol{x}^i_k + w\boldsymbol{v}^i_k)$, and a stochastic contribution $(\boldsymbol{\nu}^i_k \in \boldsymbol{\mathcal{P}}^i_k)$, for $c_1 = c_2 = 2$.

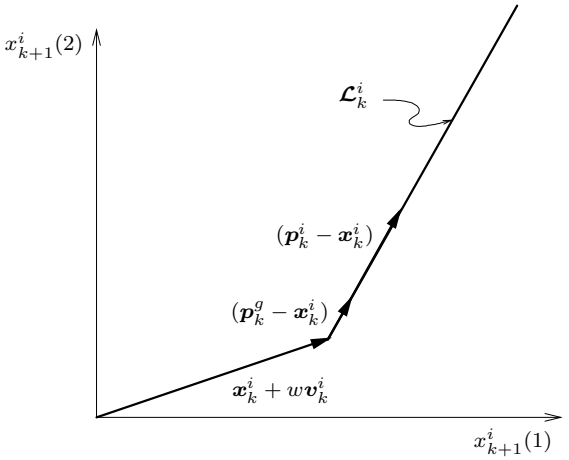


Figure 4.2: Partitioning the position vector $\boldsymbol{x}^i_{k+1}$ into a deterministic contribution $(\boldsymbol{x}^i_k + w\boldsymbol{v}^i_k)$, and a stochastic contribution $(\boldsymbol{\nu}^i_k \in \boldsymbol{\mathcal{L}}^i_k)$, for $c_1 = c_2 = 2$.

n-dimensional space by the addition of $\boldsymbol{x}^i_k$ and $w\boldsymbol{v}^i_k$, as depicted in Figure 4.1.

Whenever the cognitive and social vectors $c_1(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $c_2(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ are parallel, Eq. (4.1) may be interpreted as the vector equation of a bounded line $\boldsymbol{\mathcal{L}}^i_k$ in n-dimensional space. Again, the bounded line is translated in n-dimensional space by the addition of $\boldsymbol{x}^i_k$ and $w\boldsymbol{v}^i_k$, as depicted in Figure 4.2.

The intrinsic properties of a vector are its magnitude and direction; these exist independent of a reference frame [57]. In PSOAF1, only the vector magnitudes (which are invariant) are randomly scaled. Also, since the vectors $c_1(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $c_2(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ are constructed through the subtraction of two vectors, they are also *translationally invariant*. Both criteria for observer independence are met, hence PSOAF1 is objective.

4.3.1 PSOAF1: Investigation of the instantaneous search domain

Objectivity of PSOAF1 is now illustrated by conducting Monte Carlo simulations. Similar simulations will be conducted for the algorithmic formulations in sections to come.

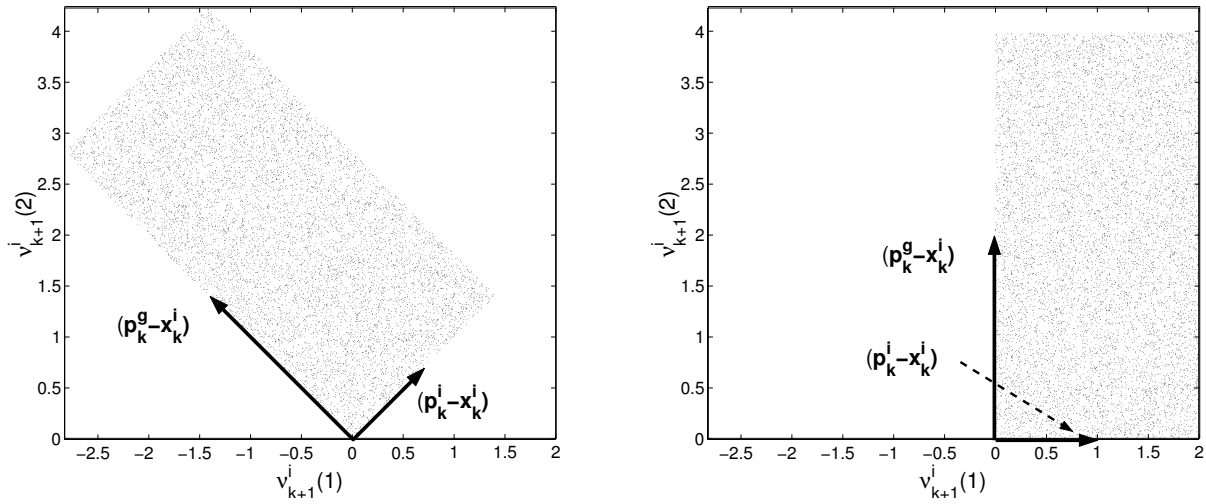


Figure 4.3: PSOAF1: Scatter plot of 10^4 possible stochastic vectors $\boldsymbol{\nu}^i_k$, generated using Monte Carlo simulations, with a) $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right) = [\frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{2}}]$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right) = [-\sqrt{2}\ \sqrt{2}]$ and b) $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right) = [1\ \ 0]$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right) = [0\ \ 2]$. Each point represents the end point of a stochastic vector with origin at [0 0].

First, a study is conducted for non-parallel cognitive and social vectors $c_1(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $c_2(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$. In Figure 4.3(a), the vectors $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right)$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right)$ are respectively given by $[\frac{1}{\sqrt{2}}\ \ \frac{1}{\sqrt{2}}]$ and $[-\sqrt{2}\ \sqrt{2}]$. A scatter plot yields the plane $\boldsymbol{\mathcal{P}}^i_k$, with c_1 and c_2 merely scaling $\boldsymbol{\mathcal{P}}^i_k$.

A scatter plot is then constructed after rotating the vectors $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right)$ and $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right)$ $45°$ clockwise, as depicted in Figure 4.3(b). Hence $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right)$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right)$ are respectively given by $[1\ 0]$ and $[0\ 2]$. From Figure 4.3(b), it is clear that the domain remains a bounded plane $\boldsymbol{\mathcal{P}}^i_k$, which is merely rotated $45°$ clockwise.

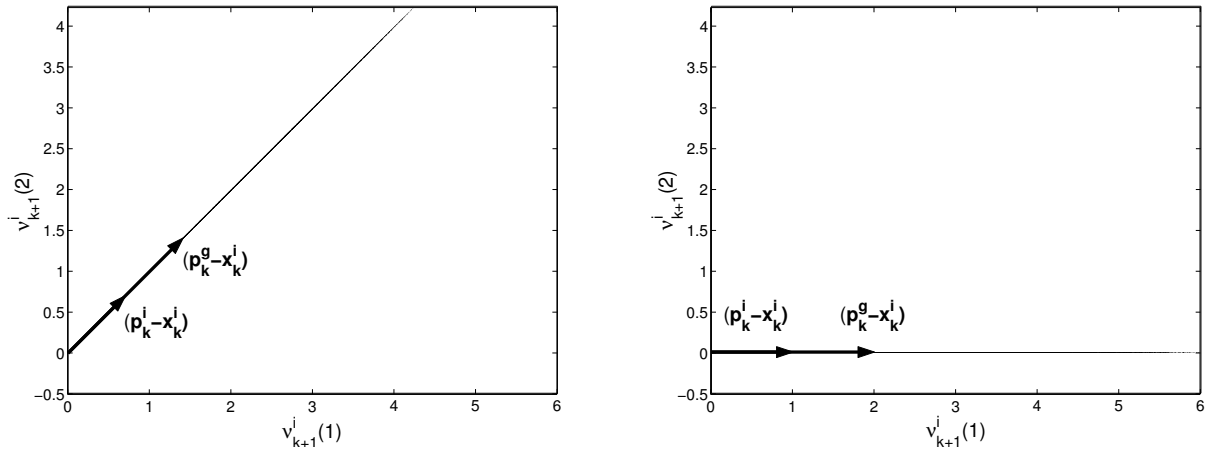
It also follows from random variable theory that the probability distribution over the domain $\boldsymbol{\mathcal{P}}^i_k$ is uniform [50], as illustrated in Figures 4.3(a) and 4.3(b).

Secondly, a similar study is conducted for parallel cognitive and social vectors $c_1(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $c_2(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$, as depicted in Figure 4.4. In Figure 4.4(a), the parallel vectors $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right)$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right)$ are respectively given by $[\frac{1}{\sqrt{2}}\ \ \frac{1}{\sqrt{2}}]$ and $[\sqrt{2}\ \sqrt{2}]$. The domain is a bounded line $\boldsymbol{\mathcal{L}}^i_k$ with c_1 and c_2 merely scaling the length of $\boldsymbol{\mathcal{L}}^i_k$.

Again a scatter plot is constructed after rotating $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right)$ and $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right)$ $45°$ clockwise, as depicted in Figure 4.4(b). Now, $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right)$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right)$ are respectively given by $[1\ 0]$ and $[2\ 0]$. As shown in Figure 4.4(b), the bounded line $\boldsymbol{\mathcal{L}}^i_k$ is merely rotated.

It follows from random variable theory that the probability distribution over the bounded line is tri-linear [50], for different vector lengths $||\boldsymbol{p}^i_k - \boldsymbol{x}^i_k||$ and $||\boldsymbol{p}^g_k - \boldsymbol{x}^i_k||$.

As discussed earlier and graphically demonstrated here, PSOAF1 is objective. A rotation of the vectors $\boldsymbol{p}^i_k$, $\boldsymbol{p}^g_k$ and $\boldsymbol{x}^i_k$ merely results in a rotation of the stochastic domains, $\boldsymbol{\mathcal{P}}^i_k$ and $\boldsymbol{\mathcal{L}}^i_k$. This



(a) (b)

Figure 4.4: PSOAF1: Scatter plot of 10^4 possible stochastic vectors $\boldsymbol{\nu}_k^i$, generated using Monte Carlo simulations, with a) $\left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right) = [\frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{2}}]$ and $\left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right) = [\sqrt{2}\ \sqrt{2}]$ and b) $\left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right) = [1\ \ 0]$ and $\left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right) = [2\ \ 0]$.

follows since only the magnitude of the cognitive and social vectors are scaled in PSOAF1.

4.4 Formulation 2 (PSOAF2)

The stochastic vector $\boldsymbol{\nu}_k^i$ of PSOAF2 is given by

$$\boldsymbol{\nu}_k^i = c_1 \boldsymbol{r}_{1k}^i \circ (\boldsymbol{p}_k^i - \boldsymbol{x}_k^i) + c_2 \boldsymbol{r}_{2k}^i \circ (\boldsymbol{p}_k^g - \boldsymbol{x}_k^i), \tag{4.2}$$

where the $\circ$ operator indicates component by component multiplication between two vectors. Hence the random vectors $\boldsymbol{r}_{mk}^i$ are given by

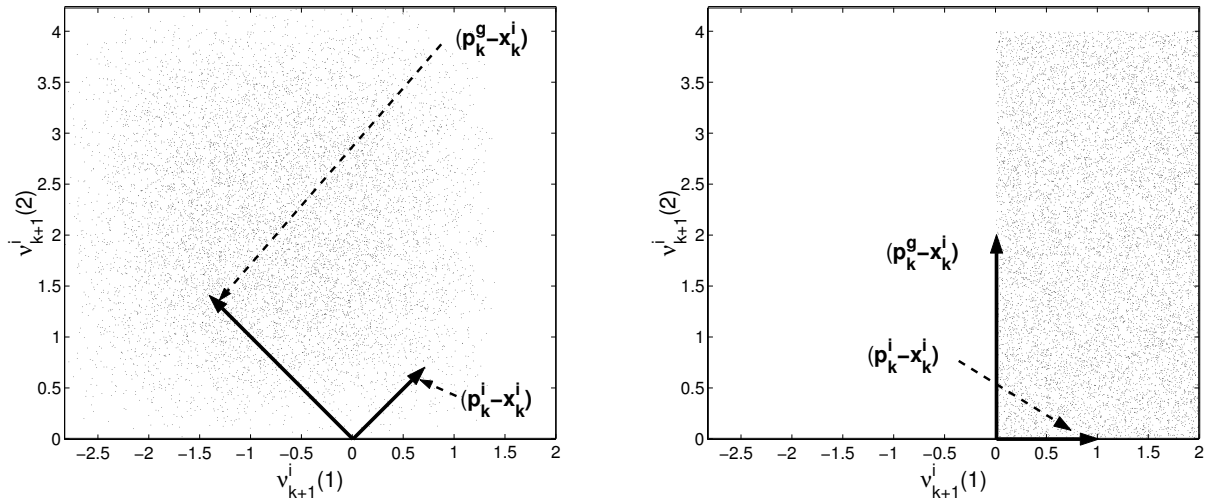
$$\boldsymbol{r}_{mk}^i = (\rho_{1k}^i,\ \rho_{2k}^i, \cdots,\ \rho_{nk}^i), \quad m = 1, 2, \tag{4.3}$$

with $\rho_{lk}^i, l = 1, 2, \cdots, n$ uniform random numbers between 0 and 1. Eq. (4.2) is no longer a vector equation of a bounded plane $\boldsymbol{\mathcal{P}}_k^i$, since every non-zero component of $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ is independently scaled. As a result, the domain of possible stochastic vectors is generalized to n-dimensional space $\boldsymbol{\mathcal{S}}_k^i$.

However, since the components of a vector are given with respect to a specific reference frame, PSOAF2 is *rotationally variant*. (Although PSOAF2 is of course translationally invariant.) Nevertheless, PSOAF2 is observer dependent, since only one of the two criteria of objectivity is met.

4.4.1 PSOAF2: Investigation of the instantaneous search domain

The observer dependence of PSOAF2 is now quantified, using Monte Carlo simulations, similar to those in Section 4.3.1.



(a) (b)

Figure 4.5: PSOAF2: Scatter plot of 10^4 possible stochastic vectors $\boldsymbol{\nu}_k^i$, generated using Monte Carlo simulations with a) $\left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right) = [\frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{2}}]$ and $\left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right) = [-\sqrt{2}\ \sqrt{2}]$ and b) $\left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right) = [1\ \ 0]$ and $\left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right) = [0\ \ 2]$.

As before, the study is conducted for non-parallel cognitive and social vectors $c_1(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $c_2(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$. Figure 4.5(a) depicts that the domain is an n-dimensional space $\boldsymbol{\mathcal{S}}_k^i$ (with $n = 2$ in this case), with c_1 and c_2 merely scaling $\boldsymbol{\mathcal{S}}_k^i$. It is also clear that the probability distribution over $\boldsymbol{\mathcal{S}}_k^i$ is non-uniform.

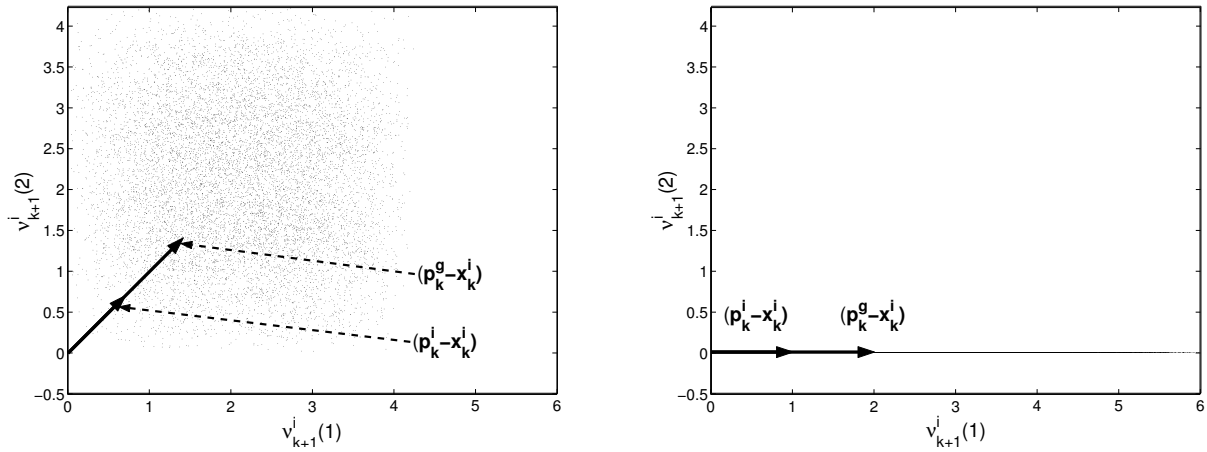
The scatter plot after rotating the vectors $\left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right)$ and $\left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right)$ $45°$ clockwise, is depicted in Figure 4.5(b). It is clear that the *domain changes* after rotation of the vectors. However, the domain remains an n-dimensional space $\boldsymbol{\mathcal{S}}_k^i$, but the size of, and the probability distribution over, the domain depends on the orientation w.r.t. the Cartesian coordinate axis.

The study is repeated for parallel cognitive and social vectors $c_1(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $c_2(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$, as depicted in Figure 4.6(a). The domain is still generalized to n-dimensional space $\boldsymbol{\mathcal{S}}_k^i$ with c_1 and c_2 merely scaling the size of $\boldsymbol{\mathcal{S}}_k^i$.

The scatter plot after rotating $\left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right)$ and $\left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right)$ $45°$ clockwise, is depicted in Figure 4.6(b). It is clear that the domain changes significantly after rotation of the vectors. In fact, the domain collapses to a bounded line $\boldsymbol{\mathcal{L}}_k^i$, since both vectors are parallel to one of the Cartesian basis vectors.

As discussed earlier and graphically demonstrated here, PSOAF2 is observer *dependent*. A rotation of the vectors $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ results in the size of, and the probability distribution over, the stochastic domain to change. This follows since PSOAF2 scales the *components* of the cognitive and social vectors. Since the components of a vector are observer dependent, PSOAF2 is also observer dependent.

However, the advantage of PSOAF2 is that the particle trajectories remain space filling in n-dimensional space as shown in Chapter 3. The result is that diversity in particle trajectories are maintained.



(a) (b)

Figure 4.6: PSOAF2: Scatter plot of 10^4 possible stochastic vectors $\boldsymbol{\nu}^i_k$, generated using Monte Carlo simulations with a) $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right) = [\frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{2}}]$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right) = [\sqrt{2}\ \sqrt{2}]$ and b) $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right) = [1\ \ 0]$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right) = [2\ \ 0]$.

4.5 Novel Formulation: PSOAF1*

As discussed in Section 4.3, PSOAF1 is objective, although the particle trajectories collapse to lines. (The advantage of diverse (n-dimensional) particle search trajectories are quantified in Chapter 3.) On the other hand, PSOAF2 allows for particles to have diverse search trajectories, but unfortunately this comes at the cost of sacrificing objectivity.

An implementation of the PSOA is now presented that allows for diverse particle search trajectories, while retaining objectivity. Based on PSOAF1, the novel, diverse implementation, is denoted PSOAF1*.

In PSOAF1*, the vector magnitudes are scaled, and the vector directions of $(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ perturbed, by multiplying each of the above vectors with an independent *random rotation matrix*. The random rotation matrices are constructed anew for each particle i and for every iteration k, hence

$$\boldsymbol{\nu}^i_k = c_1 r^i_{1k} \boldsymbol{Q}^i_{1k}(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k) + c_2 r^i_{2k} \boldsymbol{Q}^i_{2k}(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k), \tag{4.4}$$

with each $\boldsymbol{Q}^i_{lk},\ l = 1, 2$, a random rotation matrix of dimension $n \times n$.

$\boldsymbol{Q}$ is a proper orthogonal matrix (with determinant 1). Numerous methods are available to construct rotation matrices (e.g. see the approach of Salomon [54]. Constructing $n \times n$ matrices using Salomon's routine is however computationally expensive, since $(n-1)(n-2)$ matrix-matrix multiplications are required.)

As a computationally viable alternative, the exponential map is used [58]. There are again numerous ways to construct exponential maps. The simple series method is selected [58]. The general

series expansion of an exponential map is given by

$$\boldsymbol{Q} = \boldsymbol{I} + \boldsymbol{W} + \frac{1}{2}\boldsymbol{W}\boldsymbol{W} + \frac{1}{6}\boldsymbol{W}\boldsymbol{W}\boldsymbol{W} + \cdots, \tag{4.5}$$

where $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{W}$ is a skew matrix.

The random skew matrix $\boldsymbol{W}$ is constructed as follows:

$$\boldsymbol{W} = \frac{\alpha\pi}{180}(\boldsymbol{A} - \boldsymbol{A}^{\mathrm{T}}), \tag{4.6}$$

with $\boldsymbol{A}$ an $n \times n$ random matrix with each entry a uniform random number between -0.5 and 0.5. α is a real scaling factor and superscript T denotes the matrix transpose.

The author selects to construct the exponential map $\boldsymbol{Q}_k^i$ for "small" perturbations, using only the first two terms of a truncated series method, i.e.

$$\boldsymbol{Q}_k^i = \boldsymbol{I} + \boldsymbol{W}_k^i. \tag{4.7}$$

This is the linear approximation to a rotation matrix, and is valid for small perturbations, since the entries of the higher order terms are close to zero. The advantage of the simplification is that the number of matrix-matrix multiplications is zero.

(It is important to note that the variable bounds defining D should be normalized, such that the boundary ranges are equal.)

4.5.1 PSOAF1*: Investigation of the instantaneous search domain

As before, the objectivity of PSOAF1* is quantified using Monte Carlo simulations. In 2 dimensions, $\alpha = 15$ is selected. (Although this is not "small", this serves to clearly illustrate the proposed concept).

Again, the study for non-parallel cognitive and social vectors is conducted, as depicted in Figure 4.7(a). The domain generalizes to n-dimensional space $\boldsymbol{\mathcal{S}}_k^i$, with c_1 and c_2 scaling $\boldsymbol{\mathcal{S}}_k^i$.

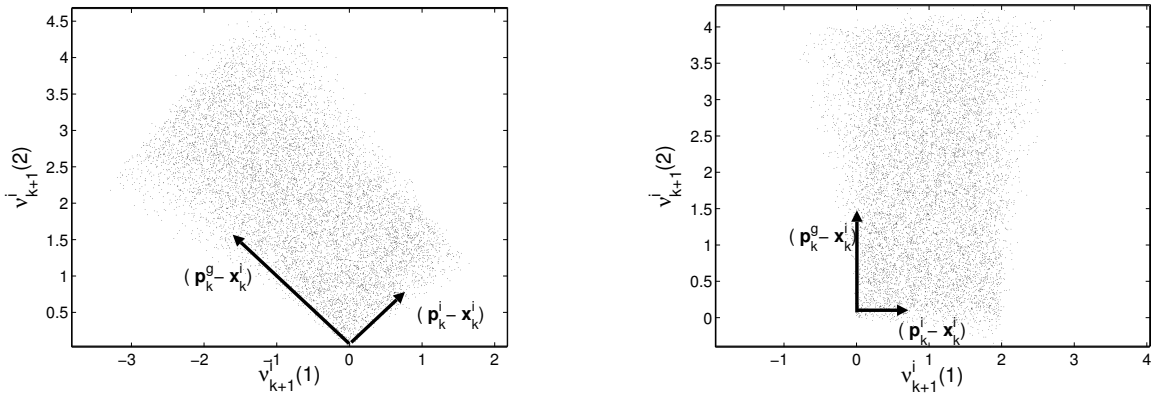
The scatter plot after rotating the vectors $\left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right)$ and $\left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right)$ 45° clockwise is depicted in Figure 4.7(b). Clearly, the domain remains generalized to n-dimensional space $\boldsymbol{\mathcal{S}}_k^i$, rotated 45° clockwise. The probability distribution over the domain $\boldsymbol{\mathcal{S}}_k^i$ is non-uniform.

Secondly, the study for parallel cognitive and social vectors $c_1(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $c_2(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ is conducted. Again the domain generalizes to n-dimensional space $\boldsymbol{\mathcal{S}}_k^i$, with c_1 and c_2 merely scaling the domain.

A scatter plot after rotating $\left(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i\right)$ and $\left(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i\right)$ 45° clockwise is constructed, as depicted in Figure 4.8(b). Evidently, the n-dimensional space $\boldsymbol{\mathcal{S}}_k^i$ is merely rotated, and the probability distribution over the domain is non-uniform.

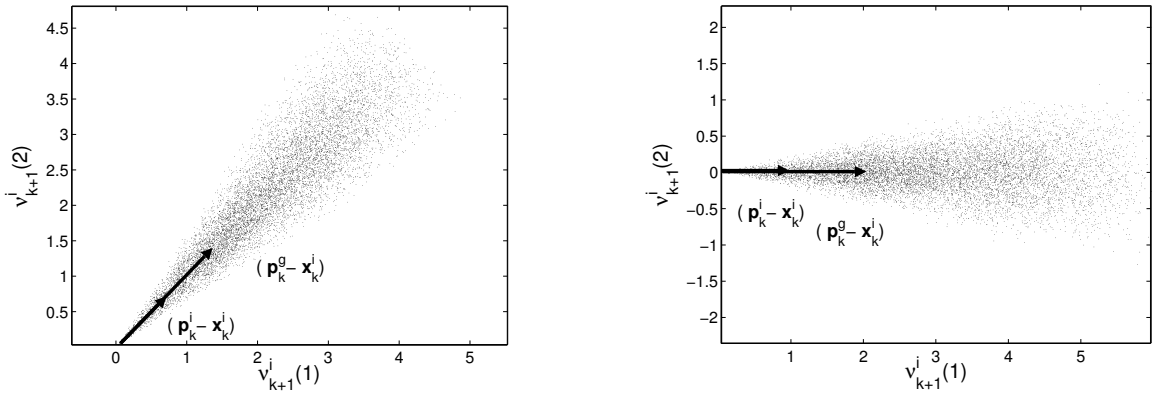
As discussed earlier and graphically demonstrated here, PSOAF1* is an objective formulation. A rotation of the vectors $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ merely results in a rotation of the stochastic domain $\boldsymbol{\mathcal{S}}_k^i$.

The drawback of PSOAF1 is overcome in PSOAF1*, where in addition to scaling the vector magnitudes, the vectors are directionally perturbed. The magnitudes and directions of $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and



(a) (b)

Figure 4.7: PSOAF1*: Scatter plot of 10^4 possible stochastic vectors $\boldsymbol{\nu}^i_k$, generated using Monte Carlo simulations, with a) $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right) = [\frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{2}}]$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right) = [-\sqrt{2}\ \sqrt{2}]$ and b) $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right) = [1\ \ 0]$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right) = [0\ \ 2]$.



(a) (b)

Figure 4.8: PSOAF1*: Scatter plot of 10^4 instances of the stochastic vectors $\boldsymbol{\nu}^i_k$, generated using Monte Carlo simulations, with a) $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right) = [\frac{1}{\sqrt{2}}\ \ \frac{1}{\sqrt{2}}]$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right) = [\sqrt{2}\ \ \sqrt{2}]$ and b) $\left(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k\right) = [1\ \ 0]$ and $\left(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k\right) = [2\ \ 0]$.

$(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$ are used to only indicate potential improvement, thereby placing only *some faith* in *both* direction and step size.

This is in contrast to PSOAF1, where *absolute faith* is placed in the directions prescribed by $(\boldsymbol{p}_k^i - \boldsymbol{x}_k^i)$ and $(\boldsymbol{p}_k^g - \boldsymbol{x}_k^i)$, while only *some faith* is placed on step size.

(Incidentally, termination occurs when $\boldsymbol{p}_k^i$, $\boldsymbol{p}_k^g$ and $\boldsymbol{x}_k^i$ converge on the same point in n-dimensional space, combined with $w\boldsymbol{v}_k^i \to \mathbf{0}$.)

4.6 Numerical experiments

An empirical study to quantify the (lack of) objectivity of the three discussed implementations of the PSOA is now performed. A synchronous updating method is used [31]. Real variables are implemented using double-precision floating-point arithmetic. For this study the algorithm parameters are $c_1 = c_2 = 2$, the swarm size is $p = 20$ particles and the computations are performed for various constant inertia factors w. Initial velocities are assumed to equal $\mathbf{0}$. In PSOAF1*, $\alpha = 3$ is simply selected. (The author does not seek an optimal value for α, but merely wishes to illustrate the effects of perturbing the vector directions.) Furthermore, no boundary or velocity restrictions are implemented. Each run consists of 200000 function evaluations (10000 iterations). All results presented are averaged over 100 runs.

In the study the following five test functions are used:
i) The Rosenbrock function (unimodal, f_0):

$$f_0(\boldsymbol{x}) = \sum_{i=1}^{\frac{n}{2}} \Big(100\big(x_{2i} - x_{2i-1}^2\big)^2 + \big(1 - x_{2i-1}\big)^2\Big).$$

ii) The Quadric function (unimodal, f_1):

$$f_1(\boldsymbol{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2.$$

iii) The Ackley function (multimodal, f_2):

$$\begin{aligned} f_2(\boldsymbol{x}) = \; & -20 \exp\left(-0.2\sqrt{\tfrac{1}{n}\textstyle\sum_{i=1}^{n} x_i^2} \right) \\ & - \exp\left(\tfrac{1}{n}\textstyle\sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e. \end{aligned}$$

iv) The generalized Rastrigin function (multimodal, f_3):

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

v) Finally, the generalized Griewank function (multimodal, f_4):

$$f_4(\boldsymbol{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1.$$

Table 4.1: Test function parameters

Function	n	domain
f_0	30	$\pm$ 2.048
f_1	30	$\pm$ 100.0
f_2	30	$\pm$ 30.0
f_3	30	$\pm$ 5.12
f_4	30	$\pm$ 600.0

The parameters used in the study are given in Table 4.1. The *domain* column represents the range of each dimension of the design variables; the test function domains are symmetrical about 0.0 in all dimensions.

The multimodal functions f_2 and f_3 are decomposable [54], viz. the design variables are uncoupled. This implies that once an optimal value for a given design variable is obtained, it remains optimal, independent of the other design variables. This is similar to optimizing n 1-dimensional optimization problems, instead of 1 n-dimensional coupled optimization problem. The test set is therefore studied in the unrotated or decomposable reference frame $f(\boldsymbol{x})$, as well as in an arbitrary rotated reference frame $f(\boldsymbol{Rx})$, in which the design variables are coupled [55]. Here, $\boldsymbol{R}$ is a random, proper orthogonal transformation matrix, constructed as in [54]. The transformation matrix results in a pure rotation of each test function. For each of the 100 independent runs, a new random rotation matrix $\boldsymbol{R}$ is constructed, to ensure that there is no bias toward any particular reference frame.

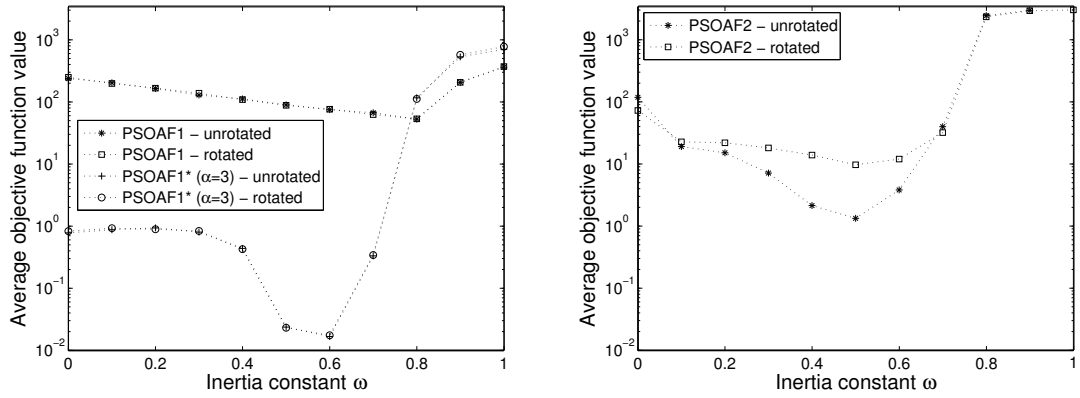
4.7 Discussion of Results

Depicted in Figures 4.9, 4.10, 4.11, 4.12 and 4.13 are the mean objective function values after 2×10^5 function evaluations (or 10000 iterations) averaged over 100 runs for both the unrotated and rotated functions.

The *rotational invariance* of PSOAF1 and PSOAF1* are evident from Figures 4.9(a), 4.10(a), 4.11(a), 4.12(a) and 4.13(a). The poor performance of PSOAF1 directly results from the particle trajectories collapsing to lines as shown in Chapter 3. There is a significant improved performance for all the test functions with PSOAF1*, due to the scaling of the vector magnitudes and perturbation of the vector directions.

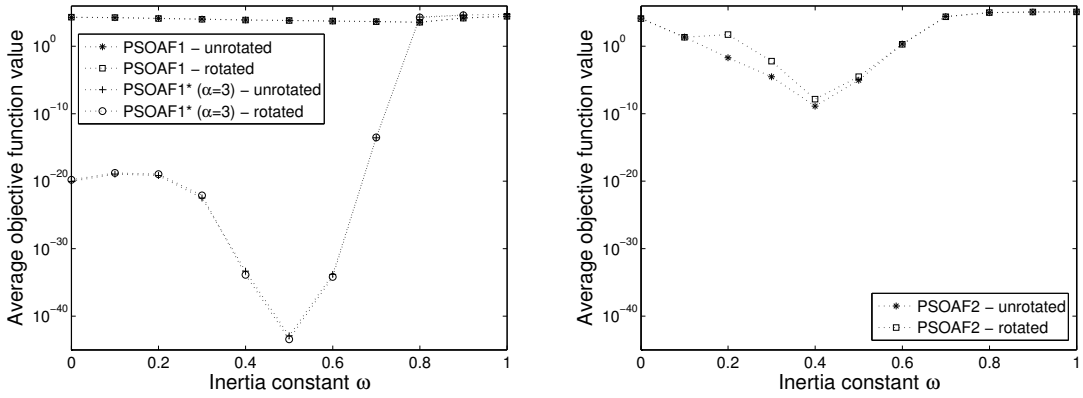
The *rotational variance* of PSOAF2 is evident from Figures 4.9(b), 4.10(b), 4.11(b), 4.12(b) and 4.13(b). There is a *severe* performance loss for some of the rotated functions compared to the unrotated functions.

Two functions result in similar performance for the rotated and unrotated functions, namely the Quadric function f_1, and the Griewank function, f_4. The Quadric and Griewank functions are almost insensitive to rotation. (The Griewank function is a spherical function on which sinusoidal "noise" is imposed. Hence this function is artificially indifferent to rotation, since many local minima appear, irrespective of rotation.)



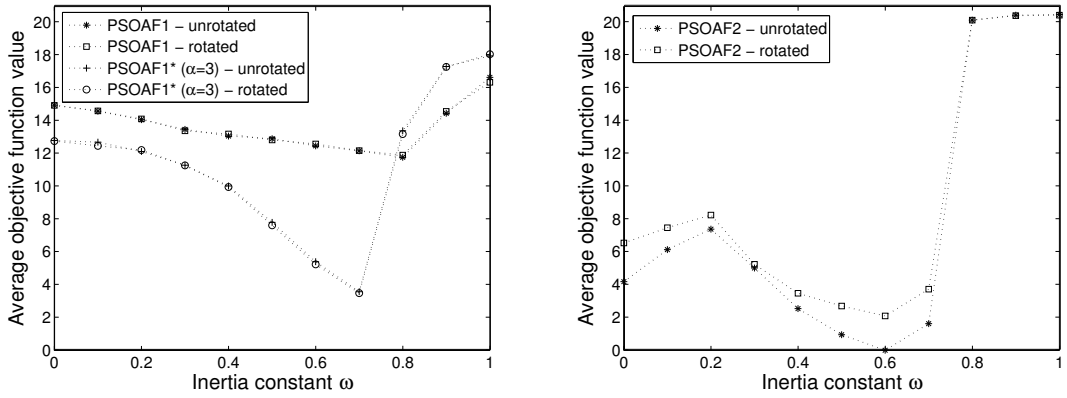
(a) (b)

Figure 4.9: Average function value obtained with a) PSOAF1 and PSOAF1*, and b) PSOAF2 after 2×10^5 function evaluations (10000 iterations) averaged over 100 runs on the rotated and unrotated Rosenbrock test function f_0.



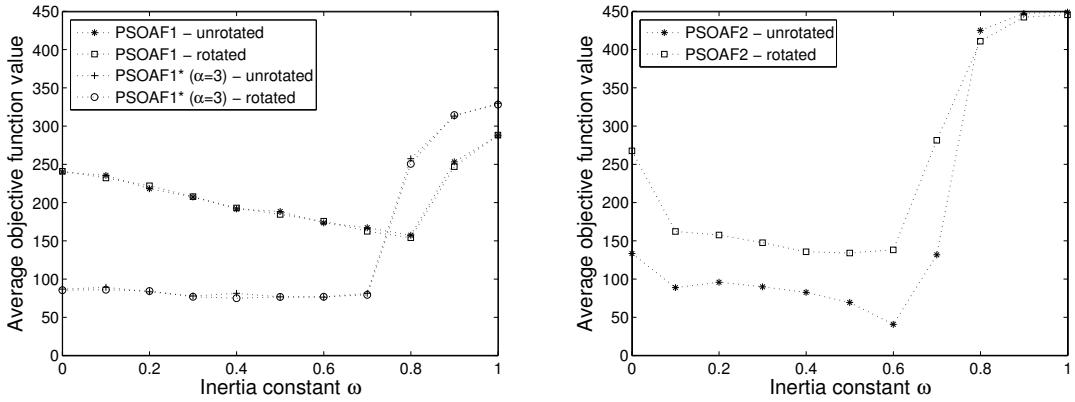
(a) (b)

Figure 4.10: Average function value obtained with a) PSOAF1 and PSOAF1*, and b) PSOAF2 after 2×10^5 function evaluations (10000 iterations) averaged over 100 runs on the rotated and unrotated Quadric test function f_1.



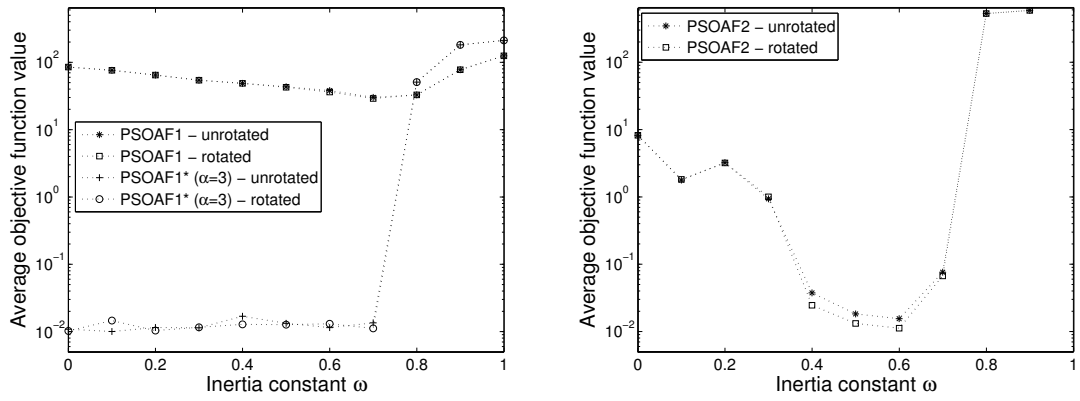
(a) (b)

Figure 4.11: Average function value obtained with a) PSOAF1 and PSOAF1*, and b) PSOAF2 after 2×10^5 function evaluations (10000 iterations) averaged over 100 runs on the rotated and unrotated Ackley test function f_2.



(a) (b)

Figure 4.12: Average function value obtained with a) PSOAF1 and PSOAF1*, and b) PSOAF2 after 2×10^5 function evaluations (10000 iterations) averaged over 100 runs on the rotated and unrotated Rastrigin test function f_3.



(a) (b)

Figure 4.13: Average function value obtained with a) PSOAF1 and PSOAF1*, and b) PSOAF2 after 2×10^5 function evaluations (10000 iterations) averaged over 100 runs on the rotated and unrotated Griewank test function f_3.

Inadvertently, this also suggest that non-spherical unimodal test functions should be used to evaluate objectivity. The two unimodal functions, namely the Rosenbrock function f_0 and the Quadric function f_1, are of some interest, since they indicate the ability of an algorithm to search within a local basin. The performance of PSOAF1* is significantly better than PSOAF2 for both functions, for both the rotated and unrotated test functions. PSOAF2 demonstrates a severe performance loss for the Rosenbrock function, for the rotated function compared to the unrotated function. (Note the scale of the graphs in Figure 4.9.)

The performance difference between PSOAF2 and PSOAF1* for the unimodal Quadric test function f_1, is depicted in Figure 4.14. Figure 4.14 depicts the mean function value convergence history of PSOAF1 (with $w = 0.8$), PSOAF2 (with $w = 0.4$) and PSOAF1* (with $w = 0.5$ and $\alpha = 3$) over 2500 iterations. The values for w are optimal for each algorithm, but no attempt was made to optimize α. Of the three formulations, it is clear that PSOAF1* is computationally the most effective on the Quadric test function.

For the multimodal functions, PSOAF2 demonstrates notable performance loss. See for example the Ackley function f_2, and the Rastrigin function f_3. In contrast, the performance of PSOAF1* is comparable to the best obtained with PSOAF2, with no performance loss due to rotation.

For the sake of clarity, an overview of the performances of PSOAF1, PSOAF2 and PSOAF1* is given in Table 4.2. The table summarizes the best function values obtained, together with the inertia factor at which the best function value is obtained after 2×10^5 function evaluations (10000 iterations). The results for both the unrotated and rotated test functions are given.

Table 4.2: Constant inertia factor at which the best average objective function value is obtained for the unrotated test functions. The accompanying average objective function value for rotated test functions is also presented.

	PSOAF1		
		$f_{\text{unrotated}}$	f_{rotated}
	w	$f_{\text{avg}}^{\text{best}}$	$f_{\text{avg}}^{\text{best}}$
f_0	0.8	54.071	54.712
f_1	0.8	4087.657	4123.801
f_2	0.8	11.791	11.925
f_3	0.8	157.656	154.368
f_4	0.7	30.924	29.731
	PSOAF2		
		$f_{\text{unrotated}}$	f_{rotated}
	w	$f_{\text{avg}}^{\text{best}}$	$f_{\text{avg}}^{\text{best}}$
f_0	0.5	1.358	9.905
f_1	0.4	1.4×10^{-9}	1.5×10^{-8}
f_2	0.6	8.6×10^{-15}	2.099
f_3	0.6	40.992	138.507
f_4	0.6	1.5×10^{-2}	1.1×10^{-2}
	PSOAF1* ($\alpha = 3$)		
		$f_{\text{unrotated}}$	f_{rotated}
	w	$f_{\text{avg}}^{\text{best}}$	$f_{\text{avg}}^{\text{best}}$
f_0	0.6	1.6×10^{-2}	1.7×10^{-2}
f_1	0.5	1.2×10^{-43}	3.8×10^{-44}
f_2	0.7	3.583	3.492
f_3	0.5	76.949	76.880
f_4	0.6	1.1×10^{-2}	1.3×10^{-2}

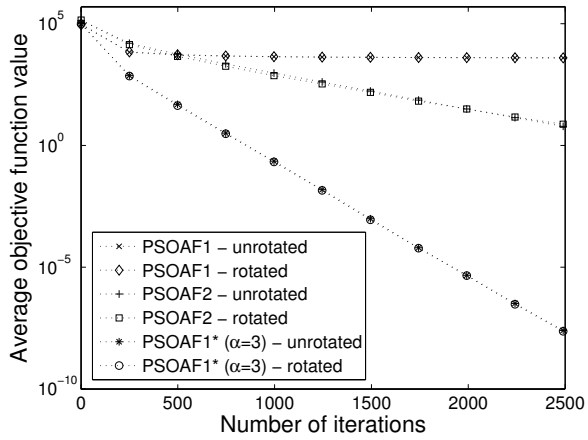


Figure 4.14: Mean function value history plot averaged over 100 runs on the rotated and unrotated Quadric test function f_3 with PSOAF1 (with $w = 0.8$), PSOAF2 (with $w = 0.4$) and PSOAF1* (with $w = 0.5$ and $\alpha = 3$).

4.8 Comments on PSOAF1*

4.8.1 On invariance

It is now shown that PSOAF1* is not strictly rotationally invariant, but only in a stochastic sense.

Consider an arbitrary vector, expressed in two different reference frames, by respectively $\boldsymbol{y}$ and $\boldsymbol{y}'$. The two reference frames are related by a pure rotation $\boldsymbol{M}$, hence

$$\boldsymbol{y}' = \boldsymbol{M}\boldsymbol{y}, \quad \boldsymbol{M} \in \text{Orth}^+, \tag{4.8}$$

where Orth^+ indicates the space of proper orthogonal matrices.

Now apply two independent directional perturbations (rotations) $\boldsymbol{Q} \in \text{Orth}^+$ and $\boldsymbol{Q}' \in \text{Orth}^+$ to $\boldsymbol{y}$ and $\boldsymbol{y}'$ respectively. The vectors $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{y}}'$ then obtained, are respectively given by

$$\hat{\boldsymbol{y}} = \boldsymbol{Q}\boldsymbol{y}, \tag{4.9}$$

and

$$\hat{\boldsymbol{y}}' = \boldsymbol{Q}'\boldsymbol{y}'. \tag{4.10}$$

Strict deterministic rotational invariance requires a one-to-one mapping of the perturbed vectors in either reference frame. Hence

$$\hat{\boldsymbol{y}}' = \boldsymbol{M}\hat{\boldsymbol{y}}, \quad \forall\, \boldsymbol{M} \in \text{Orth}^+ \tag{4.11}$$

By substituting Eqs. (4.8), (4.9) and (4.10) into Eq. (4.11), the following is obtained

$$\boldsymbol{Q}'\boldsymbol{M}\boldsymbol{y} = \boldsymbol{M}\boldsymbol{Q}\boldsymbol{y}, \quad \forall\, \boldsymbol{M} \in \text{Orth}^+. \tag{4.12}$$

Eq. (4.12) is rewritten as

$$(\boldsymbol{Q}'\boldsymbol{M} - \boldsymbol{M}\boldsymbol{Q})\boldsymbol{y} = \boldsymbol{0}, \quad \forall\, \boldsymbol{M} \in \text{Orth}^+. \tag{4.13}$$

Since Eq. (4.13) has to hold for any arbitrary vector $\boldsymbol{y}$, it follows that

$$\boldsymbol{Q}'\boldsymbol{M} = \boldsymbol{M}\boldsymbol{Q}, \quad \forall\ \boldsymbol{M} \in \text{Orth}^{+}. \tag{4.14}$$

The unique solution to Eq. (4.14) is that both $\boldsymbol{Q}'$ and $\boldsymbol{Q}$ are the second-order isotropic tensor, i.e.

$$\boldsymbol{Q}' = \boldsymbol{Q} = \boldsymbol{I}. \tag{4.15}$$

The foregoing implies that a strict enforcement of rotational invariance results in $\boldsymbol{Q}^i_{lk} = \boldsymbol{I}, \quad l = 1, 2$. In other words, PSOAF1* reduces to PSOAF1.

However, since the PSOA is a stochastic algorithm, it is adequate to satisfy Eq. (4.15) in an average sense only. In order to satisfy $\boldsymbol{Q}' = \boldsymbol{Q} = \boldsymbol{I}$ in a stochastic sense, it is sufficient to require that $\text{mean}(\boldsymbol{Q}') = \text{mean}(\boldsymbol{Q}) = \boldsymbol{I}$, if the probability distributions of $\boldsymbol{Q}'$ and $\boldsymbol{Q}$ are chosen equal over identical domains.

4.8.2 Implementational issues of PSOAF1*

Further to the implementation in Section 4.5, numerous strategies exist to achieve independent directional perturbation.

An obvious, computationally inexpensive possibility is to randomly perturb each component of the unit vectors $(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)/||(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)||$ and $(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)/||(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)||$; the vectors $(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ are then reconstructed from the normalization of the perturbed vectors. (Although this makes a rigorous mathematical analysis of the algorithm difficult.)

In the implementation, in updating $\boldsymbol{Q}$, strategies to limit the computational expense associated with matrix multiplications and the generation of random numbers may also be implemented. For example, multiplying the sum of $c_1(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $c_2(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ by a single random rotation matrix, reduces the number of matrix multiplications by half.

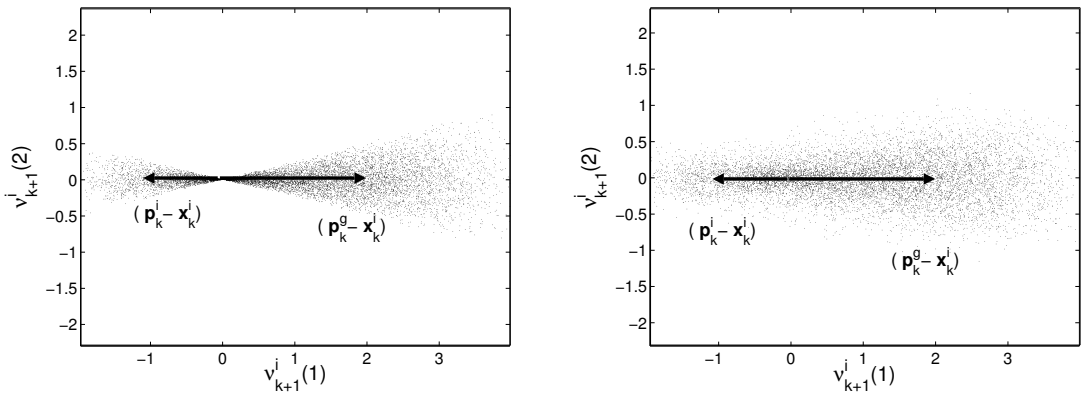
However, depicted in Figure 4.15 is the difference in instantaneous search domain that results when independent rotation matrices $\boldsymbol{Q}^i_{1k} \neq \boldsymbol{Q}^i_{2k}$ are used, as opposed to identical rotation matrices $\boldsymbol{Q}^i_{1k} = \boldsymbol{Q}^i_{2k}$.

To reduce the computational effort even further, the vectors $c_1(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $c_2(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$ of all the particles can be directionally perturbed by the same independent rotation matrices, viz. $\boldsymbol{Q}^i_{lk} = \boldsymbol{Q}_{lk}, \ l = 1, 2$ and $i = 1, 2, \cdots, p$.

4.8.3 Alternatives to PSOAF1*

Finally, there are of course numerous methods to introduce diversity into PSOAF1, as opposed to the proposed option of independent directional perturbation.

Only a single alternative is mentioned here, namely an increase in the social awareness of the particles. In turn, this may for example be effected by increasing the number of particles that contribute to Eq. (4.1) [35, 34]. (One may of course achieve n-dimensional searches, if the number of particles $p \geq n$, unless the particle trajectories are parallel.) Additional information about the objective function is then also used in the searches of any particle i.



(a) (b)

Figure 4.15: Scatter plot of 10^4 possible stochastic vectors $\boldsymbol{\nu}^i_k$, generated using Monte Carlo simulations, with $(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k) = [-1\ 0]$ and $(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k) = [2\ 0]$ using a) identical rotation matrices $\boldsymbol{Q}^i_{1k} = \boldsymbol{Q}^i_{2k}$ and b) independent rotation matrices $\boldsymbol{Q}^i_{1k} \neq \boldsymbol{Q}^i_{2k}$.

4.9 Closure

It is shown that PSOAF1 is objective, but it demonstrates an overall poor performance, due to the particle trajectories collapsing to lines. This is a direct result of only scaling the magnitude of the cognitive and social vectors $c_1(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $c_2(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$.

In turn, PSOAF2 is not objective, which results in severe performance loss for "rotated" functions. Nevertheless, PSOAF2 still outperforms PSOAF1 for both rotated and unrotated test functions, since the algorithm is diverse, i.e. the particle trajectories do not collapse to lines.

A novel implementation denoted PSOAF1* is proposed, which is both objective and diverse. In PSOAF1*, the magnitudes are scaled, and the directions perturbed independently, of both the cognitive and social vectors $c_1(\boldsymbol{p}^i_k - \boldsymbol{x}^i_k)$ and $c_2(\boldsymbol{p}^g_k - \boldsymbol{x}^i_k)$. (This however comes at the cost of an additional scaling factor.) PSOAF1* outperforms PSOAF2 for the unimodal functions used, for both the rotated and unrotated test functions. In addition, its performance is comparable to PSOAF2 for the multimodal functions, with the added advantage of being independent of the reference frame in which the objective function is formulated.

Chapter 5

Shape optimization problem

5.1 Introduction

Shape optimization involves the constrained minimization of a cost function. The cost function in turn typically involves the solutions of a system of partial differential equations, which depend on parameters that define a geometrical domain [59]. The continuum description of the geometrical domain is normally discretized. This allows for efficient solution of the system of partial differential equations, using for example the finite element method (FEM). Normally, the discretized geometric domain is defined by control variables with predefined freedom. The control variables in turn bound the geometrical domain through a predefined relationship, which may be piecewise linear, or based on B-splines, etc.

Furthermore, different meshing strategies can be used. These include fixed grid strategies [60, 61, 62], design element concepts [63], adaptive mesh strategies [64], and remeshing strategies. The first three methods imply an *a priori* mesh discretization with obvious limitations, for example when dealing with large shape changes in the geometry during optimization. Some of the drawbacks of remeshing strategies are the implementation expense, and the tendency of gradient based optimization methods to get trapped in local minima [65]. On the other hand, the (unstructured) remeshing strategies allow for generality in structural models and objective functions. Large shape changes can be accommodated using the remeshing strategy with minimum mesh distortion.

The cost function may be optimized using either a gradient free or gradient based optimization method. While the gradient free methods require only the relationship between the cost function and the discretized geometric domain to be specified, the gradient based optimization methods require additional sensitivity information. The sensitivities needed for the gradient optimization techniques can either be calculated numerically, semi-analytically or analytically. All these methods have merits and drawbacks. Numerical gradients using finite difference methods are computationally expensive, but are easily implementable. The semi-analytical and analytical methods are more complex to implement, but are computationally cheaper. The advantage of gradient free evolutionary strategies are their global optimization capability. They have been used with success by Xie and Steven [61, 62] in a fixed grid strategy. Related works of evolutionary strategies in shape optimization are the biological growth method of Mattheck and Burkhardt [66], and the genetic algorithm used by Garcia and Gonzales [60].

In shape optimization, mesh generation plays an important role and is in general an expensive aspect of a computational iteration when fixed grid strategies are not considered. Remeshing strategies in shape optimization accentuate the importance of robustness, computational speed, flexibility and accuracy of the mesh generator in discretizing the geometrical domain.

In this study a remeshing shape optimization environment is combined with the gradient free particle swarm optimization algorithm (PSOA) developed by Kennedy and Eberhart.

5.2 Problem formulation

The problem under consideration is now the general inequality constrained minimization problem stated as follows: Given a cost function $f\big(\Omega(\boldsymbol{x})\big)$, find the minimum f^* such that

$$f^* = f\big(\Omega^*(\boldsymbol{x}^*)\big) = \min_{\boldsymbol{x}\in\mathbb{R}^n}\{f\big(\Omega(\boldsymbol{x})\big) : \boldsymbol{g}\big(\Omega(\boldsymbol{x}),\boldsymbol{x}\big) \leq \boldsymbol{0}\}, \tag{5.1}$$

where $\mathbb{R}^n$ represents a set of n real numbers. The cost function $f(\boldsymbol{x})$ and the constraints $g_j(\boldsymbol{x}),\ j = 1, 2, \cdots, m$ are scalar functions of the control variables $\boldsymbol{x}$ and the geometrical domain $\Omega(\boldsymbol{x})$, which is also a function of the control variables $\boldsymbol{x}$. For the sake of brevity, the cost function and the constraints will respectively be denoted by $f(\boldsymbol{x})$ and $\boldsymbol{g}(\boldsymbol{x})$; this notation will however imply dependency on $\Omega(\boldsymbol{x})$. We choose to represent the geometrical domain boundary $\partial\Omega$ by a simple piecewise linear interpolation between the control variables. However, Bezier curves or B-splines, etc. may of course also be used.

In our case, the cost function $f(\boldsymbol{x}) = f\big(\boldsymbol{u}(\Omega(\boldsymbol{x}))\big)$ is an explicit function of the nodal displacements $\boldsymbol{u}$, which is obtained by solving the approximate finite element equilibrium equations for linear elasticity, formulated as

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}, \tag{5.2}$$

where $\boldsymbol{K}$ represents the assembled structural stiffness matrix and $\boldsymbol{f}$ the consistent structural loads.

5.2.1 Accommodation of constraints

Shape problems are subjected to an inequality constraint $g\big(\Omega(\boldsymbol{x}),\boldsymbol{x}\big) \leq \boldsymbol{0}$, being the maximum allowed volume.

The inequality constraints $\boldsymbol{g}\big(\Omega(\boldsymbol{x}),\boldsymbol{x}\big) \leq \boldsymbol{0}$ can be accommodated using a simple exterior penalty formulation. With the exterior penalty formulation the optimal design problem represented by (5.1) is now modified to become: Find the minimum f^* such that

$$\begin{aligned} f^* &= f\big(\Omega^*(\boldsymbol{x}^*)\big) = \min_{\boldsymbol{x}\in R^n}\left\{f\big(\Omega(\boldsymbol{x})\big) + \sum_{j=1}^{m}\Lambda_j[g_j\big(\Omega(\boldsymbol{x}),\boldsymbol{x}\big)]^2\mu_j(g_j)\right\}, \quad \text{with} \\ \mu_j(g_j) &= \begin{cases} 0 & \text{if} \quad g_j\big(\Omega(\boldsymbol{x}),\boldsymbol{x}\big) \leq 0 \\ 1 & \text{if} \quad g_j\big(\Omega(\boldsymbol{x}),\boldsymbol{x}\big) > 0 \end{cases} \quad \text{and penalty parameters } \Lambda_j > 0, \text{ prescribed.} \end{aligned} \tag{5.3}$$

5.3 Mesh generation

In this section, we present triangulation based on the truss structure analogy proposed by Persson and Strang [17]. This incorporates a Delaunay strategy [67] which ensures good mesh quality, albeit at the cost of potentially introducing discontinuities between consecutive meshes due to the addition or removal of nodes. The continuous geometrical domain Ω is discretized by nodes $\boldsymbol{\mathcal{X}}$. The nodes $\boldsymbol{\mathcal{X}}$ are the union of the boundary nodes $\boldsymbol{\mathcal{X}}^{\partial\Omega}$ and the interior nodes $\boldsymbol{\mathcal{X}}^{\Omega}$.

5.3.1 Mesh generator based on a truss structure analogy

The mesh generator is based on a truss structure analogy that solves for the equilibrium position of a truss structure. The geometrical domain is defined by a signed distance function that signs the nodes outside the domain as positive, inside as negative and zero on the boundary. The distance function is a function of the control variables through the interpolation of the domain.

The truss force function F is defined with a force discontinuity, as no tensile forces are permitted in the truss elements. This allows the propagation of the nodes $\boldsymbol{\mathcal{X}}$ to the boundary $\partial\Omega$. The nodes are kept inside the geometrical domain by external forces acting on the boundary nodes $\boldsymbol{\mathcal{X}}^{\partial\Omega}$. The forces act perpendicularly to the boundary, keeping the nodes from moving outside the boundary while allowing movement along the boundary.

The truss force function F is defined as

$$F(l, h_0) = \begin{cases} k(h_0 - l) & \text{if } l < h_0 \\ 0 & \text{if } l \geq h_0 \end{cases} \tag{5.4}$$

with k the spring (truss) stiffness, l the current spring length and h_0 the undeformed spring length (also referred to as the ideal element length). In Persson and Strang's [17] implementation, all springs are precompressed by 20%, which provides the driving force necessary to propagate nodes to the boundary.

The truss system $\boldsymbol{F}(\boldsymbol{\mathcal{X}}) = \mathbf{0}$ is transformed to a system of ordinary differential equations through the introduction of artificial time-dependence in the equations. The system is then solved by a forward Euler method

$$\boldsymbol{\mathcal{X}}_{n+1} = \boldsymbol{\mathcal{X}}_n + \Delta t \boldsymbol{F}(\boldsymbol{\mathcal{X}}_n). \tag{5.5}$$

The forward Euler method is essentially a matrix free method ideally suited to create meshes with a very large number of elements. This method exhibits linear convergence rates.

5.4 Numerical results for the cantilever beam problem

Consider the cantilever beam depicted in Figure 5.1. The domain has a predefined length of 30 units and a height of 10 units. The objective is to minimize the maximum vertical displacement of the structure, subject to a maximum volume constraint of 70%. The magnitude of F is 10 N.

Numerical results for the cantilever beam problem are presented, using linear elastic finite element analyses under plane stress conditions. All the simulations are performed using material property

values of $E = 200 \times 10^3$ for Young's modulus, and $\nu = 0.3$ for Poisson's ratio. An elemental thickness of unity and ideal element length $h_0 = 1$ is used throughout. 21 control variables are used.

For the PSOA, a swarm size of $p = 20$ is used. The social and cognitive parameters are selected as $c_1 = c_2 = 2$. The constant inertia factors for PSOAF1 is $w = 0.7$, for PSOAF2 $w = 0.5$ and for PSOAF1* $w = 0.6$, together with $\alpha = 5$. Each run consists of 20000 function evaluations (1000 iterations). Results are obtained by conducting 10 independent runs for each formulation.

The volume constraint is accommodated using the exterior penalty approach with the exterior penalty parameter selected as $\Lambda = 10^8$. When a particle's position violates a simple bound the position update is allowed but no function evaluation is conducted. The simple bounds for this problem are $1 \leq x_k^i(d) \leq 10, \quad d = 1, 2, \ldots, n$. No velocity restrictions are implemented and the initial velocities are set equal to $\mathbf{0}$.

The chosen settings for each formulation are based on judgment obtained from the results of the test sets in Chapter 4. The aim is not to determine or use optimal settings for each formulation, but merely to illustrate the applicability of the PSOA in shape optimization.

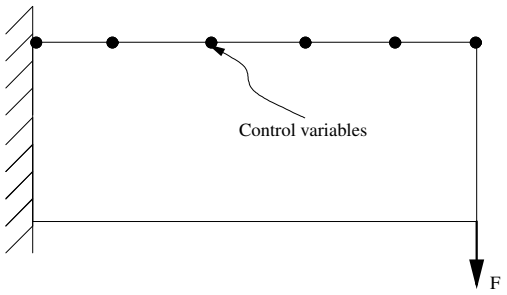


Figure 5.1: Initial structure and definition for the cantilever beam.

The results are summarized in Table 5.1 and Figure 5.2 illustrates the average convergence history over 10 runs. As shown, for the selected settings, PSOAF1* demonstrates the fastest initial convergence. PSOAF1 stagnates rapidly, due to the particle trajectories collapsing to line searches.

Selected shapes are depicted in Figures 5.3–5.11; this includes the best function value f^{best} obtained for each run, and the mean best function value $f^{\text{best}}_{\text{mean}}$ obtained over 10 runs. The statistical significance of using only 10 runs allows at most for only general observations. The performance of PSOAF1 is poor, as illustrated visually and shown numerically. PSOAF1* and PSOAF2 have comparable performance.

Table 5.1: Optimal results for the cantilever beam after only 1000 iterations.

Run	PSOAF1	PSOAF2	PSOAF1*
	$f^{\text{best}}(\times 10^{-2})$	$f^{\text{best}}(\times 10^{-2})$	$f^{\text{best}}(\times 10^{-2})$
1	1.981224	1.001660	1.000307
2	2.585177	1.001963	1.004114
3	1.814116	1.002475	1.014873
4	1.570862	1.002285	1.000066
5	1.578468	1.002799	1.004792
6	2.586973	1.000644	1.000284
7	1.488452	1.003684	1.000723
8	1.612004	1.000872	1.000594
9	2.136010	1.001666	1.001907
10	2.082606	1.000784	1.000346
$f^{\text{best}}_{\text{mean}}(\times 10^{-2})$	1.943589	1.001883	1.002800

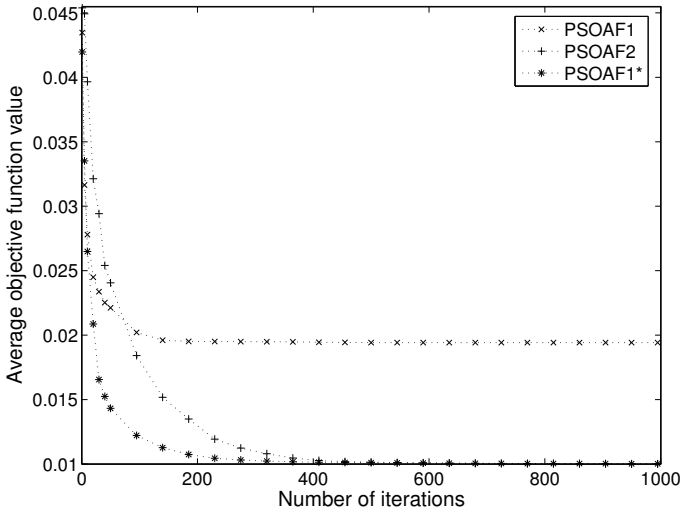
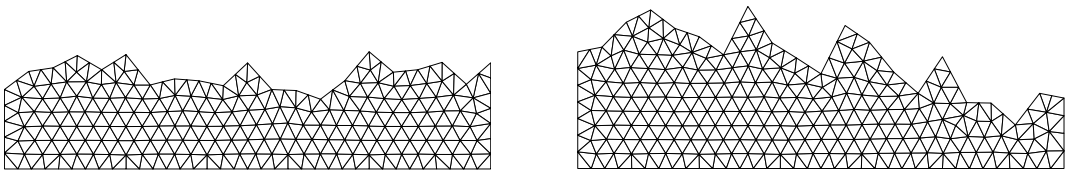
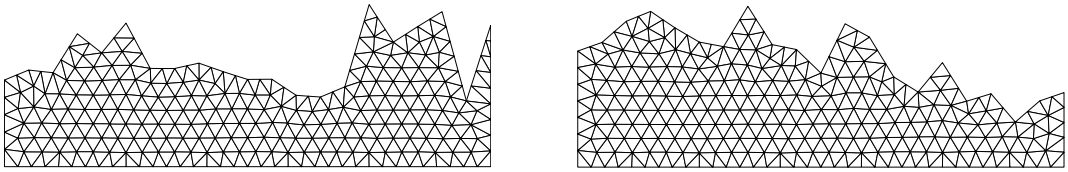


Figure 5.2: Mean function value history plot averaged over 10 runs for the cantilever beam shape optimization problem for PSOAF1 (with $w = 0.7$), PSOAF2 (with $w = 0.5$) and PSOAF1* (with $w = 0.6$ and $\alpha = 5$).



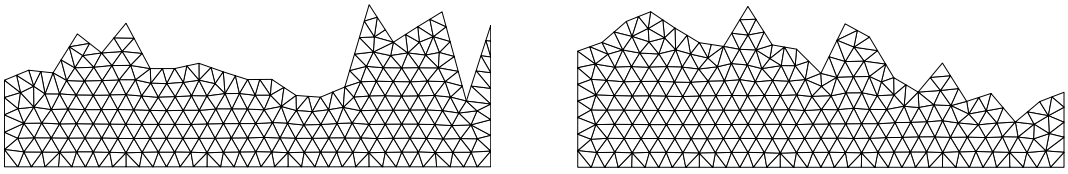
(a) (b)

Figure 5.3: Cantilever beam: results obtained after 100 iterations with PSOAF1 for a) the worst run (run 6), and b) the best run (run 7).



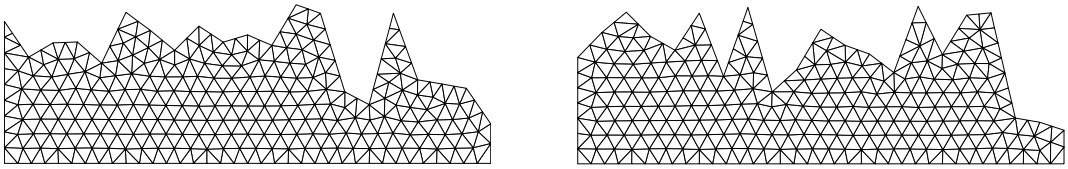
(a) (b)

Figure 5.4: Cantilever beam: results obtained after 500 iterations with PSOAF1 for a) the worst run (run 6), and b) the best run (run 7).



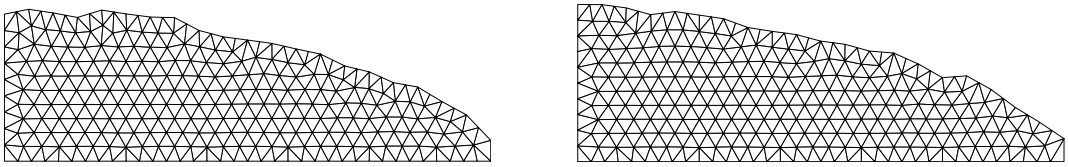
(a) (b)

Figure 5.5: Cantilever beam: results obtained after 1000 iterations with PSOAF1 for a) the worst run (run 6), and b) the best run (run 7).



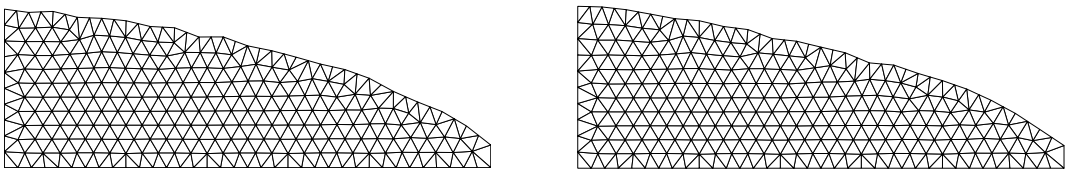
(a) (b)

Figure 5.6: Cantilever beam: results obtained after 100 iterations with PSOAF2 for a) the worst run (run 7), and b) the best run (run 6).



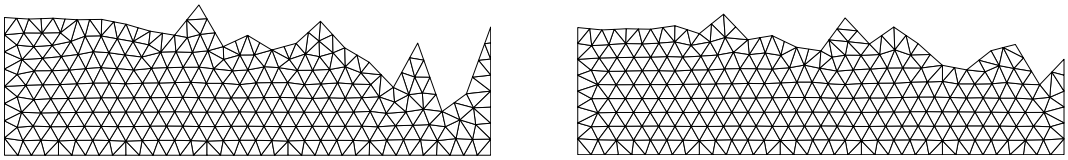
(a) (b)

Figure 5.7: Cantilever beam: results obtained after 500 iterations with PSOAF2 for a) the worst run (run 7), and b) the best run (run 6).



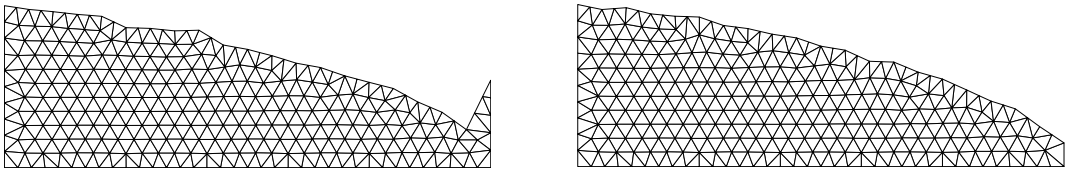
(a) (b)

Figure 5.8: Cantilever beam: results obtained after 1000 iterations with PSOAF2 for a) the worst run (run 7), and b) the best run (run 6).



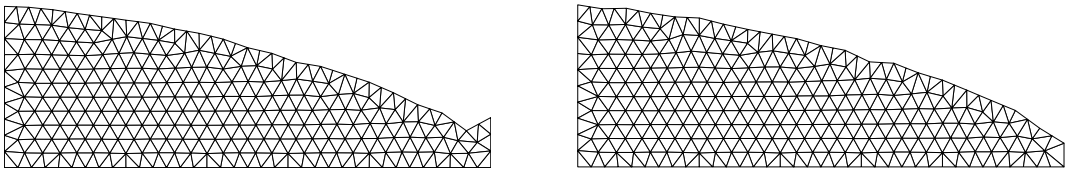
(a) (b)

Figure 5.9: Cantilever beam: results obtained after 100 iterations with PSOAF1* for a) the worst run (run 3), and b) the best run (run 4).



(a) (b)

Figure 5.10: Cantilever beam: results obtained after 500 iterations with PSOAF1* for a) the worst run (run 3), and b) the best run (run 4).



(a) (b)

Figure 5.11: Cantilever beam: results obtained after 1000 iterations with PSOAF1* for a) the worst run (run 3), and b) the best run (run 4).

5.5 Closure

In Chapters 3 and 4 it was demonstrated with a popular analytical test function set that the performance of PSOAF1 is poor. In turn, it was demonstrated that PSOAF2 and PSOAF1* demonstrate comparable performance.

In this chapter, the three different formulations of the PSOA were applied to a real engineering optimization problem, namely the optimal shape design of the cantilever beam. This was done using an unstructured remeshing optimization strategy, based on a truss structure analogy. Although the statistical significance of using only 10 runs at most allows for only general observations, it was again shown that PSOAF1 demonstrates poor performance. In turn, comparable performance for PSOAF1* and PSOAF2 was again demonstrated.

Chapter 6

Conclusions and recommendations

6.1 Conclusions

The main conclusions of this study are:

1. Implementation subtleties due to ambiguous notation have resulted in two distinctly different formulations of the PSOA, respectively denoted by PSOAF1 and PSOAF2 herein.

2. PSOAF1 as given by Eq. (3.1) is strictly *observer independent*, but the particle trajectories *collapse* to line searches.

3. PSOAF2 as given by Eq. (4.2) is *observer dependent*, although the particle trajectories are *space filling*.

4. Even though PSOAF2 is observer dependent, it outperforms PSOAF1 for both the rotated objective functions and unrotated objective functions (or decomposable multimodal objective functions).

5. A novel formulation, denoted PSOAF1*, was proposed, which is *observer independent*, while the particle trajectories are *space filling*.

6. The performance of PSOAF1* is comparable to the performance of PSOAF2 for the test function considered, with the added advantage of being objective.

7. The three different formulations were applied to a real engineering problem, namely the shape optimization of a cantilever beam.

8. For both the analytical test functions and the shape optimization problem, PSOAF1* and PSOAF2 outperformed PSOAF1, with the performance of PSOAF1* and PSOAF2 being comparable.

6.2 Recommendations

It is recommended that PSOAF1* is used in practice. Other formulations may of course also be formulated, but care should be taken to ensure that these formulations are objective and diverse, viz. the search trajectories should be space filling in n-dimensional space, and not collapse to line searches.

Secondly, when reporting on algorithms, it is of importance that the following points are reported upon, to ensure that research is reproducible:

1. General:
   (a) the selected velocity rule (using unambiguous notation),
   (b) the updating strategy (whether synchronous or asynchronous), and
   (c) the stopping criteria used.
2. The heuristics used (with a detailed description thereof):
   (a) minimum velocity restriction (and arithmetic precision),
   (b) maximum velocity restriction,
   (c) the implemented inertia strategy,
   (d) position restriction,
   (e) etc.
3. The parameters used:
   (a) the population size p,
   (b) the initial inertia and velocity values, and
   (c) the social and cognitive scaling factors c_1 and c_2.

Furthermore, in order to ensure robustness (objectivity) of an algorithm, decomposable functions should be used in both the rotated and the unrotated reference frames.

Bibliography

[1] J.A. Snyman. *Practical Mathematical Optimization*, volume 97 of *Applied Optimization*. Springer, New York, USA, 2005.

[2] R. Horst and P.M. Pardalos. *Handbook of Global Optimization*. Kluwer Academic Publishers, Dordrecht, Netherlands, 1995.

[3] J. Kennedy and R.C. Eberhart. Particle swarm optimization. In *Proc. IEEE Int. Conf. Neural Networks*, volume 4, pages 1942–1948, Perth, Australia, November 1995.

[4] R.C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proc. 6th Int. Symp. Micro Machine and Human Science*, pages 39–43, Nagoya, Japan, October 1995.

[5] J. Salerno. Using the particle swarm optimization technique to train a recurrent neural model. In *Proc. 9th IEEE Int. Conf. Tools with Artificial Intelligence*, pages 45–49, Newport Beach, CA, USA, November 1997.

[6] J. Kennedy and R.C. Eberhart. A discrete binary version of the particle swarm algorithm. In *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, volume 5, pages 4104–4108, Orlando, FL, USA, October 1997.

[7] P.J. Angeline. Using selection to improve particle swarm optimization. In *Proc. IEEE Int. Conf. Evolutionary Computation*, pages 84–89, Anchorage, AK, USA, May 1998.

[8] Y. Shi and R.C. Eberhart. A modified particle swarm optimizer. In *Proc. IEEE Int. Conf. Evolutionary Computation*, pages 69–73, Anchorage, AK, USA, May 1998.

[9] J. Kennedy and W.M. Spears. Matching algorithms to problems: an experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator. In *Proc. IEEE Int. Conf. Evolutionary Computation*, pages 78–83, Anchorage, AK, USA, May 1998.

[10] V. Tandon, H. El-Mounayri, and H. Kishawy. NC end milling optimization using evolutionary computation. *Int. J. Machine Tools and Manufacture*, 42(5):595–605, 2002.

[11] G. Ciuprina, D. Ioan, and I. Munteanu. Use of intelligent-particle swarm optimization in electromagnetics. *IEEE Trans. Magnetics*, 38(2):1037–1040, 2002.

[12] W.H. Slade, H.W. Ressom, M.T. Musavi, and R.L. Miller. Inversion of ocean color observations using particle swarm optimization. *IEEE Trans. Geoscience and Remote Sensing*, 42(9):1915–1923, 2004.

[13] E. Ozcan and C.K. Mohan. Particle swarm optimization: surfing the waves. In *Proc. IEEE Congr. Evolutionary Computation*, volume 3, pages 1939–1944, Washington D.C., USA, July 1999.

[14] M. Clerc and J. Kennedy. The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evolutionary Computation*, 6(1):58–73, 2002.

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

[16] Y.-L. Zheng, L.-H. Ma, L.-Y. Zhang, and J.-X. Qian. On the convergence analysis and parameter selection in particle swarm optimization. In *Proc. 2nd Int. Conf. Machine Learning and Cybernetics*, volume 3, pages 1802–1807, Xi-an, China, November 2003.

[17] P.-O. Persson and G. Strang. A simple mesh generator in Matlab. *SIAM Review*, 46(2):329–345, 2004.

[18] A.R. Cockshott and B.E. Hartman. Improving the fermentation medium for Echinocandin B production part II: Particle swarm optimization. *Process Biochemistry*, 36:661–669, 2001.

[19] B. Brandstätter and U. Baumgartner. Particle swarm optimization – mass-spring system analogon. *IEEE Trans. Magnetics*, 38(2):997–1000, 2002.

[20] U. Baumgartner, Ch. Magele, and W. Renhart. Pareto optimality and particle swarm optimization. *IEEE Trans. Magnetics*, 40(2):1172–1175, 2004.

[21] M.A. Abido. Optimal power flow using particle swarm optimization. *Int. J. Electrical Power and Energy Systems*, 24(7):563–571, 2002.

[22] H. Yoshida, Y. Fukuyama, S. Takayama, and Y. Nakanishi. A particle swarm optimization for reactive power and voltage control in electric power systems considering voltage security assessment. In *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, volume 6, pages 497–502, Tokyo, Japan, October 1999.

[23] P.C. Fourie and A.A. Groenwold. The particle swarm optimization algorithm in size and shape optimization. *Structural and Multidisciplinary Optimization*, 23(4):259–267, 2002.

[24] P.C. Fourie and A.A. Groenwold. The particle swarm algorithm in topology optimization. In *Proc. 4th World Congr. Structural and Multidisciplinary Optimization*, Dalian, China, May 2001. Paper no. 154.

[25] J. Kennedy. The particle swarm: social adaptation of knowledge. In *Proc. IEEE Int. Conf. Evolutionary Computation*, pages 303–308, Indianapolis, IN, USA, April 1997.

[26] P.N. Suganthan. Particle swarm optimiser with neighbourhood operator. In *Proc. IEEE Congr. Evolutionary Computation*, volume 3, pages 1958–1962, Washington D.C., USA, July 1999.

[27] Y. Shi and R.C. Eberhart. Empirical study of particle swarm optimization. In *Proc. IEEE Congr. Evolutionary Computation*, volume 3, pages 1945–1950, Washington D.C., USA, July 1999.

[28] Y. Shi and R.C. Eberhart. Fuzzy adaptive particle swarm optimization. In *Proc. IEEE Congr. Evolutionary Computation*, volume 1, pages 101–106, Seoul, Korea, May 2001.

[29] M. Clerc. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In *Proc. IEEE Congr. Evolutionary Computation*, volume 3, pages 1951–1957, Washington D.C., USA, July 1999.

[30] J. Robinson and Y. Rahmat-Samii. Particle swarm optimization in electromagnetics. *IEEE Trans. Antennas and Propagation*, 52(2):397–407, 2004.

[31] A. Carlisle and G. Dozier. An off-the-shelf PSO. In *Proc. Workshop on Particle Swarm Optimization*, Purdue School of Engineering and Technology, Indianapolis, IN, USA, April 2001.

[32] M.M. Noel and T.C. Jannett. Simulation of a new hybrid particle swarm optimization algorithm. In *Proc. 36th Southeastern Symp. System Theory*, pages 150–153, Atlanta, GA, USA, 2004.

[33] T. Aruldoss, A. Victoire, and A.E. Jeyakumar. Hybrid PSO-SQP for economic dispatch with valve-point effect. *Electric Power Systems Research*, 71(1):51–59, 2004.

[34] S. Baskar and P.N. Suganthan. A novel concurrent particle swarm optimization. In *Proc. IEEE Congr. Evolutionary Computation*, volume 1, pages 792–796, Portland, OR, USA, June 2004.

[35] R. Mendes, J. Kennedy, and J. Neves. The fully informed particle swarm: simpler, maybe better. *IEEE Trans. Evolutionary Computation*, 8(3):204–210, 2004.

[36] F.J. Solis and R.J.-B. Wets. Minimization by random search techniques. *Mathematics of Operations Research*, 6(1):19–30, 1981.

[37] K.A. de Jong. *An analysis of the behaviour of a class of genetic adaptive systems*. PhD thesis, Univ. Michigan, Ann Arbor, MI, USA, 1975.

[38] D. Beasley, D. R. Bull, and R. R. Martin. An overview of genetic algorithms: Part 1, Fundamentals. *University Computing*, 15(2):58–69, 1993.

[39] I. Rechenberg. *Evolutionsstrategie - optimierung technischer systeme nach prinzipien der biologischen evolution*. Frommann-Holzboog, Stuttgart, DE, 1973.

[40] R. Storn and K. Price. Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optimization*, 11(4):341–359, 1997.

[41] S. Yang, M. Wang, and L. Jiao. A quantum particle swarm optimization. In *Proc. IEEE Congr. Evolutionary Computation*, volume 1, pages 320–324, Portland, OR, USA, June 2004.

[42] T. Krink, J.S. Vesterstrøm, and J. Riget. Particle swarm optimisation with spatial particle extension. In *Proc. IEEE Cong. Evolutionary Computation*, volume 2, pages 1474–1479, Honolulu, HI, USA, May 2002.

[43] J.S. Vesterstrøm, J. Riget, and T. Krink. Division of labor in particle swarm optimisation. In *Proc. IEEE Cong. Evolutionary Computation*, volume 2, pages 1570–1575, Honolulu, HI, USA, May 2002.

[44] G. Venter and J. Sobieszczanski-Sobieski. Particle swarm optimization. *AIAA J.*, 41(8):1583–1589, 2003.

[45] B.R. Secrest and G.B. Lamont. Visualizing particle swarm optimization - Gaussian particle swarm optimization. In *Proc. IEEE Swarm Intelligence Symposium*, pages 198–204, Indianapolis, IN, USA, April 2003.

[46] D.N. Wilke and A.A. Groenwold. On constrained optimisation of structural systems using particle swarms. In *Proc. 5th World Congr. Structural and Multidisciplinary Optimization*, Lido di Jesolo, Italy, May 2003. Paper no. 84.

[47] Y. Shi and R.C. Eberhart. Parameter selection in particle swarm optimization. In *Evolutionary Programming VII: Proc. 7th Annual Conf. Evolutionary Programming*, pages 591–600, San Diego, CA, USA, March 1998.

[48] D.N. Wilke, S. Kok, and A.A. Groenwold. Particle swarm searches and adaptive unstructured meshing in shape optimization. In *Proc. 4th Int. Conf. Engineering Computational Technology*, Lisbon, Portugal, September 2004. Paper 97.

[49] J. Kennedy. Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In *Proc. IEEE Congr. Evolutionary Computation*, volume 3, pages 1931–1938, Washington D.C., USA, July 1999.

[50] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill Kogakusha, Ltd., Tokyo, Japan, 1965.

[51] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Trans. Evolutionary Computation*, 1(1):67–82, 1997.

[52] T.M. English. Evaluation of evolutionary and genetic optimizers: no free lunch. In *Evolutionary Programming V: Proc. 5th Annual Conf. Evolutionary Programming*, pages 163–169, San Diego, CA, USA, February 1996.

[53] G.A. Holzapfel. *Nonlinear Solid Mechanics*. John Wiley and Sons Ltd., Chichester, UK, 2001.

[54] R. Salomon. Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms. *BioSystems*, 39(3):263–278, 1996.

[55] R. Salomon. Some comments on evolutionary algorithm theory. *Evolutionary Computation*, 4(4):405–415, 1996.

[56] N. Metropolis and S. Ulam. The Monte Carlo method. *J. American Statistical Association*, 44(247):335–341, 1949.

[57] G. E. Mase and G. T. Thomas. *Continuum Mechanics for Engineers*. CRC Press, Boca Raton, FL, USA, 1992.

[58] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[59] E. Laporte and P. Le Tallec. *Numerical Methods in Sensitivity Analysis and Shape Optimization*. Modeling and Simulation in Science, Engineering and Technology. Birkhäuser, Boston, USA, 2003.

[60] M.J. Garcia and C.A. Gonzales. Shape optimization of continuum structures via evolution strategies and fixed grid finite element analysis. *Structural and Multidisciplinary Optimization*, 26(2):92–98, 2004.

[61] Q. Li, G.P. Steven, O.M. Querin, and Y.M. Xie. Evolutionary shape optimization for stress minimization. *Mechanics Research Communications*, 26(6):657–664, 1999.

[62] Y.M. Xie and G.P. Steven. A simple evolutionary procedure for structural optimization. *Computers and Structures*, 49(5):885–896, 1993.

[63] M.H. Imam. Three-dimensional shape optimization. *Int. J. Numerical Methods in Engineering*, 18(5):661–673, 1982.

[64] A.D. Belegundu and S.D. Rajan. A shape optimization approach based on natural design variables and shape functions. *Computer Methods in Applied Mechanics and Engineering*, 66(1):87–106, 1988.

[65] G. Allaire, F. Jouve, and A.-M. Toader. Structural optimization using sensitivity analysis and a level-set method. *J. Computational Physics*, 194(1):363–393, 2004.

[66] C. Mattheck and S. Burkhardt. A new method of structural shape optimization based on biological growth. *Int. J. Fatigue*, 12(3):185–190, 1990.

[67] H. Edelsbrunner. *Geometry and Topology for Mesh Generation*. Number 7 in Cambridge Monographs on Applied and Computational Mathematics. Cambridge Univ. Press, Cambridge, UK, 2001.