An Evaluation of Performance Enhancements to Particle Swarm Optimisation on Real-World Data

Thesis

An Evaluation

of

Performance Enhancements

to

Particle Swarm Optimisation

on

Real-World Data

IAN KENNY, BSc(Hons)

Submitted in partial completion of the requirements for the degree of

Doctor of Philosophy in Computing.

July 2016

Dedication

To my Dad who sadly didn’t live to see me complete this thesis.

Acknowledgements

I would like to thank the following people for the help and support they provided while I was researching this thesis:

- My family, especially my Mum without her support in audio recording the academic papers, which I needed to read, I would not been able to complete this thesis.
- My supervisors, for their encouragement and support, without which this thesis would have not been completed.
- Nick Mount, School of Geography, University of Nottingham, for his support and assistance in providing and helping me to understand the hydrographical and glacier data used in Chapter 5 and Chapter 6. His assistance was invaluable and is gratefully acknowledged.

Table of Contents

Table of Figures

Abstract

Swarm Computation[1] is a relatively new optimisation paradigm. The basic premise is to model the collective behaviour of self-organised natural phenomena such as swarms, flocks and shoals, in order to solve optimisation problems. Particle Swarm Optimisation (PSO) is a type of swarm computation inspired by bird flocks or swarms of bees by modelling their collective social influence as they search for optimal solutions.

In many real-world applications of PSO, the algorithm is used as a data pre-processor for a neural network or similar post processing system, and is often extensively modified to suit the application. The thesis introduces techniques that allow unmodified PSO to be applied successfully to a range of problems, specifically three extensions to the basic PSO algorithm: solving optimisation problems by training a hyperspatial matrix, using a hierarchy of swarms to coordinate optimisation on several data sets simultaneously, and dynamic neighbourhood selection in swarms.

Rather than working directly with candidate solutions to an optimisation problem, the PSO algorithm is adapted to train a matrix of weights, to produce a solution to the problem from the inputs. The search space is abstracted from the problem data.

A single PSO swarm optimises a single data set and has difficulties where the data set comprises disjoint parts (such as time series data for different days). To address this problem, we introduce a hierarchy of swarms, where each child swarm optimises one section of the data set whose gbest particle is a member of the swarm above in the hierarchy. The parent swarm(s) coordinate their children and encourage more exploration of the solution space. We show that

[1] Swarm Computation or Swarm Computing is often called Swarm Intelligence however, my preferred term is Swarm Computation.

hierarchical swarms of this type perform better than single swarm PSO optimisers on the disjoint data sets used.

PSO relies on interaction between particles within a neighbourhood to find good solutions. In many PSO variants, possible interactions are arbitrary and fixed on initialisation. Our third contribution is a dynamic neighbourhood selection: particles can modify their neighbourhood, based on the success of the candidate neighbour particle. As PSO is intended to reflect the social interaction of agents, this change significantly increases the ability of the swarm to find optimal solutions. Applied to real-world medical and cosmological data, this modification is and shows improvements over standard PSO approaches with fixed neighbourhoods.

Chapter 1

Introduction

Zee: *The* whole system *makes me* feel*… insignificant.*

Psychologist: *Excellent. You've made a real breakthrough.*

Zee: *I have?*

Psychologist: *Yes, Zee. You ARE insignificant.*

(Antz 1998)

This chapter begins with a brief introduction to the paradigm of biologically inspired computation. Combinatorial optimisation is then discussed as the branch of mathematics to which biologically inspired computation may be applied, and goes on to detail the research question considered in this thesis and the contributions made. The chapter concludes with an outline of the thesis.

1.1 Biologically Inspired Computation

Biologically inspired computation involves borrowing ideas from the natural world in order to carry out some optimisation task. On its own, a single ant in a colony is insignificant, but the colony as a whole may demonstrate useful emergent behaviour. Although emergence as a phenomenon had been studied for some time, its first general introduction to the wider research community was by Holland (2000), work which led to the development of genetic algorithms. The concept of emergent behaviour emerging from groups or crowds has been elucidated by Johnson (2002) and more recently Strogatz (2004). There has also been an upsurge in the study of emergent behaviour of insect societies, influenced by the original studies of Wilson (1971) and Hölldobler & Wilson (2008).

Within the computing research community, there has been significant growth in research focusing on the development of heuristics and methodologies inspired by biological phenomena. This has proved to be a strong basis for the development of new heuristics, given that nature has already found solutions to search problems such as finding food or burying corpses. Biologically inspired computational techniques cover a diverse set of heuristics: Corne et al. (1999) and de Castro & Von Zuben (2004) offer a good introduction to the topic.

Within biologically inspired computation, the class of heuristics which model collective self-organisational behaviour is generally referred to as Swarm Intelligence. The recognition that biological organisms frequently rely on stigmergy to influence one another's behaviour has led to the development of a group of heuristics which attempt to simulate stigmergic interactions in order to address combinatorial optimisation problems. The term *stigmergy* was introduced by French biologist Pierre-Paul Grassé, quoted in Wilson (1975, 2000), and originally referred to the spontaneous indirect coordination between agents, through influences on the environment which subsequently affect the actions of another agent or agents. The concept has subsequently been extended to include the notion that social organisms can influence each other's behaviour through their previous actions (Bonabeau & Théraulaz, 2000; Bonabeau, 1999). By extension, the stimulation of a secondary behaviour via the reward received from another agent can also be considered as stigmergy. For example, in an ant colony, a returning forager may lay a pheromone trail that will stimulate nest mates to follow the trail to find more food. From these kinds of interaction between multiple agents, rather than from a top-down organisational structure, collective emergent behaviour appears. This concept is important when we consider how PSO interacts with the landscape of the problem being solved; it allows the particle swarm to move to areas of improved solutions based on information from other particles.

Swarm computation is a radical departure from the classical way of organising computation, in which processes are centrally controlled. Rather, Swarm Computation is a means of computing solutions to certain problems through self-organisation and emergence. We shall therefore use the term throughout this thesis.

1.2 Combinatorial Optimisation

We will now discuss the concept of combinatorial optimisation in some detail, in order to establish a context within which Swarm Computation is useful. Combinatorial optimisation is one of the most important areas of applied computation, applying to a significant set of real-world problems. A combinatorial problem is one in which the aim is to *optimise*, or find the best solution for, a *combination* of values for a set of variables. For these types of problems there is no direct means of determining a solution within a realistic time, although the validity and "correctness" or otherwise of a given solution can be easily determined Macready & Wolpert (1995). The problem to be optimised is described in terms of parameters to an *objective function*, which a feasible solution *minimises* or *maximises*. Theoretically an almost universal optimiser is possible English (2000), however, the No Free Lunch (NFL) theorem Wolpert & Macready, (1995, 1997) proves that no algorithm will perform better than any other over the set of all possible computationally accessible problems. Many heuristics for solving combinatorial problems aim to arrive at an optimal solution by eliminating unsuitable solutions by one means or another. A real-world example of such a problem would be the optimal route for a supermarket delivery van to take, given the timed deliveries required by customers, the minimum number of vans required, the maximum fuel consumption considerations and other constraints relevant to the home delivery operation. This can be seen as a special case of the travelling salesman problem Michalewicz (1996), a PSO implementation of which is given by Zhi et al. (2004).

A useful way of conceptualising the process of finding a solution to combinatorial optimisation problems is to develop the concepts of *solution space* and *fitness landscape*. A *solution space* is the *n*-dimensional space of all possible values of the *n* representative variables that meet the constraints of the problem. A *Fitness landscape* is a hyperplane within the *n+1*-dimensional space formed by the solution space and a fitness dimension. An optimisation algorithm traverses the fitness landscape to find the high (or low) points that represent optimal solutions.

1.3 Research Question and Rationale

This thesis concentrates on improvements to one heuristic within Swarm Computation, namely Particle Swarm Optimisation (PSO). The steps that taken were as follows:

1. A technique was developed in which the parameters used in the objective function are discovered during the optimisation process; this proved particularly useful when optimising highly non-linear systems.

2. Current PSO implementations generally depend on a single layer swarm of particles, meaning that information gathered during the course of the search of the solution space is lost, rather than being made subject to later processing. A scalable robust extension to PSO was developed, in which the best candidate solutions[2] for each subset are determined by a separate swarm within a hierarchy, with swarms at each level of hierarchy maintaining their own objective function. If each level of the hierarchy were to use exactly the same objective function then some of the advantage of using a hierarchy would be lost; indeed some of the information gathered at low levels would not be utilised. This research will demonstrate that a hierarchy of swarms can learn

[2] Candidate Solution: a member of a population of possible solutions that satisfy all constraints.

from each other's experience, and therefore direct the learning process, thereby improving overall performance. By this method, it is possible to demonstrate the application of particle swarm optimisation to noisy, dynamic and multiple optima datasets. The model respects the intrinsic relationships between the data subsets, increases diversity and encourages the spread of information across the solution space. The combined information is used to guide the collective swarm effectively to a better solution, improving the performance of all swarms. This intrinsic strength is little explored by much of the early literature, for example, van den Bergh (2002), Engelbrecht et al. (2005), Franken & Engelbrecht (2004), Franken & Engelbrecht (2005), Angeline (1998b), Bo Liu et al. (2006), Kennedy (2005) and Higashi & Iba (2003). More recent papers focusing on real-world datan – for example, Wen-Tsai Sung et al. (2014), Li et al. (2013), Peng et al. (2014), Siano & Mokryani (2013), Chen et al. (2013) – tend to respect the velocity equation whilst maintaining good results; this is achieved principally through abstraction in the search space. The hierarchical extension is applied to problems with multiple, related data subsets of time series data which are processed simultaneously.

3. A further extension to this model is introduced, based on models of human social interaction, inspired by Watts & Strogatz (1998) and Watts et al. (2005), in which particles are able to recruit and retire members of its neighbourhood, or social group, dependent on the member's value to the particle.

The research questions associated with this work are as follows:

- Can a combinatorial optimisation problem be modelled for PSO without the use of pre-processing or prior knowledge of the characteristics of the solution space like this extension represented by the dataset?

- Is it possible to develop the PSO heuristic as a tool for optimising solutions for complex problems, through the abstraction of the heuristic from the problem space and objective function?

- Can PSO be extended to simultaneously process multiple, related datasets, with complex interconnected relationships between variables, and the influence between the variables varying across their full range of values?

1.4 Contributions

The main contributions to knowledge are:

- The application of PSO to a complex real-world problem using a straightforward transfer function which maps the input space to the output, and the main parameters of this function are discovered during the optimisation process, avoiding problem specific assumptions. This concept is developed in Section 0 and applied to real world data in Chapter 5.

- Multiple swarms: The use of multiple swarms to optimise multiple distinct parts of a data set, such as disjoint sections of time series data from the same system. Each swarm separately optimises a predictor for each part, with the results coordinated by a parent swarm. The overall best predictor is found by the parent swarm, and the child swarms communicate indirectly with each other through this parent swarm. The approach is described in Section 0 and applied to real world data in Chapter 6.

- An extended neighbourhood topology model which allows neighbourhood interactions to be influenced dynamically by the contributions made by individual particles to the success of another particle. This demonstrates an improvement in the quality of solutions achieved over the standard neighbourhood topologies tested. This concept is discussed in Section 0 and applied to real-world data in Chapter 7.

- The application of both of these techniques to noisy data (Sneyers, 1997), demonstrating an amendment to PSO which improves the capability of the heuristic over the single swarm version, when applied to complex real-world problems that include noisy data, along with an improvement over empirical results using the same data.

1.5 Thesis Outline

Chapter 2 presents the history of PSO's development, initially as a social technology and then as an optimisation technique. Each section considers significant research within the development of PSO. This presents a foundation for our understanding, and for our research topic. The chapter also contains a major subsection on neighbourhood topologies, the context for the work presented in Chapter 7. Chapter 2 ends with a discussion of computational optimisation in the context of evolutionary based optimisation techniques

Chapter 3 extends the review in Chapter 2 to offer a taxonomy and critique of recent developments of PSO including ad hoc approaches and extensions to the basic PSO approach Chapter 3 considers a number of problems with the early research.

Chapter 4 begins with a principled analysis of the stages in the convergence of the PSO algorithm, as different aspects of the velocity equation become the dominant influence on the velocity; the role of velocity in the convergence on optima; and some of the consequences that arise from the formulation of the velocity equation. Based on these analyses, three innovations to standard PSO are proposed in this chapter, establishing the foundation for the experiments proposed in later chapters.

Chapter 5 establishes a predictive model for the River Severn hydrographical flow using real-world environmental variables as inputs. The aim is to establish a generalised model which is able to predict chaotic systems – in this case hydrographical flow – with improved accuracy over existing techniques.

Chapter 6 presents and develops the hierarchical extension to PSO, based on the generalised technique presented in Chapter 5, which allows it to process multiple data subsets simultaneously, allowing independent processing of all those datasets and improving the overall predictive accuracy of the derived model. The chapter demonstrates that the proposed hierarchal swarm is a significant development of existing PSO models, which maintains the swarm's coherence and maintains the implicit feedback from the underlying data, guiding the search – an intrinsic strength of PSO.

Chapter 7 presents the new neighbourhood topology for PSO. Within this topology, a particle's neighbourhood can change dynamically, dependent on the relative value the particle places on the contributions made by other particles in its neighbourhood. The intuitive idea behind this innovation is that a particle's neighbourhood should be affected by the contribution made by the interaction with its neighbours to the improvement of the particle's candidate solution. The aim is to produce a more dynamic neighbourhood which is more reflective of social interaction between the particles.

Chapter 8 presents a summary of the research on the various datasets used. For each of these datasets we summarise the contributions our performance enhancements make to PSO. The chapter summarises the issues we identify and contributions to knowledge made.

Chapter 2

A Brief History of Particle Swarm Optimisation

This chapter provides a history of Particle Swarm Optimisation (PSO) starting with its formulation as a social interaction technique and its transition into an optimisation technique. The chapter is divided into sections that consider various stages in the development of PSO. Each section considers significant papers within the PSO research being considered. The chapter starts with an introduction to the foundations of PSO and goes on to consider in section 0 the major heuristic developments, establishing the foundations of our research topic.

Section 0 discusses developments of the PSO model that are significant from the point of view of the research presented in later chapters, including a major subsection on neighbourhood topologies. These can have a significant influence on the effectiveness or PSO as an optimisation technique. Section 0 considers PSO's application to dynamic and multi-objective optimisation problems and section 0 discusses the application of multi-swarms to these.

The chapter concludes with an observation on swarm behaviour in section 2.7, a discussion of PSO's applications to real-world problems in section 2.8, and finally a brief introduction to other optimisation techniques in section 2.9. These sections provide an introduction to computational optimisation in the context of evolutionary based optimisation techniques, in order to set the scene for Chapter 3 that looks at some of the weaknesses of the early PSO research.

Introduction

Particle Swarm Optimisation operates as a model of social influence, with each particle representing a potential solution to the problem being solved. The particles are ‘flown’ through the solution space under the influence of their previous best solution and the best solution of the population. The best solution is determined by a problem-specific objective function, which returns a measure of the fitness of a candidate solution.

The problem space is visualised as an abstract landscape, with hills and valleys, and with the height of any point in the space representing how good the solution at that point is, referred to as that solution’s *fitness*. In PSO, a number or particles are distributed at random across this landscape, as illustrated in Figure 1. In the basic version of PSO, each particle has a *neighbourhood*, and can exchange information with other particles in this neighbourhood. The structure of these neighbourhoods is termed the *topology* of the swarm and is generally fixed when the swarm is created. Each particle will have the following characteristics:

- a *current position*, represented by a vector $\mathbf{x}_i$;
- a *velocity* (since particles move), represented by a vector $\mathbf{v}_i$;
- a vector recording the position in the solution space at which it achieved its fittest result (its *personal best*, $\mathbf{p}_i$);
- a vector recording the position $\mathbf{p}_g$ of the particle in its neighbourhood with best fitness.

The particle marked **gBest** in Figure 1 is closest to the fittest solution in the whole landscape, and is marked as the global best.

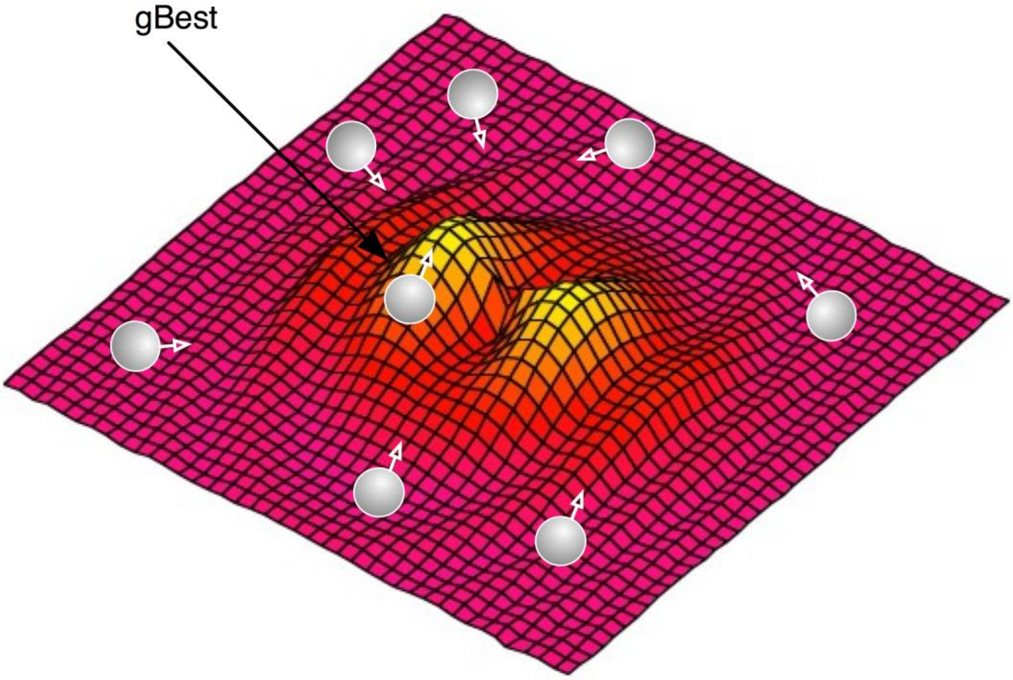


Figure 1: Particles moving in solution space

The PSO system evolves across a number of time steps or iterations, with particles moving across the landscape seeking out better and better solutions. At each time step:

- every particle compares the fitness of the possible solution its position vector **x** represents with those of the other particles in its neighbourhood;

- the best particle then becomes the neighbourhood's best, and the $\mathbf{p}_g$ of all particles in the neighbourhood are updated accordingly.

The other particles in the neighbourhood are thus pulled towards the best particle and the current particle's previous best in the following iteration. See section 0 for a more detailed discussion of neighbourhood topologies.

In early papers, typical particle populations used were up to 40 particles Eberhart & Shi (2001). However, there remains a wide range in the number of particles used in the literature, ranging from 10 to 100 or alternatively a population number which relate to the number dimensions in the problem being solved, for example, Hu & Tan (2015), Munlin & Anantathanavit (2015), Mandal & Si (2015). There are no established criteria for determining the number of particles – simply, that the more particles there are, the more opportunities are available to discover better solutions per iteration, balanced against the computational cost. In our experiments, we maintain a constant population of 40 particles per swarm. The following is a commonly-used version of the equations which govern the change in direction of the particle at each step of its flight (Kennedy & Eberhart, 1995; Kennedy et al., 2001):

$$\vec{v}_i = \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i) \qquad \text{(a)}$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}_i \qquad \text{(b)}$$

where $\vec{p}_i$ is the particle’s previous best, and $\vec{p}_g$ is the neighbourhood best, $\vec{v}'_i$ is the particle’s velocity, $\vec{x}_i$ is the particle's current position, φ_{1i} and, φ_{2i}, are random values, conventionally in the range 1.4 to 1.9. (However, refer to the constriction coefficient and GCPSO sections below.) The subscript i is often omitted from φ_{1i} and φ_{2i}.

```
Initialize particles to uniformly random positions and velocities to zero

Repeat

    Calculate particle fitness, if better than previous best change particles previous best

    Set best particle as global best

    For each particle do

        Calculate particle's velocity according to equation (a) above

        Update particle's position according to equation (b) above

    End Do

Until Termination condition met
```

Figure 2: Pseudo code for Particle Swarm Optimisation

An algorithm for PSO is given in Figure 2. When run, the effect is that the particles converge towards the best particle in the population. Once per iteration, the solution found by each particle is compared with the solutions of the other particles in its neighbourhood. (As an aside, it should be noted that when PSO was developed. It lacked the concept of a neighbourhood, hence $\vec{p}_g$ being retained to refer to neighbourhood best.) The best particle then becomes known as the neighbourhood's best, or *gbest*, and the other particles in the neighbourhood are then pulled towards this best particle in the following iteration, using:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i) \quad \text{(a)}$$

$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i \quad \text{(b)}$$

This equation is also known as the inertia weight variant Shi & Eberhart (1998b). The w parameter acts as an inertia weight Shi & Eberhart (1998b). In the original form of the equations, the whole of the previous velocity was included without being damped. The original algorithm by Kennedy & Eberhart (1995) was even simpler than this: rather than coefficients, there were arbitrary cut-off points implemented in the algorithm programmatically as IF-THEN logic decision rules, with a simple greater than or less than decision made to pull the particle towards *gbest* (Kennedy et al., 2001).

A further observation is made if the elements of the equation are separated as follows:

$$\varphi_1 \cdot (\vec{p}_i - \vec{x}_i) \qquad \text{(a)}$$
$$\varphi_2 \cdot (\vec{p}_g - \vec{x}_i) \qquad \text{(b)}$$

In the equation above, part (a) is considered to be the cognitive element and part (b) the social element. This reflects the respective influences of a particle's previous best solution and the best solution of the population.

The algorithm also implements a V_{max} parameter, implemented as the following test after the update of the velocity for each dimension per iteration:

$$-V_{max} < x_{id} > V_{max}$$

where i is the particle's index, d is the dimension and V_{max} is a heuristic specific parameter often set to 4.0. In the original form of the heuristic, this acted as a damping constant to prevent the swarm exploding due to a greater step size in the velocity Eberhart & Shi (2000). In our research we continue to use a value of 4.0, which is consistent with van den Bergh (2002), to ensure that the swarm converges. In our early experiments we found that setting V_{max} to half the dynamic range, as suggested in Shi & Eberhart (1998a), restricted the exploration potential.

The original PSO heuristic was developed as a model of social influence, but was quickly realised to have potential application in optimisation. However, the original PSO had some shortcomings (Kennedy, 2004; Wen-Jun & Xiao-Feng, 2003), mainly that the swarm converges too quickly, making it susceptible to local optima, and that the swarm stagnates before reaching maximum optimality.

We shall now consider the main adaptations to the heuristic which have attempted to address these problems.

2.1.1 A Comment on Notation

Throughout this thesis we shall use the notation used by Shi & Eberhart (1998b) and van den Bergh (2002) when describing the particle swarm equations, in order to remain consistent with the historical development of PSO. However, it should be noted that the notation used can be misleading, not least because it implies certain assumptions about the nature of the parameters φ_1 and φ_2 . In the traditional notation, these are represented as coefficients, whereas in fact they are implemented as vectors, consistent with Poli et al. (2007).

Major Heuristic Developments

There have been many major developments of the original heuristic. The reason for this is that PSO was originally developed to model social influence and emergent behaviour, subsequently undergoing developments for application as an optimiser. Briefly, basic PSO is susceptible to local optima, especially in a uniform solution space, where the solutions are predominately suboptimal and there is no general direction of improvement. Therefore PSO is susceptible to an occurrence characterised as First Order Deception (FOD) Blum & Dorigo (2005).

FOD is characterised by a solution space with multiple areas of stable attraction or a plateau containing the globally optimal solution. During the early stages of convergence, PSO is less susceptible to FOD because any particle can become the attractor, the particle having the best solution so far drawing the other particles towards it. During the latter stage of the search the algorithm is prone to stagnation, meaning that it reaches a point where it fails to make further improvements, although further improvements are still possible, resulting in FOD Blum & Dorigo (2005). The remainder of this section considers the most significant developments of PSO, and the effect they have on the algorithm's performance.

2.1.2 Inertia weight

In the original version of the PSO heuristic, the V_{max} parameter was introduced in order to prevent explosion, which, in turn, prevented convergence; this parameter is applied after the velocity equation and restricts the velocity to. $\pm V_{max}$. However, even with the addition of the V_{max} parameter, it has been reported Shi & Eberhart (1998b) that the heuristic also suffered from poor convergence towards the optima during the latter stages of the algorithm's run (see also Clerc & Kennedy, 2002, and van den Bergh, 2002). This was believed to be because, at this point, particles which were not the global best were effectively oscillating between their previous best, and the best of the population, rather than being 'attracted' to the global best. This can be predicted by examining the two elements of the velocity update equation and the acceleration coefficients: if the particle's best position in the solution space is further from the particle's current position than that of the current *gbest*, then, assuming equality of the acceleration coefficients, the cognitive part of the equation will on average have greater influence over the particle's trajectory as it approaches the *gbest* position.

In order to improve this behaviour Shi & Eberhart (1998b) inserted an inertia weight into the equation to moderate the influence of the particle's previous velocity; this is shown below.

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i)$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i$$

The additional element w, introduces a dampener on the propensity for the particle to continue to travel in the direction in which it was travelling, in effect causing it to explore its immediate area a little farther and reducing the particle's inertia. Shi & Eberhart also varied the value of w, starting at 0.9 and descending to 0.4 over the course of the algorithm's execution lifetime. This improved the algorithm's performance over that of the original version of the PSO heuristic. However, subsequent studies, for example Ismail & Engelbrecht (1999), Higashi & Iba (2003), and Kennedy & Mendes (2003) have shown that, although this version is competitive with other heuristics, its convergence potential is not as great as other optimisation techniques – for example that of Genetic Algorithms (Goldberg, 1989; Michalewicz, 1996) and Evolutionary Algorithms (Corne et al., 1999). Further analysis demonstrated that the suggested parameter values do not induce convergent behaviour van den Bergh (2002).

2.1.3 Constriction coefficient

Another development of the original velocity equation is the use of a constriction coefficient Clerc & Kennedy (2002). Clerc and Kennedy carried out extensive research, evaluating different forms of constriction coefficient (also known as constriction factor). The authors' proposal changes the velocity equation as follows:

$$\vec{v}'_i = \chi(\vec{v}_i + c_1 \cdot (\vec{p}_i - \vec{x}_i) + c_2 \cdot (\vec{p}_g - \vec{x}_i))$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i$$

In this version of the equation, χ constricts the whole of the new velocity. The calculation for χ is given in Clerc & Kennedy (2002) and in Eberhart & Shi (2000). The latter is reproduced below:

$$\chi = \frac{2}{\left|2-\varphi-\sqrt{\varphi^2-4\varphi}\right|}, \text{where} \varphi = c_1 + c_2, \varphi > 4$$

Essentially the constriction coefficient's purpose is to restrict the maximum step size a particle can take on each dimension, for a given iteration. Clerc and Kennedy also proposed removing the V_{max} parameter, effectively allowing a continuous solution space, with the constriction coefficient acting as an inertia weight for the whole of the new velocity. Such a modification appeared to produce improved convergence over the inertia weight version of the equation in Eberhart & Shi; see van den Bergh (2002) for a detailed discussion on the parameter settings required to make the inertia weight variant equivalent to the variant suggested by Clerc & Kennedy (2002). Clerc & Kennedy also suggest that V_{max} should be equal to X_{max} for all dimensions in X , thereby enforcing a boundary constraint. The X_{max} parameter is defined as the maximum value a particular dimension can take and still be valid within a solution; in practice, X_{max} is usually implemented as $\pm X_{max}$, thereby constraining positive and negative values for each dimension.

Empirical results in Shi & Eberhart (1998a) and Higashi & Iba (2003) provide evidence that such an improvement is inevitable, purely because it restricts the effect of both the social and cognitive parts of the equation, thereby moderating the step size in accordance with the combined value of the acceleration coefficients given by φ . It does not, however, affect the convergent properties enforced through the constriction coefficient, but merely speeds convergence.

The introduction of the constriction coefficient defines a formalised relationship between the velocity equation parameters. The dependence on the problem-specific parameter V_{max} is significantly reduced. One trend has been for using the V_{max} parameter as a boundary constraint, rather than a means of controlling particle explosion (Wachowiak et al., 2004; Ratnaweera et al., 2004, as suggested by Eberhart & Shi, 2000).

2.1.4 Guaranteed Convergence Particle Swarm Optimisation (GCPSO)

Guaranteed Convergence Particle Swarm Optimisation (GCPSO) was a development by van den Bergh (2002), to produce a globally convergent variant of PSO. He argues, with the use of mathematical proof, that the standard PSO heuristic velocity equation is neither a global nor a local optimiser. This finding holds whichever implementation of the velocity equation is used: inertia weight, constriction coefficient or its original form. This realisation resulted in the development of the GCPSO variant. The characteristics of GCPSO follow from the realisation that when a particle becomes the global best it then ceases to contribute improving solutions to the swarm; and because other members of the swarm are drawn to that point, there is no guarantee of any further improvement. Such an improvement would depend on a better region existing between the current global best and the current position of all other particles. Therefore, in order to become a global optimisation technique, a randomisation or reset element needs to be added. Hence, GCPSO was developed, which implements a concept that changes the velocity equation for the *gbest* particle such that stochastic changes are made to the position within a hypercube space of the current position. If an improved solution is not discovered the size of the hypercube is increased for the next iteration (van den Bergh, 2002; van den Bergh & Engelbrecht, 2004).

An interesting implementation of GCPSO is presented by Messerschmidt & Engelbrecht (2004), in which GCPSO is used to train neural network weights to learn to play the game tictac-toe. In the paper they compare the *lbest* and *gbest* versions of PSO, with the new GCPSO version. In their experiments GCPSO significantly outperforms the other variants, when implemented in the *lbest* configuration.

2.1.5 Time-Varying Acceleration Coefficients with Self Organising Hierarchies and Mutation

Another variant of the original PSO velocity equation is Time-Varying Acceleration Coefficients with Self-Organising Hierarchies and Mutation (Ratnaweera et al., 2004). The proposal is to vary the acceleration coefficients in the velocity equation over the lifetime of a run, with a view to increasing the swarm's convergence potential during the latter stages. Their suggestion is to increase the coefficient controlling the cognitive component, while decreasing the coefficient controlling the social component, leading to growing influence of the particle's own previous experience. This paper also introduces the mutation operator. However, we suggest that the more significant development is the variation in the acceleration coefficients since it is the first time such a method has been applied. Ratnaweera et al. (2004) report that varying the coefficients showed significant improvement over the version of PSO with the inertia weights. The method demonstrated significant potential by applying a fairly simple change to the existing velocity equation: they call this PSO-TVAC. Another variant Ratnaweera et al. (2004), a hierarchic variant that is different from our own hierarchical variant which we present in Chapter 6, removes the previous velocity from the new velocity equation, and when the velocity of a particle equals zero it is reinitialised with a random velocity proportional to. V_{max} . The results presented show that the proportion of V_{max} required to improve performance is very dependent on the test function landscape, requiring prior knowledge of the problem to be solved.

Moreover, the mutation operator (Ratnaweera et al., 2004) devised is a less effective strategy on the same test functions, and additionally has higher computation costs, than the mutation operators presented in previous work (e.g., Angeline, 1998b; Jing Liu et al., 2005; Cui-Ru Wang et al., 2005; Junfeng Chen et al., 2006; and Ning Li et al., 2004). This is because each mutation operation requires that the rate of change for each dimension is calculated for each particle, regardless of whether a subsequent mutation is applied to the value of a dimension. Whilst this method of mutation seems a useful strategy, it is computationally expensive and – the results demonstrate – less effective, when compared with the hierarchical variant also presented in Ratnaweera et al. (2004).

2.1.6 Other Hierarchical PSO Implementations

There are several hierarchical swarm implementations in the literature, in addition to Ratnaweera et al. (2004) discussed earlier. These include: Janson & Middendorf (2005), Janson & Middendorf (2003), Lin Lu et al. (2008), Rezaei & Azadi (2009), Scheutz (2007), Yen & Wen-Fung Leong (2009) and Weibo Wang & Quanyuan Feng (2010), Senthilnath et al. (2012), Po-Hung Chen (2012). We briefly discuss the two of the Janson & Middendorf papers here. These papers develop a hierarchical model of PSO in which the better particles move up a level in the hierarchy. Janson & Middendorf (2005) also introduce a variation to the velocity equation such that the *gbest* particle is implemented with a randomised search within the local area after it has failed to find an improved solution for a period of time.

Lin Lu et al. (2008) implement the transfer of complete particle position vectors, both between layers and between swarms on the same layer. Although facilitating information transfer, the experiments reported in Chapter 6 show this results in premature convergence: rather than perturbing the particle's position: the best particles are accelerated to stagnation through the duplicated action of the velocity equation at each layer. Rezaei &

Azadi (2009) present a novel variant and application that of MRI scan alignment for the surgical identification of brain tumours. The variant is based on Janson & Middendorf (2005), and is a variation on a single swarm which works through implementing a hierarchical neighbourhood structure in which each particle changes position in the hierarchy, dependent on its performance in the previous iteration. One potential disadvantage of this strategy is that the worst performing particles are moved away from the best performing particle. This in turn leads to a situation where the worst performing particles are directly influenced by particles with similar performance scores, and only indirectly influenced, through intermediate layers, by better performing particles. Scheutz (2007) also presents a real-time application for facial recognition also based on Janson & Middendorf. The paper reports significant improvements over single swarm facial recognition, without the reliance on the “Haar” facial detector technique used in Janson & Middendorf. Unfortunately, there is insufficient implementation detail for conclusions to be drawn on the reason for such an improvement, or whether the improvement is application specific. Yen & Wen-Fung Leong (2009) implements a dynamic multiple swarm implementation, which aims to optimise a multi-objective problem using an adaptive swarm scenario where the number of swarms increase and reduce adaptively to the function landscape, in accordance with the swarms’ progress. Each swarm is designated a search area to explore, depending on the success of another swarm and allowing the combined swarms to exploit the search areas. Conversely, the number of swarms is reduced if an area proves to be unproductive. Weibo Wang & Quanyuan Feng (2010) implement a multi-swarm hierarchal structure where several bests are selected per iteration from the lower layer swarms to form the higher level swarm. Each particle in the higher layer swarm is then perturbed using a chaotic search within a certain radius. This is analogous to van den Bergh’s single swarm GCPSO, van den Bergh (2002), the modification here applied to the *gbest* particle attracts the criticism that it ignores the solution space information and becomes a stochastic optimiser.

Although Po-Hung Chen (2012) refers to a hierarchical approach, it is essentially a two-phased approach with a hybrid expert system-PSO solution to solve a unit commitment (UC) problem. The first phase of this creates candidate solutions which satisfy the rules of the expert system, with a second phase that optimises these candidate solutions using a variant of PSO which Po-Hung Chen refers to as Elite-PSO. Elite-PSO is implemented such that each particle has a group of best positions which can be learned from, rather than a single best, these positions being the historical best positions for the particle, thus implementing a memory with the potential to lead back to a previous best position. An interesting aspect of the implementation is that, rather than having vectors, the particle swarms are implemented as two-dimensional matrices.

For completeness, we briefly list some other hierarchical swarm research that we consider relevant: Bhattacharya & Bhattacharyya (2012); Xue Wang & Sheng Wang (2011); Senthilnath et al. (2012); Cheng-Chi Wu et al. (2013).

2.1.7 Comprehensive Learning Particle Swarm Optimizer (CLPSO)

Comprehensive Learning Particle Swarm Optimizer (CLPSO) (Liang et al., 2006) offers an innovation based on the concept that any particle should be able to learn from any other particle for each dimension. This is achieved by modifying the velocity equation as follows:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_{f_i(d)} - \vec{x}_i)$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i$$

where f defines which particle's *pbests* a particle should follow, the process for updating a particle's position is defined in Figure 3:

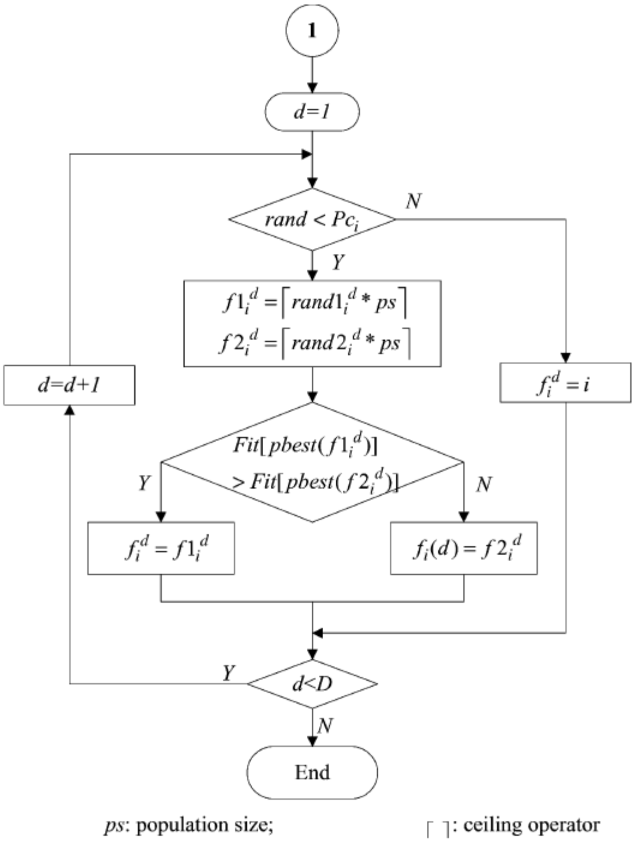

1
d=1
rand < Pc_i
N
Y
$f1_i^d = \lceil rand1_i^d * ps \rceil$
$f2_i^d = \lceil rand2_i^d * ps \rceil$
d=d+1
$f_i^d = i$
$Fit[pbest(f1_i^d)] > Fit[pbest(f2_i^d)]$
Y
N
$f_i^d = f1_i^d$
$f_i(d) = f2_i^d$
Y
d<D
N
End
ps: population size;
⌈ ⌉: ceiling operator

Figure 3: Selection of examplar dimensions for particle *i* reproduced from Liang et al. (2006)

CLPSO is tested using several standard test functions and against several different standard variants of PSO. The test functions are split into four different groups, Groups A-D. Groups C and D represent complex rotated functions and functions which have Gaussian noise added. CLPSO outperforms all other variants on these functions in Groups C and D. This is an interesting result, as it implies that introducing more variation into the possible influences for velocity update results in an increase in the convergence potential, regardless of the nature of the function being optimised. A potential limitation of this variant is that the velocity modification considers each dimension independently. This may affect problems with a high degree of interdependence between the dimensions: that is, a value on one dimension is strongly correlated with some value in another dimension. This issue is potentially addressed by the FDR-PSO variant considered below.

2.1.8 Fitness Distance Ratio (FDR) Particle Swarm Optimisation

Fitness Distance Ratio Particle Swarm Optimisation (FDR-PSO) is a variant of PSO developed by Peram et al. (2003) which, although similar to CLPSO (Liang et al., 2006), uses the fitness distance of the particles' best solution so far, rather than the Euclidean distance, to determine the influence on a particle of other particles.

FDR-PSO is conceptually similar to an early idea of my own Kenny (2005), in that it attempts to include particles which are similar in fitness to the current particle. My variant was inspired by Differential Evolution (DE) (Storn & Price, 1995; Price, 2005) to address the issue of stagnation, in which I state:

…. Taking influence from differential evolution, it is arguable that rather than being influenced solely by some sort of population best and its own previous best, an individual particle should be influenced, at least partly, by closer neighbours to itself. It is believed, although not currently confirmed through experiment, that this would lead to [a] system less influenced by local optima and therefore more fully explore the solution space.

A possible formula for [the] velocity vector, is therefore:

$$\vec{v} = \varphi_1 \cdot (\vec{r}_1 - x_i) + \varphi_2 \cdot (\vec{r}_2 - \vec{x}_i) + \varphi_3 \cdot (\vec{p}_b - \vec{x}_i)$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}$$

where φ_1, φ_2 and φ_3 are defined as a function of the total 'influence space' rather than just random values, with no relationship to the distance, and are therefore dynamic and effectively define the attractiveness of the distant particle. r_1 and r_2 are neighbours, possibly randomly chosen, with $r_1 \neq r_2 \neq p_g \neq i$, where i is the current particle and p_g is the population best. The total sum of φ_1, φ_2 and φ_3 should sum to no more than 2.0, in order to ensure convergence, which is consistent with the usual PSO condition.

FDR-PSO works by updating each particle's velocity in accordance with the following modified velocity equation:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i) + \varphi_{3i} \cdot (\vec{p}_n - \vec{x}_i)$$

The additional element, $\varphi_{3i} \cdot (\vec{p}_n - \vec{x}_i)$, which contributes a proportion of the difference between $\vec{x}_i$ and $\vec{p}_n$ is defined as:

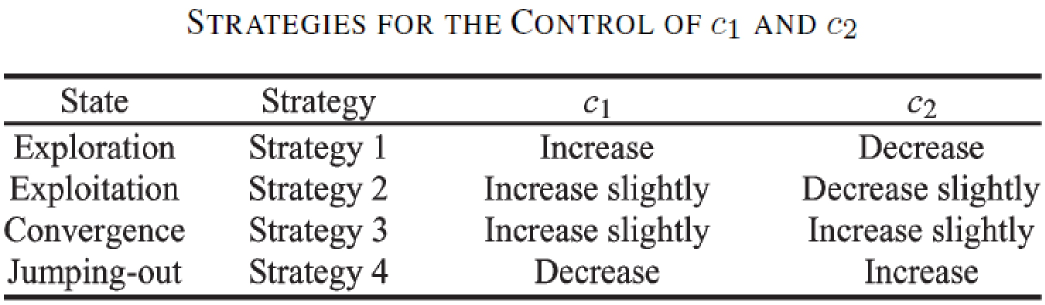
$$\frac{fitness(p_j) - fitness(p_i)}{\left| p_{jd} - p_{id} \right|}$$

where p_j is the particle that maximises the above equation, $fitness$ is the objective function which returns a measure of how good the solutions contained in p_j and p_i are.

Although this variant adds computational cost per iteration, it also brings an improvement when tested against standard test functions, in that the variant is less susceptible to premature convergence. The introduction of the element expressing a relationship between the fitness distance of a particle to that of the next best particle to maximise the fitness equation has made FDR-PSO one of the best performing variants on the test functions in Baskar & Suganthan (2004), Liang et al. (2006), Liang & Suganthan (2005), and Veeramachaneni et al. (2003).

2.1.9 Adaptive Particle Swarm Optimisation

Another development, Zhan et al. (2009), represents a substantial development on the velocity equation, and builds on the development of Ratnaweera et al. (2004), Peram et al. (2003), and van den Bergh (2002). Zhan et al. identify four stages of convergence to which they apply different settings of the PSO parameters as shown below:



STRATEGIES FOR THE CONTROL OF c_1 AND c_2

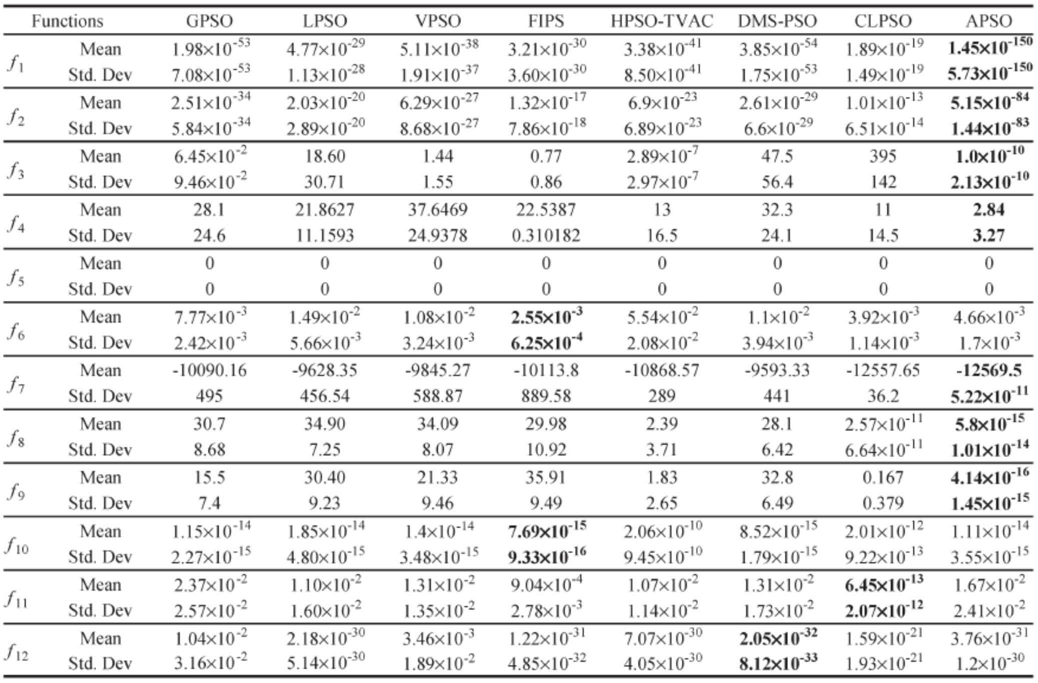
State	Strategy	c_1	c_2
Exploration	Strategy 1	Increase	Decrease
Exploitation	Strategy 2	Increase slightly	Decrease slightly
Convergence	Strategy 3	Increase slightly	Increase slightly
Jumping-out	Strategy 4	Decrease	Increase

Table 1: Strategies for setting the acceleration coefficients reproduced from Zhan et al. (2009)

Each of the strategy said above are employed depending on the stage the swarm is in. This is determined by the following equation reproduced from Zhan et al.:

$$f = \frac{d_g - d_{min}}{d_{max} - d_{min}} \in 0,1$$

with the 0,1 divided into 4 ranges to identify the current stage.



Functions		GPSO	LPSO	VPSO	FIPS	HPSO-TVAC	DMS-PSO	CLPSO	APSO
f_1	Mean	1.98×10^{-53}	4.77×10^{-29}	5.11×10^{-38}	3.21×10^{-30}	3.38×10^{-41}	3.85×10^{-54}	1.89×10^{-19}	$\mathbf{1.45\times10^{-150}}$
	Std. Dev	7.08×10^{-53}	1.13×10^{-28}	1.91×10^{-37}	3.60×10^{-30}	8.50×10^{-41}	1.75×10^{-53}	1.49×10^{-19}	$\mathbf{5.73\times10^{-150}}$
f_2	Mean	2.51×10^{-34}	2.03×10^{-20}	6.29×10^{-27}	1.32×10^{-17}	6.9×10^{-23}	2.61×10^{-29}	1.01×10^{-13}	$\mathbf{5.15\times10^{-84}}$
	Std. Dev	5.84×10^{-34}	2.89×10^{-20}	8.68×10^{-27}	7.86×10^{-18}	6.89×10^{-23}	6.6×10^{-29}	6.51×10^{-14}	$\mathbf{1.44\times10^{-83}}$
f_3	Mean	6.45×10^{-2}	18.60	1.44	0.77	2.89×10^{-7}	47.5	395	$\mathbf{1.0\times10^{-10}}$
	Std. Dev	9.46×10^{-2}	30.71	1.55	0.86	2.97×10^{-7}	56.4	142	$\mathbf{2.13\times10^{-10}}$
f_4	Mean	28.1	21.8627	37.6469	22.5387	13	32.3	11	**2.84**
	Std. Dev	24.6	11.1593	24.9378	0.310182	16.5	24.1	14.5	**3.27**
f_5	Mean	0	0	0	0	0	0	0	0
	Std. Dev	0	0	0	0	0	0	0	0
f_6	Mean	7.77×10^{-3}	1.49×10^{-2}	1.08×10^{-2}	$\mathbf{2.55\times10^{-3}}$	5.54×10^{-2}	1.1×10^{-2}	3.92×10^{-3}	4.66×10^{-3}
	Std. Dev	2.42×10^{-3}	5.66×10^{-3}	3.24×10^{-3}	$\mathbf{6.25\times10^{-4}}$	2.08×10^{-2}	3.94×10^{-3}	1.14×10^{-3}	1.7×10^{-3}
f_7	Mean	-10090.16	-9628.35	-9845.27	-10113.8	-10868.57	-9593.33	-12557.65	**-12569.5**
	Std. Dev	495	456.54	588.87	889.58	289	441	36.2	$\mathbf{5.22\times10^{-11}}$
f_8	Mean	30.7	34.90	34.09	29.98	2.39	28.1	2.57×10^{-11}	$\mathbf{5.8\times10^{-15}}$
	Std. Dev	8.68	7.25	8.07	10.92	3.71	6.42	6.64×10^{-11}	$\mathbf{1.01\times10^{-14}}$
f_9	Mean	15.5	30.40	21.33	35.91	1.83	32.8	0.167	$\mathbf{4.14\times10^{-16}}$
	Std. Dev	7.4	9.23	9.46	9.49	2.65	6.49	0.379	$\mathbf{1.45\times10^{-15}}$
f_{10}	Mean	1.15×10^{-14}	1.85×10^{-14}	1.4×10^{-14}	$\mathbf{7.69\times10^{-15}}$	2.06×10^{-10}	8.52×10^{-15}	2.01×10^{-12}	1.11×10^{-14}
	Std. Dev	2.27×10^{-15}	4.80×10^{-15}	3.48×10^{-15}	$\mathbf{9.33\times10^{-16}}$	9.45×10^{-10}	1.79×10^{-15}	9.22×10^{-13}	3.55×10^{-15}
f_{11}	Mean	2.37×10^{-2}	1.10×10^{-2}	1.31×10^{-2}	9.04×10^{-4}	1.07×10^{-2}	1.31×10^{-2}	$\mathbf{6.45\times10^{-13}}$	1.67×10^{-2}
	Std. Dev	2.57×10^{-2}	1.60×10^{-2}	1.35×10^{-2}	2.78×10^{-3}	1.14×10^{-2}	1.73×10^{-2}	$\mathbf{2.07\times10^{-12}}$	2.41×10^{-2}
f_{12}	Mean	1.04×10^{-2}	2.18×10^{-30}	3.46×10^{-3}	1.22×10^{-31}	7.07×10^{-30}	$\mathbf{2.05\times10^{-32}}$	1.59×10^{-21}	3.76×10^{-31}
	Std. Dev	3.16×10^{-2}	5.14×10^{-30}	1.89×10^{-2}	4.85×10^{-32}	4.05×10^{-30}	$\mathbf{8.12\times10^{-33}}$	1.93×10^{-21}	1.2×10^{-30}

Table 2: Search result comparisons among eight PSOs on 12 test functions, reproduced from Table VI, Zhan et al. (2009). GPSO: Global PSO; LPSO: Local PSO; VPSO: von Neumann topological structured PSO; FIPS: fully informed particle swarm; HPSO-TVAC: PSO with time-varying acceleration; CLPSO: Comprehensive-learning PSO; APSO: Adaptive PSO

Table 2, reproduced from Zhan et al. (2009) shows the experiment results. The results show that, for most of the test problems, this implementation of PSO outperforms the other implementations. The implementation is interesting because it attempts to give the heuristic adaptability depending on the stage of convergence.

Other papers which modify the velocity equation include: Changhe Li et al. (2012) and Zhi-Hui Zhan et al. (2011).

Developing the PSO model

We now consider developments on the PSO model that extended the original heuristic. A feature of PSO algorithm is that systems have converged poorly, especially at the end of the execution lifetime. As a result, many researchers have made modifications that take it outside the swarm analogy. The aim is to produce a converging optimisation technique rather than model the phenomena present in nature. In this section, the main methods and themes will be covered.

2.1.10 Neighbourhood Topologies

The introduction of neighbourhood topologies into PSO is probably one of the most significant areas of PSO research. In a given solution space, the neighbourhood topology simply describes the way in which the particles are dispersed and how they 'relate' to each other: that is, how they communicate their solutions, and how the total solution space is divided by the subpopulations of particles.

Of the many works on the subject, the most significant are Kennedy (1999), Kennedy & Mendes (2002), Mendes et al. (2004) and Iacoban et al. (2003a, 2003b). Kennedy (1999), Mendes et al. (2004), Huang & Mohan (2005) and Tsujimoto et al. (2012) explore a number of different neighbourhood options.

Kennedy (1999) and Tsujimoto et al. (2012) specifically explore the effect of several sociometries[3], with the introduction of direct connections between individuals in the population, in addition to the *gbest* and *lbest* topologies. These configurations were tested using the constriction coefficient version of particle swarm optimisation. Mendes, et al. (2004) explore the effect of several differing topologies, namely, *gbest*, *lbest*, pyramid, von Neumann and four clusters. These are defined as:

- *gbest* or Fully Informed Particle Swarm (FIPS) treats the entire population as the individual's neighbourhood.
- *lbest* where adjacent members of the population comprise the neighbourhood, such that, if *n* represents the number of neighbours and *n* = 3, the neighbourhood consists of 3 particles: the current particle and the adjacent neighbour on the right and left. Note, that *adjacent* refers to the indexing of the particles and not their position in the solution space.
- *pyramid* a three-dimensional wire-frame triangle, shown in Figure 4.
- *von Neumann* a square lattice in which the extremities connect as a torus.
- *four clusters*, completely interconnected, and connected among themselves by a few shortcuts, shown in Figure 5.

[3] Sociometry: the quantitative study of social relationships.

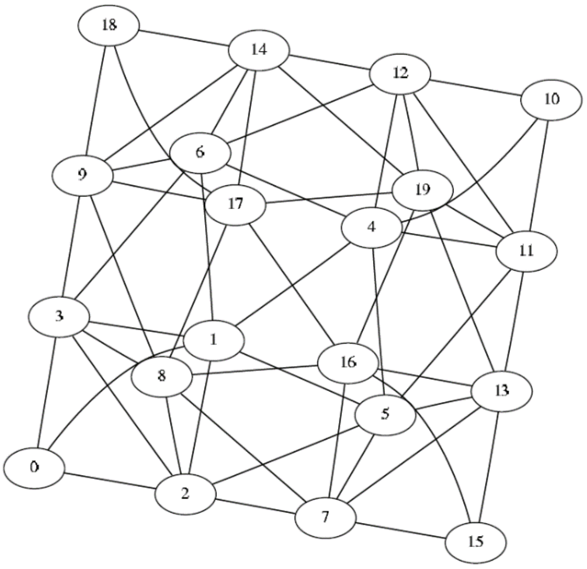


Figure 4: Pyramid neighbourhood

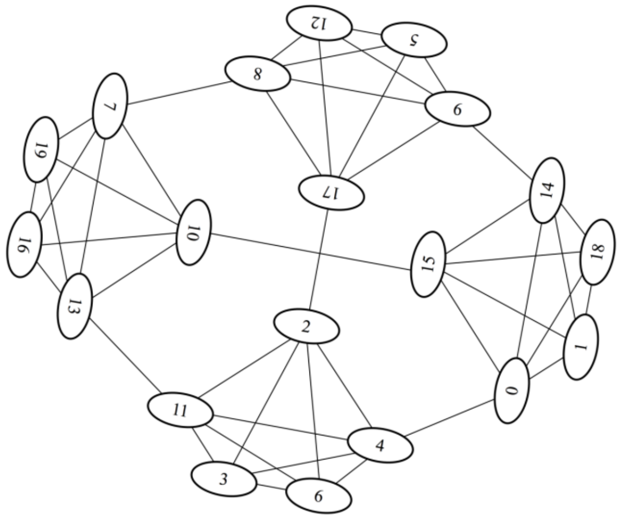


Figure 5: Four clusters neighbourhood

Although no best topology was found, the authors report that using a neighbourhood topology preserves greater population diversity since it slows down the spread of information between population members.

A particle swarm's ability to solve a given problem has been found to be dependent on the neighbourhood topology, (Kennedy, 1999; Mendes et al., 2004). To some extent this is to be expected, given the variable nature of the solution spaces, although there is an implicit conclusion in Mendes et al. (2004):

> "Even though the von Neumann sociometry seems to be a very good performer, it is important to understand why it is such a good choice. It seems also important to consider the implications of apparently not needing to consider the past experience of the individual. Also, these new variants need to be thoroughly tested with different problems, to attest for their robustness."

Reinforced by observations by Iacoban et al. (2003a) regarding the effect of variable interactions in a swarm, it is possible to draw a tentative conclusion that at the end of a run convergence is affected by the neighbourhood topology. We shall return to the notion that the structure, and to some extent the behaviour, of a swarm affects its ability to converge, since it is clear that if the ability of the algorithm to converge during the latter stages of a run is not strong enough sub-optimal solutions are likely to result. Potentially, this could be made worse by a poor relationship structure between the particles.

Other neighbourhood papers include: Qu et al. (2013), Wu & Chow (2013), Yan-Liang Li et al. (2013), and Bala Krishna & Doja (2011).

2.1.11 The gbest inclusion in FIPS

At this point it is worth pausing to make an observation on the inclusion of the *gbest* particle in its own neighbourhood. There is an inconsistency in the literature as to whether in a Fully Informed Particle Swarm (FIPS) neighbourhood a particle can be its own *gbest*. For example Kennedy (1999) and Jun-Jie & Zhan-Hong (2005) explicitly exclude the *gbest* particle from its own neighbourhood, while others (e.g., van den Bergh & Engelbrecht, 2002; Engelbrecht et al., 2005) imply the *gbest* is included in its own neighbourhood. This is contrasted with the *lbest* neighbourhood where the particle is included in its own neighbourhood Kennedy & Mendes (2003). The inclusion of *gbest* in its own neighbourhood means that the particle is drawn at a proportion of twice its new velocity towards its previous best.

This minor difference results in the *gbest* particle ceasing to move, since the new velocity becomes a zero sum (at least more rapidly). It is arguable that this contributed to the development of GCPSO van den Bergh (2002). If the *gbest* particle is not included in its neighbourhood it then becomes subject to the second best particle in the FIPS neighbourhood and hence continues to have limited exploration abilities. Of course, this may not make much difference within a closely converged swarm. However, in earlier stages it means that *gbest* retains a dual influence in its velocity from its own best position and the best position of the second best particle. It also maintains the oscillatory movement between the best and second best.

It is of course arguable that the *gbest* particle will not really benefit from a particle that is being influenced by another particle relatively close by. However, in the early stages of convergence, excluding the *gbest* particle from its own neighbourhood means that the *gbest* particle has an additional influence other than its own best position. If a particle is the best particle in a neighbourhood and used as the neighbourhood's best, the velocity update equation is equivalent to:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_i - \vec{x}_i)$$

The difference between the above equation and the usual velocity update equation is that the particle's best has been used in place of what would normally be the neighbourhood best. This would be the case if the particle were the best particle in the neighbourhood. In this case, the particle becomes influenced by its own best position twice, effectively doubling its velocity to its own best position, and thus increasing the likelihood of stagnation for the particle. The decay in velocity via the inertia weight leads to convergence on a point within a hypersphere between its current position and its best position. The particle will not escape the area already searched.

2.1.12 Boundary Conditions

We briefly consider what options are available when a particle reaches the condition where its position in a given dimension equals $\pm Xmax$. In summary, the general options are either that the particle's position in that dimension is reflected back, or it is mirrored, or has a random perturbation added to move it away from the boundary, or it is made increasingly difficult for a particle to reach the $Xmax$ value. The few papers that explicitly consider what happens under boundary conditions in PSO include: Huang & Mohan (2005), Xu & Rahmat-Samii (2007), Mikki & Kishk (2007), Wenhua et al. (2011), and Khan & Brown (2012). Helwig et al. (2013) review a number of alternatives on several test functions. These include: Hyperbolic, Random Back, Nearest-Z, Random-Z, Reflect-Z, Rounded Mirror, Infinity, Infinity-C . Each of these Boundary Methodologies is tested on the CEC’05 set of test functions, with a full description of each boundary methodology given in the paper. Although the researchers use an extensive suite of test functions, it is notable that no boundary condition outperforms another over the entire suite.

2.1.13 Mutation Operators

Work has been carried out to improve on this performance of PSO by drawing on researchers’ previous experience with genetic algorithms, in which mutation operators or other heuristic techniques play a part.

The use of mutation or crossover operators to enhance PSO is exemplified by: Higashi & Iba (2003), Wen-Jun & Xiao-Feng (2003) and Abdelbar & Abdelshahid (2004). The latter adds a local search operator. The former take the option of perturbing a stagnated swarm by using Gaussian mutation, effectively nudging the particles to explore slightly farther afield. Wen-Jun & Xiao-Feng (2003) reports on the combination of the Differential Evolution (DE) (Storn & Price, 1995; Price, 2005) crossover operator to calculate a new position vector for a particle. In DE the crossover operator constructs a new solution by probabilistically introducing the difference between two vectors or the value of the parent vector, such that:

$$y_i = a_i + F(b_i - c_i)$$

where a, b and c are randomly selected from the population and $F \in \{0,2\}$. The mutation operator is applied if two random variables r and CR, $r, CR \in \{0,1\}$ meet the following criteria $r < CR$ otherwise the standard velocity update is used.

Of significance in this variation is that the particle only moves to the new position if that position is better than its previous position, as defined by the objective function. Effectively, the selection process improves exploration capacity if the existing solution cannot be exploited, because the velocity is small and the landscape surrounding the candidate solution results in non-improving solutions. However, it should be noted that, as with any crossover operator, the combination of good values matched with a combination of poor values might in fact just miss the global optimum.

Dynamic and Multi-objective Problems

In this sub-section we consider adaptations to PSO in order to handle optimisation problems which have dynamic or multi-objective optima. These are characteristics which add significant overheads to the processing required by the swarm in order to maintain convergence on the true optima. Both of these areas are still the subject of significant PSO research, for dynamic problems: Eberhart & Shi (2001), Eberhart & Shi (1998), Carlisle & Dozier (2000, 2001), Roberge et al. (2013). And for multi-objective problems: Coello Coello et al. (2004), Hu & Eberhart (2002), Coello Coello & Maestria (2002), Xu et al. (2006), Xiao et al. (2003), Blackwell & Branke (2004), Yen & Wen-Fung Leong (2009), Zeng et al. (2011), Changhe Li & Shengxiang Yang (2012).

2.1.14 Dynamic Problems

There are many techniques in the literature to deal with dynamic problems, where the characteristics of the problem and therefore the optimal solution change. One common approach considers the re-initialisation of part of the swarm; another uses a sentinel approach.

In the re-initialisation case (Eberhart & Shi, 2001, 1998), two options were considered. First, periodically 'forgetting' the previous *gbest* and using the existing swarm positions as starting positions for the new swarm. Second, re-initialising the swarm with new randomised starting positions. This has proven to be effective, (Carlisle & Dozier, 2000, 2001) in maintaining track of the best solution in dynamic environments; but clearly it is essentially rerunning the swarm.

Eberhart & Shi (2001) suggests that the parameter USEBETTER, a parameter discarded in PSO's early development Kennedy & Eberhart (1995), might be used in optimisation of systems where the dynamic change in the optimal value is small. The effect of the USEBETTER parameter is to change the particle's current position only if the new position results in an improved solution. However, it is noted here that such an implementation decision requires additional prior knowledge of the dynamics of the problem domain.

The second solution for dynamic problems was presented by Carlisle & Dozier (2000, 2001), in which a sentinel particle is proposed; the sentinel periodically checks the optimal value of its existing solution, forcing the other particles to re-evaluate their solutions if a change in the optimal solution has been detected. A random particle is selected as the sentinel on iteration, storing a copy of its score for its current position. Before updating its current position in the next iteration, it re-evaluates its old position; if the two results differ, then the whole swarm re-evaluates. This solution has the benefit of retaining the previous experience of the other particles for consideration as a new best solution. However, both solutions affect the dynamics of the swarm, in that between periodic updates it is possible that particles are optimising to an out-of-date optimal value. Carlisle & Dozier (2000) evaluate the sentinel solution on several types of dynamic environment, in all cases it outperforms other PSO variants.

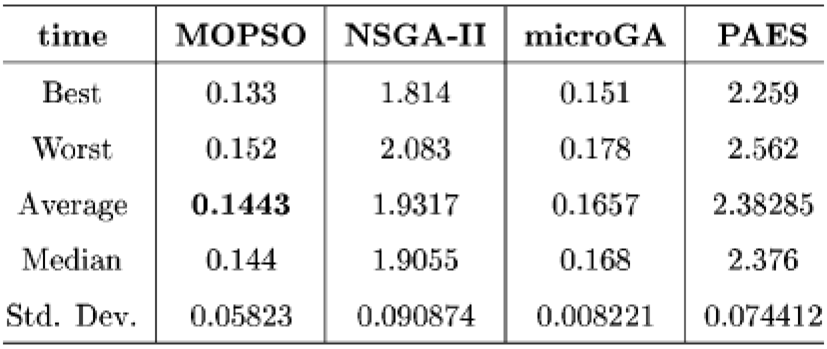
2.1.15 Multi-Objective Applications and their Problems

Multi objective problems are problems with multiple independent variables to be optimised simultaneously. Many real-world problems are of this type, for example Chemical Reaction Engineering, Chemical Plant Design, Printed Circuit Board Design, Process Optimisation, Facial Recognition, Structural Support Compliance and Antenna Arrays. The proposals we consider the most significant are discussed below. All research reported in this section uses Pareto front dominance[4].

Coello Coello et al. (2004) represents a significant contribution. It contains a significant review of the other work completed and presents substantial developments which

[4] Pareto front dominance is a line defined by solutions in a multi objective solution space, which are not completely dominated by other optimal solutions on the line. A vector x* is called Pareto optimal if there exists no feasible vector x which would decrease some criterion without simultaneously increasing at least one other criterion. The Pareto front is therefore, the line containing Pareto optimal points.

significantly improve on the existing PSO algorithm for multi-objective problems. Sample results are below in Table 3:



time	MOPSO	NSGA-II	microGA	PAES
Best	0.133	1.814	0.151	2.259
Worst	0.152	2.083	0.178	2.562
Average	**0.1443**	1.9317	0.1657	2.38285
Median	0.144	1.9055	0.168	2.376
Std. Dev.	0.05823	0.090874	0.008221	0.074412

Table 3: Summary of results reproduced from Coello Coello et al. (2004), Table XX showing runtimes. MOPSO: Multiobjective PSO; NGSA-II: nondominated sorting genetic algorithm; microGA: microgenetic algorithm form multiobjective optimisation; PAES: Pareto archive evolutionary strategy.

The algorithm is developed and refined through four experiments. The authors' strategy is to maintain an extended archive of Pareto dominated solutions. A solution is replaced in the archive by a new solution only if the new solution dominates an existing solution. The original velocity equation for PSO is then modified to include a random Pareto dominant solution from the existing archive, thereby extending the role of *gbest* in the original equation. In addition, to address a shortcoming identified in previous work Coello Coello & Maestria (2002), a mutation operator is used to mutate a reducing subset of the population in order to avoid premature convergence on a non-dominated Pareto front. It should also be noted that the same mutation operator is applied to the ranges of each dimension, gradually reducing the size of the solution space as the position of the Pareto front is identified. The researchers' findings are summarised as:

- Introduce a mutation operator, whose range of action varies over time to improve the exploratory capabilities, eliminating the need to fine tune the inertia weight.
- Introduce a results repository of size 250.

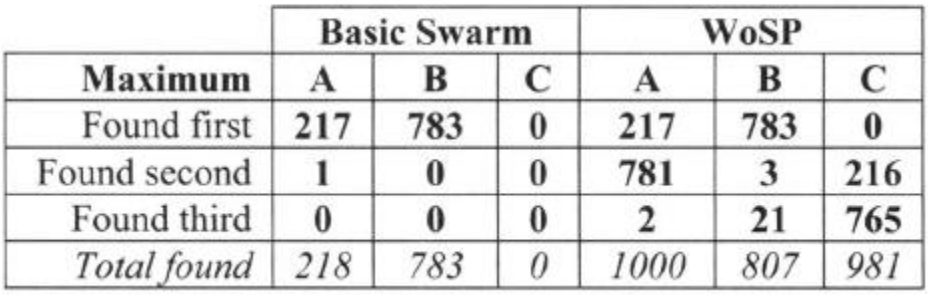
- On average a population of 100 particles is best to optimise the Pareto front for all test functions.

- A grid of 30 divisions for the solution space is optimal.

Hu & Eberhart (2002) proposes a different technique: that of a dynamic neighbourhood. This again uses Pareto dominated solutions, although this implementation finds solutions on the Pareto front less successfully than the system proposed by Coello Coello et al. In Hu and Eberhart's algorithm, the Euclidean distance from every particle to every other particle is calculated per iteration; the particle then selects its neighbourhood for that generation from the nearest *n* particles. The local optimum within the new neighbourhood is then calculated. In addition, the particle's best is only updated if the current solution dominates all previous bests in the particle's history. Other multi-objective research, not discussed here, includes: Ki-Baek Lee & Jong-Hwan Kim (Ki2013), Xiangtao Li & Minghao Yin (2013), Senthilnath et al. (2013), Lixin Tang & Xianpeng Wang (2013), Rubio-Largo et al. (2012), Sarikhani & Mohammed (2011), Ashabani & Mohamed (2011), Xiaodong Li & Xin Yao (2012), and Maltese et al. (2015).

Multi Swarm Implementations

Multi-swarms are implementations of PSO in which multiple swarms are used to maximise exploration and exploitation. There have been several implementations of multi-swarms, of which our innovation in Chapter 6 is one. Notable among these are Hardin (2005), Blackwell & Branke (2006), Zhao et al. (2008), Meng-xin et al. (2014) and Srivastava & Singh (2015). The aim of these multi-swarm implementations is generally to try and maintain a balance between exploitation and exploration. The implementation used in Hardin (2005) divides the swarm into particles in one of two modes. In the first mode the particles are operating far apart, defined as proportional to the particles' velocity and distance. When the particles have settled in an area close to each other they are ejected to the next *wave*, determined by an index value. Particles only interact with other particles in the same wave. The WoSP variant is tested on a multi-objective problem, with some success, see below:



	Basic Swarm			WoSP		
Maximum	A	B	C	A	B	C
Found first	**217**	**783**	**0**	**217**	**783**	**0**
Found second	**1**	**0**	**0**	**781**	**3**	**216**
Found third	**0**	**0**	**0**	**2**	**21**	**765**
Total found	*218*	*783*	*0*	*1000*	*807*	*981*

Table 4: The relative performance of the basic PSO and WoSP algorithms, reproduced from Hardin (2005).

The innovation in Zhao et al. (2008) by contrast arranges the particles into sub-swarms based on their solution fitness with the addition of a randomised regrouping interval to allow for the global communication of information. This variant was tested on the 7 Test Function from CEC'08, namely:

F1: Shifted Sphere Function
F2: Shifted Schwefel's Problem
F3: Shifted Rosenbrock's Function
F4: Shifted Rastrigin's Function
F5: Shifted Griewank's Function
F6: Shifted Ackley's Function
F7: FastFractal "DoubleDip" Function

Although the innovation reports good results on all functions, no comparative results are reported. Other multi-swarm innovations, not discussed here, include: Chen (2009), Röhler & Chen (2011), Chen & Montgomery (2011), Xue Wang & Sheng Wang (2011), Röhler & Chen (2012), and Changhe Li & Shengxiang Yang (2012).

Observation on Swarm Behaviour

From the literature review presented above, there is significant uncertainty over which implementation of PSO is best for given problem domains, though Clerc (2006) and van den Bergh (2002) suggest useful parameter settings. Given that PSO is a very effective convergent technique, it is reasonable to suppose that it would be effective working on real-world data, which will inevitably include noise. Some examples of papers which consider noisy applications are: Kennedy & Eberhart (1995), Eberhart & Kennedy (2004), Slade et al. (2004), Wachowiak et al. (2004), Lei & Liying (2005), Scriven et al. (2010), Ying Li et al. (2011), Kachroudi et al. (2012), Fodorean et al. (2013), and Senthilnath et al. (2013). Many the applications researched rely on pre-preparation, either in the form of user adjustment or pre-processing of the data before subjecting it to the particle swarm, and usually in support of another heuristic such as Artificial Neural Network (ANN). An interesting exception is Ince et al. (2009), in which PSO is used to develop the structure of the ANN as well as find its weights, for patient specific ECG classification. Nevertheless, a useful research topic would result from processing some well-defined, but noisy, real-world data with a suitably defined objective function. Could PSO discriminate between wanted and unwanted information successfully?

In the remainder of the section, we briefly discuss an observation that has been little explored in the literature. In the discussion above, one of the recurring themes has been the swarm's inability to converge at all, or a tendency to stagnate around the optima. The following is a brief discussion and analysis, which attempts to highlight an underlying theme in PSO research. It is a theme which appears for the most part to have been explicitly missed and some examples of work which is implicitly acknowledged. It concerns what happens to the *gbest*[5] particle.

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i)$$

In the inertia weight variant, above Shi & Eberhart (1998b), if no further improvement is found, the velocity of *gbest* would decrease to zero. In this scenario, once the particle becomes *gbest* then it effectively ceases to be an effective contributor to the swarm's convergence potential.

We would suggest that this should not be the case, and that *gbest* should continue to explore, adopting an exploration method which reflects its previous experience. Instead of the arbitrary resetting of the velocity equation to a randomised re-initialisation value (e.g., Ratnaweera et al., 2004), the particle might explore using the difference between its previous velocity, the velocity before arriving at *gbest*, and the velocity which caused its arrival at the *gbest* position. Another possibility is to use the gradient descent technique to find the optimal solution in the area of the solution space surrounding the current *gbest*. This remains an open research question.

The problem of *gbest* stagnation was identified by van den Bergh (addressed in van den Bergh, 2002; van den Bergh & Engelbrecht, 2004), with the development of the idea of Guaranteed Convergence PSO. The behaviour of the particles once a non-improving *gbest* has been found leads to the stagnation condition, van den Bergh (2002). This is a weakness in the behaviour of PSO: the inertia behaviour of a swarm means that the necessary diversity required to build the momentum to escape is lost before stagnation is reached.

[5] Although we consistently refer to *gbest*, this is for convenience. The argument applies equally to the neighbourhood best. The term pbest for "population best" is not used here to avoid confusion between population and particle best.

A more recent attempt to addressing stagnation is a Triangular Particle Swarm Optimisation Qais & AbdulWahid (2013). This paper implements a cosine modification to the velocity equation as follows:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot \cos(\vec{p}_i / \vec{x}_i) \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot \cos(\vec{p}_g / \vec{x}_i) \cdot (\vec{p}_g - \vec{x}_i)$$

The inclusion of the cosine of the ratio of the particle’s best position to the particle’s current position will have the effect of reducing the length of the vector produced by the equation. Theoretically, this will mean that the swarm will be better able to find improved solutions at convergence. However, the solution was only tested on standard test functions, and it would be interesting to see the effect where real-world data introduces noise or has a significant number of local optima.

Research on Real-World Data

As discussed earlier PSO was originally developed as a model of social interaction. It was later developed as an optimisation technique. However despite this, there were relatively few real-world applications developed in PSOs early development. Both Eberhart & Kennedy (2004), and Slade et al. (2004) use PSO to successfully categorise different datasets. Given this success, it is perhaps surprising that further applications on real-world data were not developed earlier. The following paragraphs briefly review the real-world PSO literature.

The PSO literature to include real-world data has become very broad, for example: Wen-Tsai Sung et al. (2014), Li et al. (2013), Peng et al. (2014), Siano & Mokryani (2013), Chen et al. (2013), Wu et al. (2008), Sun et al. (2014), and Antoniou et al. (2013). Kulkarni & Venayagamoorthy (2011) present a review and include useful analysis of issues and PSO variants applied to optimising Wireless Sensor Networks. Senthilnath et al. (2012, 2013) and Kit Yan Chan et al. (2013) include applications as diverse as medical analytics, through product sale forecasting, multiple sequence alignment (Bioinformatics), wireless data traffic management, and image alignment through to short-term vehicle traffic flow prediction. Of these, probably the most interesting are Wen-Tsai Sung et al. (2014), Antoniou et al. (2013), and Kit Yan Chan et al. (2013). The first, Wen-Tsai Sung et al. (2014), uses PSO to provide analytics to life sign data collected through Bluetooth and ZigBee sensors in Android devices; PSO is then used to provide analytical functionality before the data is forwarded to the Cloud. This application has significant potential within the field of telemedicine, allowing clinicians to assess patient data on a near real-time basis, and thus potentially avoiding unnecessary acute admissions to hospital. Such an application of PSO emphasises the flexibility of PSO to produce useful analytical functionality without being part of a bigger hybrid solution. Kit Yan Chan et al. (2013) present an alternative application, that of reducing vehicle traffic congestion by optimising mobility in real-time using data gathered from road sensors. In Antoniou et al. (2013), PSO is applied to the application of wireless network congestion. The application uses PSO to produce an adaptive traffic management policy in real-time, depending on the network traffic at that point in time. This application is interesting because of its real time use of PSO.

Since PSO has become an established optimisation technique, there have been numerous other papers concentrating on biomedical data, including Delgado Saa & Cetin (2013) and Cheng-Chi Wu et al. (2013). Kentzoglanakis & Poole (2012) used a combination of PSO and ACO to reverse engineering gene expression for the Escherichia coli bacterium.

Perhaps one of the most intriguing applications of PSO is Matsumura et al. (2013). Although strictly speaking, real-world data is not used, these researchers develop a hybrid algorithm combining PSO with Evolutionary Strategies in order to produce an optimisation model (see Section 0 for a discussion of evolutionary strategies). The paper initially presents a series of experiments on standard test functions and demonstrates that the hybrid algorithm is more consistent, although not always better, than the comparative algorithms. The paper goes on to use the algorithm relatively successfully to produce computer models of dinosaur gaits. A useful enhancement would be to see how well it models the gait of current carnivores or similar living flightless birds or lizards.

Other Optimisation Techniques

In this section we touch on several other optimisation techniques that are used within the computational optimisation community to find solutions for complex mathematical problems. As we have seen above, many of the adaptations to PSO have been inspired by other successful optimisation techniques. This section briefly discusses the details of three of the more widely used techniques and some of the principles used in applying them to specific problems We also briefly discuss the difference between test and training data, and multiple linear regression, which is used as the benchmark comparison technique in this thesis.

2.1.16 Genetic Algorithms

Genetic algorithms (GAs) attempt to mimic the way genomes evolve (Goldberg, 1989; Michalewicz, 1996), using three main operators: selection, mutation and crossover (crossover is sometimes called recombination). For each generation, one or all of the operators may be applied in order to create the new population for the next generation. GAs are normally suited to discrete problem representations, usually binary.

An initial population of candidate solutions is randomly generated, each generally a bit array representing a possible solution to the problem. Each successive generation is a recombination of selected individuals from the current generation, using a combination of the mutation and crossover operators. The process of forming the next generation starts by selecting the fittest or best solutions in the current generation, as measured by a *fitness function*, a function which evaluates each candidate solution and returns an evaluation of its quality as a solution to the problem in question. After randomly selecting at least two of the fittest solutions as parents, the mutation and crossover operators are chosen probabilistically. The mutation operator may randomly flip one or more bits from its original state before adding it to the new solution. The crossover operator randomly selects one or more points in the bit array, and the arrays then exchange bits after, or between, these points. Although GA uses the best solutions to generate the next generation, the GA model is not directly influenced by the landscape of the solution space being searched. Rather, it comprises a randomised re-pairing of the existing good solutions. A more complete discussion is available in Goldberg (1989), Michalewicz & Fogel (2004), and Michalewicz (1996).

2.1.17 Evolutionary Strategies

Evolutionary Strategies (ES), (Corne et al., 1999; Storn & Price, 1995), generate new offspring by calculating some sort of 'difference' between two or more selected parents. However, ES differs from GA in that it only uses the mutation operator to generate the next population. When implemented, the individuals making the parents for the next generation are derived from either of the following possibilities:

> (μ + λ): In this variant the selection of the parents for the next generation is derived from the parents and offspring. The worst performing parents are then discarded to keep the population constant.
>
> (μ, λ): In this variant only the offspring are considered for possible selection as parents for the next generation. Obviously in this version better performing parents from the previous generation are "forgotten".

The other advantage of ES, as with PSO, is that it works efficiently in the real number space.

2.1.18 Artificial Neural Networks

Artificial Neural Networks (ANNs) are inspired by the central nervous system attempting to model the interactions between neurons. In artificial intelligence, they are typically used for machine learning and recognition tasks. Although there are different types of ANN all have a similar structure of layers of nodes interconnected by weights. A sample ANN is illustrated in Figure 6.

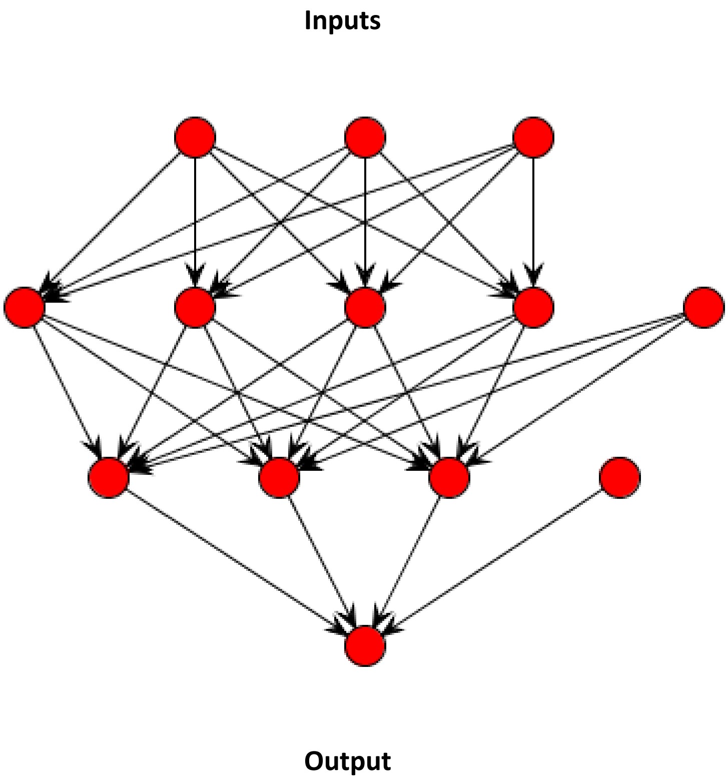


Figure 6: Example Multilayer Artificial Neural Network

In the above figure, the red dots are *nodes;* each node incorporates a mathematical function that varies depending on the network topology – typical functions used are sum or the sigmoid. Each connection between the nodes is *weighted* with a numerical factor, which represents the importance of the contribution made by that node. Nodes receiving no input connections are referred to as *biases*, and allow for the relative influence of the nodes to which they are connected to be modified. ANNs need to be trained in order to find the optimal values of the weights connecting the nodes. There are two main types of training paradigm, *supervised* and *unsupervised.* In supervised learning the weights are found through the identification of a training set to which the weights are trained. In unsupervised learning the ANN is provided with a cost function, and the aim is to find the set of weights which minimises this function. More detailed information can be found in Haykin (1998).

2.1.19 Multiple Linear Regression

Multiple linear regression is a statistical technique used to model the dependence of a variable on one or more independent variables. To be more specific, simple linear regression models the relationship between a pair of variables, multiple linear regression includes multiple variables. Multiple linear regression also differs from logistic regression in that it models a relationship of the form:

$$y = X\mathbf{B} + h$$

where y, X and h are vectors, $\mathbf{B}$ is the matrix which defines the relationship between the variables in y . The size of the matrix X determines the *degrees of freedom* in the model. The degrees of freedom are the number of variables which are free to change in order to model the relationship between the dependent variable and the independent variables. Multiple linear regression is useful when the independent variables are not highly correlated; this is particularly useful for establishing trend lines. Example applications are: risk analysis, epidemic spread, and the prediction of consumer spending. Further information on multiple linear regression can be found in Murphy (2012).

2.1.20 Test and Training Data Sets in Optimisation

In developing an optimisation model, typically the data is split into test and training sets. The training data is the dataset used to refine the model such that it can then make successful predictions on data contained in the test set. The test data set is used to validate the model, and typically will be smaller than the training data set, but will contain a range of sample input datasets which represent the full dataset as a whole.

The training data set will contain input values, each normally in the form of a vector, as is the case in this thesis. Output values will also be represented as a vector. The validation of a model against the test data set is important and should be performed using data that is independent of the training set. One of the potential complications which arise is that the model may over fit the training data: that is, the model matches the required outputs of training data so well that it is unable to predict new inputs which are not included in the training dataset. Further details on test and training data can be found in Haykin (1998), Goldberg (1989) and Michalewicz (1996).

Such models are usually termed regression models, two forms of which are *linear regression* and *logistic regression*. Linear regression is discussed in greater detail below. Logistic regression is used to measure the dependency between one categorical variable and one or more independent variables. In the following chapters we use Multiple Linear Regression for validation of our techniques.

This section has provided a brief introduction to heuristics used for problem optimisation. For a more complete discussion, see Corne et al. (1999) and de Castro & Von Zuben (2004).

Summary

This chapter presented a brief history of the development of particle swarm optimisation. The aim has been to present a roughly chronological account of development of the original heuristic, and of the enhancements made to it since inception in 1995. In addition, we have presented some of the specific implementations of the heuristic. We contend that PSO is better than other techniques for the following reasons:

- A small population size is required compared with other evolutionary algorithms.
- On a typical problem, fewer iterations are required to find an acceptable solution.
- PSO adapts its solutions between exploration and exploitation during convergence. Uses a guided search such that the values between the current and best solutions have to potential to be exploited

Sections 0 - 0 established the background, with subsequent sections exploring the major developments of, and modifications made to, the original algorithm, required by specific implementations for given problems, such as dynamic and multi objective problems. We suggested that many of the changes made to PSO in its early development were essentially modifications to the equations which claimed to be an improvement over another variant; this is also recognised by de Oca et al. (2009) who, in a detailed paper, evaluated a composite PSO algorithm integrating the modifications of others based on the modification's merit.

In the next chapter, we build on the foundations established in this chapter to offer a critique of past, and more recent, research. We develop a taxonomy of particle swarm optimisation, the purpose of which is to establish the weaknesses in the research and establish the basis for our suggested innovations.

Chapter 3

A Critique of Past Research

Particle swarm optimisation was developed with a view to modelling social behaviour, rather than the optimisation of nonlinear problems. In this section, we present a critical analysis of some of the developments of PSO since its inception.

Although in its original form PSO has proved to be a very fast converging algorithm, Kennedy & Eberhart (1995), it suffered from poor convergence to a good quality solution. The reason for this is that towards the end of its run the social and cognitive elements of the velocity equation contribute only a small value to the overall result, rather than being the dominant contribution, as they are in the earlier stages when the particles are more dispersed. The particle's most significant influence becomes its previous velocity, causing it to 'over-fly' the area of solution space containing the best solution. Therefore, no further improvement is seen from any other particle in the population, leading to a stagnation condition. Many researchers have attempted to develop improvements to address this.

What follows is an analysis of the developments in PSO grouped into four main classes these are:

1) Heuristics or Trial and Error: drawing from existing heuristics with a trial and error approach to finding improvements.

2) Lack of Analysis: the initial developments in PSO lacked a strong analytical theme.

3) Pragmatic Modification: these are modifications which were considered sensible by researchers to improve on results

4) Velocity Equation Modification: changes to the velocity equation after analysis of PSO convergence performance.

The chapter ends with a section covering later work important to PSO's development. An important point is that in doing this type of analysis we are intending to draw out the many themes which we have identified in the early development of PSO, and not to judge which of these is best, although developments in subsequent chapters draw more from analytical approaches.

Mathematical Analysis of PSO convergence

Investigation of the mathematical features of PSO has led to some advances. For instance, Shi & Eberhart (1998a, 1998b) and Clerc & Kennedy (2002) developed the inertia weight and constriction factor versions of the velocity equations respectively, along with developments such as van den Bergh & Engelbrecht (2002, 2004) and van den Bergh (2002), as discussed above, and Ratnaweera et al. (2004). However, there is still little detailed analysis of the performance of the particle swarm heuristic during the latter stages of convergence, when the majority of the particles are in the same area of the solution space and the dominant influence on new positions is the difference between the respective bests multiplied by the acceleration coefficients. Early approaches (Clerc & Kennedy, 2002; van den Bergh, 2002) analysed this phase of the PSO by removing the stochastic elements from the velocity equation, effectively reducing successive iterations to a recurrence relation. However, this also has the effect of reducing the swarm to a deterministic swarm while the analysis is being carried out, with the stochastic characteristics of the swarm not considered. The passivity theorem and Lyapunov stability analysis is used by a later analysis (Kadirkamanathan et al., 2006) retained the stochastic nature of PSO and used the passivity theorem and Lyanunov stability analysis to understand the dynamics of the particle swarm algorithm.

Incorporating features from genetic algorithms

There are several papers in which authors use mutation in some form to perturb the stagnation state in a pragmatic way; examples are: Higashi & Iba (2003), Jian et al. (2004), and Stacey et al. (2003). It is not our intention to imply that mutation within PSO algorithms can never be a useful operator. Rather, it is to suggest that using it purely to resolve the stagnation condition at the end of a run cycle addresses the issue of stagnation *after* it has occurred.

As noted above, Angeline (1998a) has also produced a variation on the existing PSO algorithm, using the concept of selection to remove poor performing individuals. The variation includes the tournament selection algorithm, in which each individual particle competes against k other individuals. The result of the tournament is then sorted to produce a rank table. The particles in the bottom half of this table have their current positions and velocities replaced with those of the top half of the table whilst retaining their own best positions. The results presented in the paper demonstrate a significant improvement in the time taken to find the optima for most selected test functions, indicating that refining the population's performance in this way would seem to be a reasonable step. However, it is noted here that nudging the particle's position and effectively moving it towards the direction of the last improvement means that one is assuming that the solution landscape has a definite direction of improvement. However, we do not intend to imply that tournament selection should not be considered as part of a particle swarm implementation, but that the implementation used should respect the swarm's integrity and the velocity equation, not the pragmatics of a particular application, thereby preserving the collective learning strengths of PSO.

Wen-Jun & Xiao-Feng (2003) use the crossover technique first developed by Storn & Price (1995) for differential evolution. The basic principle behind this crossover operator is that new solutions are generated by recombining parents using a different operator, similar to the strategy used in conventional evolutionary algorithms. If the new solution is an improvement on the existing one, as measured by the objective function, it forms a member of the new population replacing the existing *pbest* – otherwise the new solution is discarded. Rather intriguingly, it is the particle's best which receives modification rather than the best particle's current position; although not explicitly stated, the rationale would appear to be that this has the effect of giving the particle two simultaneous ways of finding an improved solution and is an attempt to introduce an adaptive learning process to the *gbest*. This development preserves the velocity equation and therefore the swarm's collective cognition. The recombination operator means that the solutions effectively perform hyperspatial jumps in the selected dimensions, potentially disconnecting them from the converging swarm. The tendency not to replace good solutions with inferior solutions, combined with hyperspatial jumps, also means potential convergence difficulties in a solution space with small feasible regions or a variable contour landscape.

Velocity Equation Modification

In the discussion above, we have considered modifications comprising the addition of operators derived from other optimisation techniques could be considered as 'tinkering' with the convergent properties of PSO. In this section, we will discuss suggested variants of PSO that are founded on analysis of the basic equations. It should be made clear that the significance of the modifications have mostly resulted from analysis prior to the modification being made, rather than a post experiment adaptation. For this reason, more weight should be placed on the validity of the arguments presented.

The first two velocity equation modifications, namely inertial weight and constriction factor, have already been discussed, together with time varying coefficients in Ratnaweera et al. (2004). They are reviewed below to complete the discussion.

The equations below are the canonical PSO particle update equations:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i)$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i$$

where $\vec{p}_i$ is the particle's previous best, and $\vec{p}_g$ is the global best, $\vec{v}'_i$ is the particle's velocity, $\vec{x}_i$ is the particle's current position, w is an inertia weight φ_{1i} and, φ_{2i}, are the acceleration coefficients. Following experimentation, Shi & Eberhart (1998b), found the best results were gained by reducing the inertia weight from a starting value of 0.9 to 0.4 over the course of a run. This relatively simple change has led to its wide adoption, because it allows the swarm to converge.

Through the use of formal mathematical proof, Clerc & Kennedy (2002) showed that the constriction factor produces particle behaviour which is guaranteed to converge. The use of the constriction factor model also removes some of the parameter choice decisions from the implementer. Subsequent work reported by Eberhart & Shi (2000) comparing the velocity equation implemented with the inertia weight and constriction factor showed that the constriction factor resulted in a better rate of convergence on most test functions. However, for some test functions the constriction factor failed to reach the error threshold within a reasonable number of iterations; therefore *Vmax* was reintroduced, such that *Vmax* = *Xmax*, to avoid the swarm diverging too far from the optima. It was noted in Clerc & Kennedy

(2002) that the reintroduction of *Vmax* is presented as necessary, whereas in fact, as pointed out in Eberhart & Shi (2000), it only speeds up the inevitable convergence. With reference to van den Bergh (2002), Section 3.3, it should be noted that GCPSO, discussed in Chapter 2, is not guaranteed to converge on a global or even local optimum; it merely guarantees that the swarm will converge.

In sum, Clerc's substantive work formalised the relationship between the parameters of the velocity equation. Clerc also demonstrates a significant improvement in the algorithm's ability to find the optima when the constriction factor was tested on a suite of test functions. The problem still remains that there is no guarantee of convergence on local or global optima. In addition, an understanding of the interaction and relationship between particles in a swarm has yet to be defined.

Other modifications

We will now briefly discuss some of the other investigations carried out on the original particle swarm heuristic.

van der Merwe & Engelbrecht (2003) apply particle swarm optimisation to data clustering using the K-means technique (Lloyd, 1957, 1982). This presents a different model to the classic PSO implementation, which is not strictly consistent with the usual understanding of how PSO works. Rather than flying the particles through the solution space, the data vectors are brought individually to each particle representing a candidate cluster centroid; the data vector is assigned to the closest cluster centroid by Euclidian distance. Although the authors' description of the technique is somewhat incomplete (for example, is it possible to stop the swarm gravitating to a single centroid point?), an important question is this: will the clusters, in fact, be defined by the initial conditions in the swarm? The particle is a greater distance from most of the data values to be clustered resulting in a high initial velocity. The description given in the paper implies that a particle on the boundary of the solution space would only receive influence in the direction which leads away from the boundary. Since the author describes a process of allocating data values to a cluster centroid, we assume that the data value cannot be allocated to other cluster centroids in a given iteration. Returning to our boundary particle representing a cluster centroid, initial iterations are likely to produce a high velocity value in this centroid. Such a centroid potentially leads to a cluster made up of data values which are not necessarily close to each other. They just happened to be closer to the particle which started on the boundary, and while they are allocated to this cluster centroid they cannot be allocated to another cluster with which they may have more affinity. The centroid particle may thus come to rest at the equilibrium point between two clusters, with data values from each cluster forming an unnatural cluster around the candidate centroid, the member data values forming part of this cluster are also not able to join other potential clusters around other candidate centroids. Due to the relationships within the velocity equation between the particle's current position, its best position and the neighbourhood best, we conjecture that this effect will be more pronounced if the granularity of the solution space is coarse or the positions of the data to be clustered are widely spread.

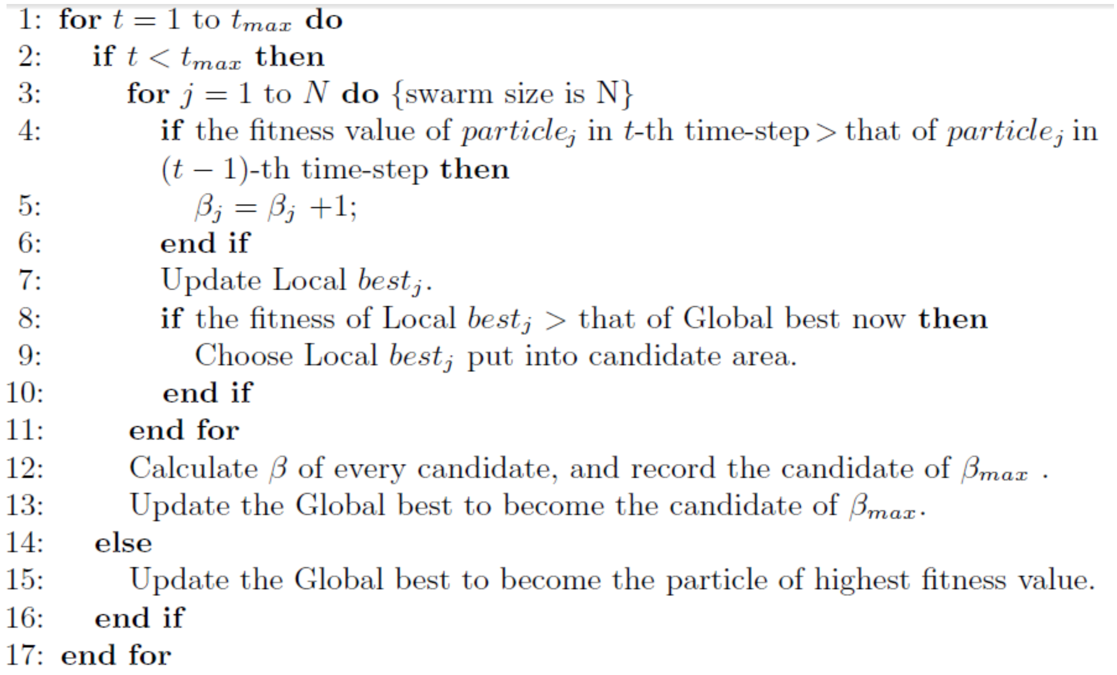
Another PSO clustering paper, Yuwono et al. (2014), addresses the limitations identified in van der Merwe & Engelbrecht (2003). This is achieved by developing the original algorithm presented by van der Merwe & Engelbrecht to reduce the number of evaluations performed each iteration. Specifically, the limitations, identified and addressed in the paper are: 1) high computational cost with high data volumes and/or high dimensionality, 2) the PSC algorithm identified in van der Merwe & Engelbrecht has a tendency to stagnate on clusters which represent local optima dependent on the starting position of the particles (Yuwono et al., 2012) – this observation mirrors our concerns on reading van der Merwe &

Engelbrecht, 3) the maximum number of clusters is pre-determined by the number of particles in the swarm. The paper introduces the following innovations to address the issues identified:

- each particle's position is updated once per iteration, after all particles closest to that particle are calculated
- the distance matrix is updated only after all particles have been updated
- a fitness function is used to minimise the sum of the intracluster distances
- each cluster is represented by a group of particles, allowing for an increase in swarm size without increasing the number of clusters
- two strategies to detect swarm stagnation and convergence are employed, to identify and address convergence on local optima

These innovations are then tested on seven datasets from the UCI machine learning repository against alternative algorithm. The innovations demonstrate an improvement across all benchmark datasets, whilst reducing computational cost, when compared with the alternative algorithms.

In contrast to van der Merwe & Engelbrecht (2003) and van den Bergh & Engelbrecht (2004), Abraham et al. (2008), develop a new variant of PSO, called MEPSO, which they test successfully on four real-world datasets. The key feature of MEPSO is the implementation of a growth rate through which the algorithm ensures that the new global best will be the particle best, which they call the local best, with the highest growth rate, as shown below:



```
for $t = 1$ to $t_{max}$ do
    if $t < t_{max}$ then
        for $j = 1$ to $N$ do {swarm size is N}
            if the fitness value of $particle_j$ in $t$-th time-step $>$ that of $particle_j$ in $(t-1)$-th time-step then
                $\beta_j = \beta_j + 1$;
            end if
            Update Local $best_j$.
            if the fitness of Local $best_j >$ that of Global best now then
                Choose Local $best_j$ put into candidate area.
            end if
        end for
        Calculate $\beta$ of every candidate, and record the candidate of $\beta_{max}$ .
        Update the Global best to become the candidate of $\beta_{max}$.
    else
        Update the Global best to become the particle of highest fitness value.
    end if
end for
```

Figure 7: MEPSO Algorithm reproduced from Abraham et al. (2008)

The advantage of MEPSO is that it preserves the particles with the greater potential to improve in addition to improving the solution fitness over the previous best particle.

van den Bergh & Engelbrecht (2004) develop a cooperative swarm model of PSO, in which each swarm optimises a single dimension of the problem. The best particle from each swarm is then combined with the other bests to provide an overall solution. The authors' motivation is that in standard PSO, improvements in individual dimensions may be lost due to cumulative deteriorations in other dimensions. This problem is common to other optimisation techniques, as discussed by Clearwater et al. (1992). Van den Bergh & Engelbrecht's approach to recombining the individual particles into a single solution is to provide a context vector, which is initially the overall *gbest* derived from concatenating the *gbest* from each swarm. In subsequent iterations, the dimension optimised by each swarm is updated with each particle in turn to update the swarm's *gbest*. This is a novel development, but the implementation, in which a single particle is updated at a time, ignores the potential for improvement in combination. The authors argue that their technique copes with deceptive spaces or limitations within the objective function. Arguably the implementation only addresses some issues: for example, if there exists a complex non-linear relationship between several dimensions, the individual swarms will not be able to represent this in terms of emergent behaviour within a swarm. As we shall see later in Chapter 5 , these issues can be addressed by making the search space more independent of the solution space.

A paper applying PSO to a clustering application takes a different, and in our view, more flexible approach. Li (2004) uses a Euclidean distance measure to identify clusters through the identification of species – hence, SPSO. Each species is defined as being those particles within a radius, τ , with the distance being measured as follows:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2}$$

where x is a vector, i and j are the indices identifying the individual particles and k is the dimension index, n is the number of dimensions and d is the distance function. One significant advantage of this approach over those discussed above is that it does not require the number of clusters to be specified. One of its shortcomings is that the radius represented by τ , which determines the species separation, needs to be set to a reasonable value for the given problem, requiring a prior understanding of the solution space. However it may be possible to derive a self-adaptive solution – this is an open research question.

Another paper, interesting for its potential for global optimality, is that of Xiao-Feng Xie et al. (2002). This uses a technique of mass extinction, which has the effect of resetting the population. Unfortunately, however, the description given in the paper implies that the whole of the population is reset, as if the swarm was being restarted. It is unclear whether the previous bests of the particles are retained during the reset. Rationally, one has to assume that the best positions of each particle are retained, since failure to do so would render it effectively a standard genetic algorithm: in a standard GA the previous history of a gene does not directly influence the mutation or crossover for the next generation – only the current generation has any direct influence. If it is the case that all the current positions of the particles are lost, this might provide the potential for global optimality. However, resetting the whole swarm, rather than, say, the worst 50%, undermines the strength of particle swarm optimisation, in that the convergence potential is lost.

Work by Secrest & Lamont (2003) provides a way to visualise the activity of a swarm during convergence, and also takes an analytical approach before suggesting changes to the algorithm. The stated aim of the paper is to use a visualisation technique to improve the performance of particle swarm optimisation by moving the swarm a Gaussian distance from the local and global best. However, the authors use a variant of the velocity equation:

$$V_{t+1} = wV_t \oplus C_1 rand(P_{\mathrm{lg},i} - X_i) \oplus C_2 Rand(P_{\forall g,i} - X_i)$$

where $\oplus$ is the vector addition operator. V Is the velocity of the particle, w is the inertia weight, C_1 and C_2 are the acceleration coefficients, X represent the particle's current position, P the best particles and $rand$ and $Rand$ are random numbers in the range 0..1.

$P_{\mathrm{lg},i}$ and are defined as "The neighbourhood (from i to g) best position" and "The global best position", respectively. If $P_{\mathrm{lg},i}$ is defined and implemented as stated, the velocity of a given particle should be expected to be more erratic than if it was the particle's own previous best

. It is unclear what affect this difference in the way the authors implement the standard versions of PSO equations has on the results presented in the paper.

A Taxonomy of PSO Developments

The following table draws together the taxonomy of early PSO developments we have established, in order to more clearly define our relation between the approach taken and the modification made:

	Hybrid: Genetic Algorithm Operators & Selection	**Velocity Equation Modification**
Heuristically or Trial and Error	Angeline, 1998b Higashi & Iba, 2003 Jian et al., 2004 Stacey et al., 2003 Wang & Li, 2004 Wen-Jun & Xiao-Feng, 2003 Ratnaweera et al., 2004	Ratnaweera et al., 2004 Shi & Eberhart, 1998b Peram et al., 2003
After Analysis of PSO	Clerc & Kennedy, 2002	Clerc & Kennedy, 2002 van den Bergh, 2002 van den Bergh & Engelbrecht 2004

Table 5: A taxonomy of the early developments in PSO.

The above table contains only those papers which have recommended specific changes to the original PSO equations and is not intended as a full analysis. The intention is to focus on the manner in which developments in PSO have been made. It is noticeable that the lower left box of the table, that which identifies the addition of an evolutionary operator after analysis, has no entries. This supports our conjecture that many of the amendments suggested resulted from the researchers' previous Genetic Algorithm and Evolutionary

Strategies knowledge, rather than understanding based purely on the behaviour of Particle Swarm Optimisation.

Conclusions

Developments to PSO have tended to introduce ad hoc amendments to the original PSO equations, tailored to certain specific problems. We have been critical of many of the developmental papers written on PSO. This has been for three main reasons:

- Much of the research into PSO incorporates other evolutionary based techniques, rather than identifying the strengths of PSO and building on these strengths. We suggest that this has led to modifications to PSO that are not easily transferrable to other problems.
- Many of the researchers have addressed the symptoms of premature convergence, rather than directly examining the causes inherent in PSO.
- Many applications of PSO have been to pre-process data in support of other techniques such as artificial neural networks, rather than using PSO alone for problem-solving.

We argue that there is significant potential remaining under-researched in the PSO heuristic. This potential is to be found in a greater understanding, and modelling, of natural behaviours, in order to produce a more flexible approach to optimisation, which can be done without undermining the strengths inherent in the velocity equation.

Finally, it is worth noting that the main developments in PSO have been evaluated against standard test functions, usually drawn from the genetic algorithm field. Very few applications have explored the use of PSO on a variety of noisy real-world data, and many of the real-world implementations of PSO to date have involved significant pre-processing of the data before applying the algorithm. Can PSO be made to discriminate effectively between wanted and unwanted data, noise, successfully?

Chapter 4

Innovations to PSO

In the first section of this chapter we offer reflections on the benefits of the PSO velocity equation and a principled analysis of how PSO actually performs as it converges on a solution. This then leads to a second section, in which we offer three innovations to PSO, each of which is then tested in the succeeding chapters.

The Value of Velocity

Previously, we have considered the developments to the particle swarm equations from the point of view of the convergent behaviour and subsequent performance of the swarm. As already discussed, arguably the most important of these developments in the early development of PSO are: the inertia weight variant Shi & Eberhart (1998b), improving the convergent behaviour of PSO; the constriction factor variant Clerc & Kennedy (2002) formalising the necessary relationships between the PSO parameters to affect convergence; and development of Guaranteed Convergence PSO (van den Bergh, 2002) – a variant which, the author argues, is guaranteed to converge on local optima. The inertia weight variant and the constriction factor variant were compared by Eberhart & Shi (2000), who argue that the constriction factor variant is a special case of the inertia weight variant, defining the optimal relationship for convergence between the parameters.

The Stages in Convergence

We will now consider the behaviour of the swarm, using data from the River Severn model, which is discussed in detail in Chapter 5. Four separate stages of convergence can be observed in the lifetime of a swarm. The exact point at which each stage ends and the following stage begins is difficult to define: this depends on the application. However, it can be argued that these stages are, in principle, identifiable. Figure 8 shows three graphs of converging swarms on different datasets from the River Severn. The graphs show the change in position of the *gbest* and its velocity as a proportion of the particle's position, simply stated as $\frac{|v|}{|p|}$ where v is the current velocity of a given particle and p is the particle's current position. By considering the velocity as a proportion of the change in position, one can see the particle's relative potential for change, and therefore its potential to improve on its previous best solution. In addition, it is possible to see from this which of the major influencing factors, the velocity or the current position, contributes most to the next position adopted by the particle.

The following stages four stages were identified from observation of swarm convergence sampled from experiments completed on all the datasets used in our research.

First stage: assuming a uniformly distributed swarm at the outset, two things are happening: firstly a particle's distance from the other particles' positions in the solution space is relatively large; and therefore, even given the reduction that the inertia weight applies, the particle will perform relatively large oscillations across the solution space. Secondly, the particle's personal best is moving rapidly through the solution space[6]. These two factors will combine to produce large changes in velocity, each particle building up an alternating positive/negative momentum.

Second stage: During this stage, the momentum of all particles in the swarm levels off, as the velocity reduces. This stage is characterised by a plateau in the velocity expressed as a proportion of the position for a given particle. (The plateau is relatively clearly defined in Figure

[6] We are using the phrase solution space as analogous to phase space in this context. The states which represent the candidate solutions are moving more rapidly through the phase space of possible solutions.

8c, but in Figure 8a it is slightly difficult to see, due to the complex nature of the graph; this section of the graph has been enlarged in Figure 8b to show iterations 150 to 175 and the plateau). The reason for these distinct changes is that the amplitude of the oscillations progressively reduces, due to the closeness of its relative position to its own previous best and the neighbourhood's best.

Third stage: We define this stage as the final stage before convergence, where the collective sum of the oscillations builds to a point substantial enough to start off a series of new oscillations. This is caused by outlying particles converging towards the *gbest* on different trajectories and finding improved solutions some distance from the previous best. The new momentum comes from the cumulative element of the velocity equation, so that a single improved solution found some distance from the existing *gbest* has the potential to significantly increase the momentum in the velocity for the next iteration, in turn increasing the exploration potential in subsequent iterations, before declining under the influence of the inertia weight. By referring to momentum in a particle's velocity we mean the existing value of v . The velocity equation can be described as an attractor and the behaviour is analogous to other attractors[7] such as the Lorenz attractor (Lorenz, 1963), defined by the following equations:

$$\begin{aligned} X' &= \partial(X - Y) \\ Y' &= -XZ + rX - Y \\ Z' &= XY - bZ \end{aligned}$$

where ∂ is the Prandtl number, $r = \dfrac{Ra}{Ra_c}$ Ra is the Rayleigh number, Ra_c is the critical Rayleigh number, $b = 8/3$, the geometric factor Tabor (1989).

[7] An attractor is a set of states towards which a dynamic system evolves over time.

These equations produce an oscillatory pattern between two centres of influence, which is similar to the behaviour of PSO when the momentum in the velocity equation nears zero. The question of using PSO to find the attractor for existing datasets – that is to use PSO to find a set of transformations for which the attractor is close to the original set – is potentially an area of further research.

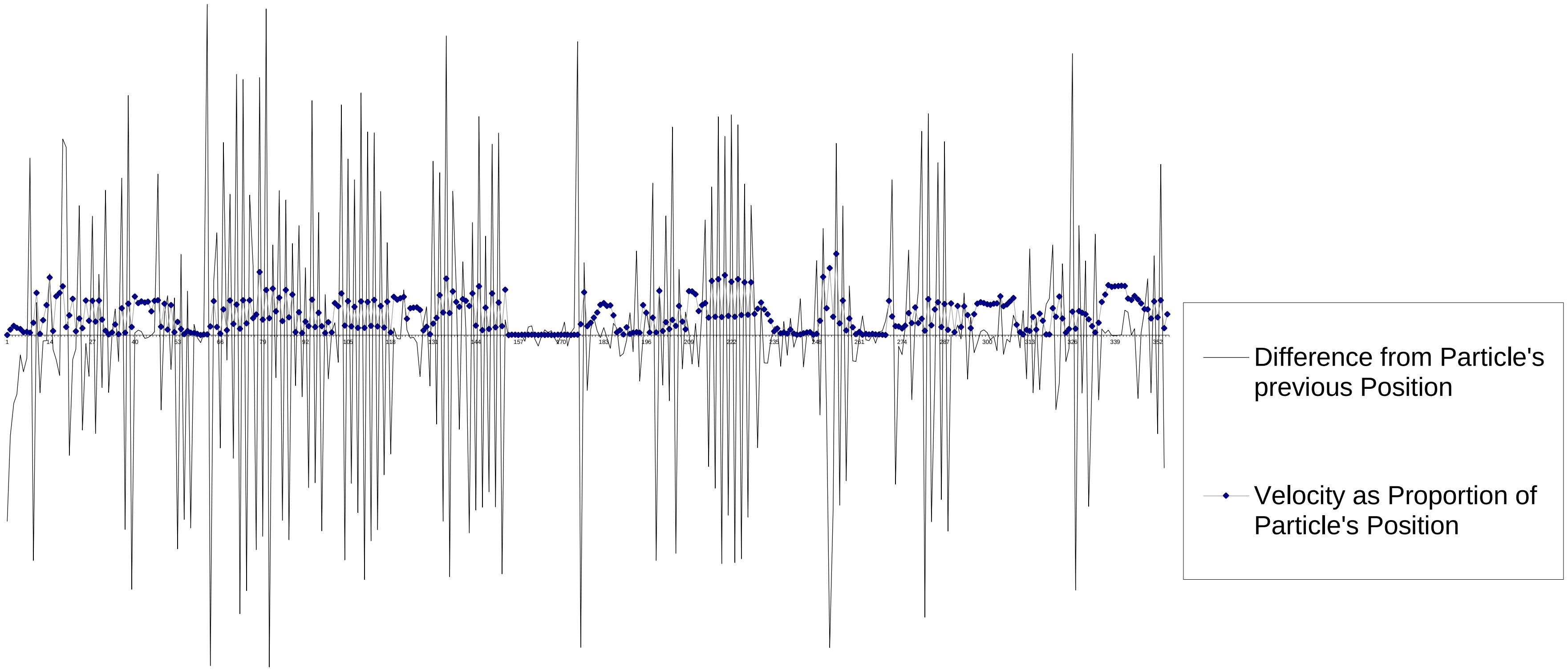
Final stage: this is a state in which the distances between the particles are so small relative to the size of the solution space that their influence on the velocity becomes negligible. At this point the swarm has converged and there is relatively little potential for any other particle to find an improved solution, and therefore to influence the swarm. The dominant influence on the velocity is now the reduction applied by the inertia weight, as described above. However, assuming the inertia weight has a value guaranteed to produce convergence, the momentum built into the velocity equation reduces to zero. In this phase, changes in velocity caused by the momentum have little influence on the particles' direction. At this stage, the swarm is tightly clustered, and the momentum is gradually reduced by the inertia weight or constriction factor variants of the PSO equations. (The constriction factor variant is a special case of the inertia weight variant, van den Bergh (2002)). Clearly, the stages we identify above are similar to the stages identified in Zhan et al. (2009). Whilst this is reassuring, our reason for presenting the detail above is to establish the theoretical basis for our developments in later chapters.

The swarm eventually reaches a stagnation state, with the velocity tending to zero, when the particles' positions become closer to the best known position. The best known position, however, may be far away from the true optimal value. We have shown that the reduction in a particle's momentum continues throughout all stages of convergence. In order to explore further, the swarm as a whole is dependent on the oscillatory nature of the particles' behaviour to build up momentum.

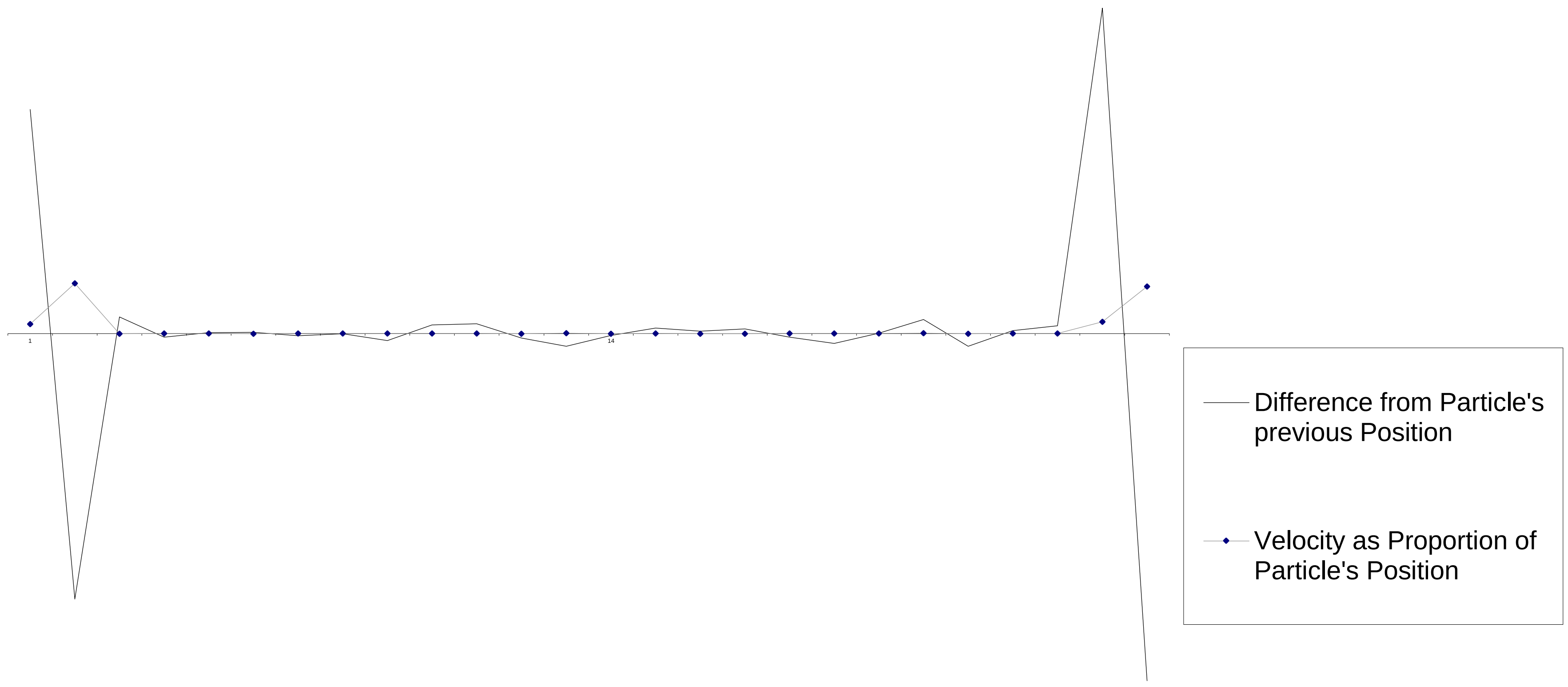
Given uniformly distributed starting positions and a direction of improvement, it is reasonable to assume the particle which is the swarm's *gbest* changes more often in the early stages of a run, due to the initially larger step size of all particles. In addition, assuming a uniform distribution in the initial widely-spread swarm, the *gbest* has a larger negative effect on a particle's velocity than a particle's own previous best. As the swarm converges, the new element of the velocity has a proportionally reducing influence on a particle's existing velocity. If a particle is to converge, it follows an oscillating trajectory with a decaying orbit around the influences of the *gbest* and its own previous best. This, in turn, means that the diminishing velocity of particles makes them less and less effective at exploration. The most significant influence on the velocity is the effect of the inertia weight, but this merely reduces the velocity to zero, limiting a particle's advance, which in turn reduces a particle's ability to escape the converged neighbourhood. The ability to explore beyond the converged neighbourhood is particularly important for real-world applications, for example, data such as EEG, temperature readings or hydrological data containing complex relationships between variables in real number space. Clearly, this points to a need to develop an extension to PSO that makes the velocity more independent of a particle's converged position.

A possible source of inspiration is Generalised Extremal Optimisation (GEO) de Castro & Von Zuben (2004). GEO was developed as a local search for bitstrings; the technique takes the weakest solutions in the population and probabilistically flips a bit, or bits, proportionally to the improvement it makes to the solution. This could be extended to PSO such that a move is made, per dimension, probabilistically proportional to the improvement it contributes to the overall solution. An alternative solution is to consider the abstraction of the particle solution space away from the problem being solved: for example, as in the hydrographical flow data model highlighted in this chapter, where the solution space in which PSO operates does not directly represent a solution in terms of hydrographical flow for the River Severn, but rather a matrix space for mapping inputs to the desired output.

(a)


1
14
27
40
53
66
79
92
105
118
131
144
157
170
183
196
209
222
235
248
261
274
287
300
313
326
339
352
Difference from Particle's previous Position
Velocity as Proportion of Particle's Position

(b)


1
14
Difference from Particle's previous Position
Velocity as Proportion of Particle's Position

(c)

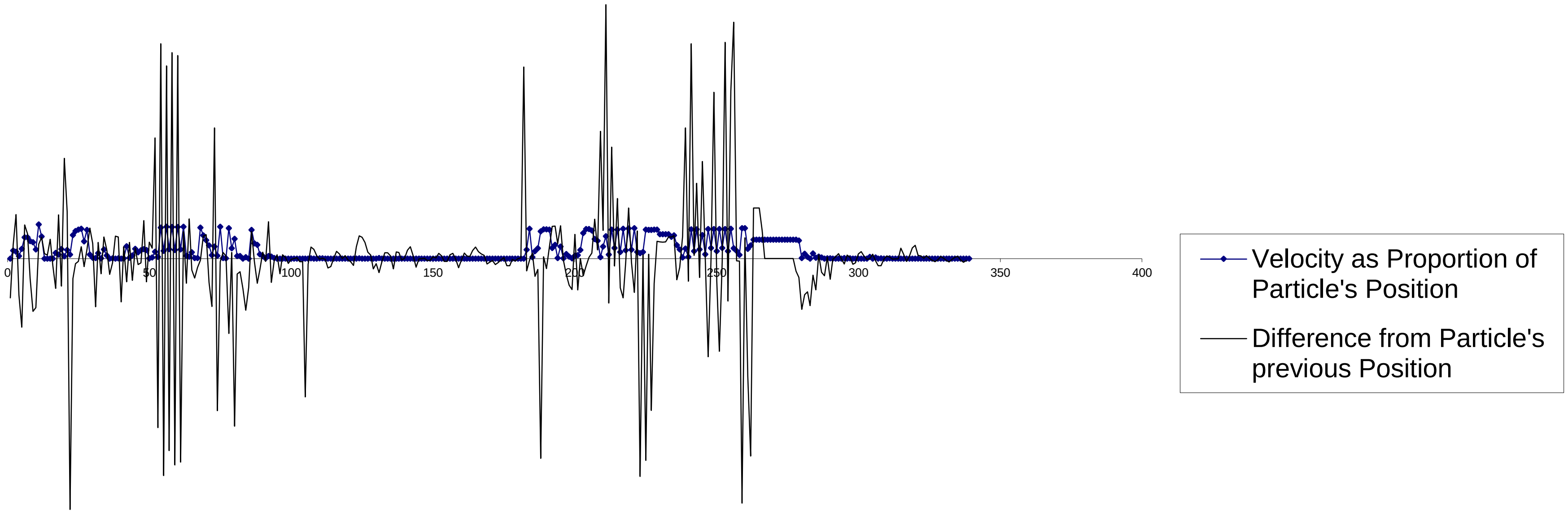


Figure 8: the convergent patterns of two swarms on different datasets. The respective figures show the difference in position (sharply oscillating line), and the velocity as a proportion of the position (smaller oscillating line), (b) shows an enlarged portion of (a) to highlight the plateau

Indeed, the convergence proof in van den Bergh (2002) identifies one of the significant intrinsic characteristics of PSO, when compared with other heuristics such as GA, simulated annealing or branch and bound, (these techniques are discussed in Corne et al. (1999)). As PSO converges towards an optimal value, it is being directed by the shape of the landscape it is searching. GA, by contrast, converges due to the retention of previous experience, achieved through the offspring of bitstrings, or the retention of the fittest parent strings from previous generations. Improved solutions are found almost by accident through the randomised recombination of existing solutions Goldberg (1989). In our opinion PSO's direct connection with the solution landscape is a significant strength of swarm computation heuristics. It is of course possible to argue that this would potentially make PSO more susceptible to a deceptive landscape. However, we suggest that because PSO learns directly from the shape of the solution landscape itself, at least one of the particles within a swarm will avoid being deceived.

The PSO process of finding optima

In this section we will consider the behaviour of the swarm in cases where the function to be optimised is a minimisation function: for example, where the aim is to minimise the distance travelled or costs incurred. As discussed previously, the equations for PSO are as follows:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i)$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i$$

where $\vec{p}_i$ is the particle's previous best, and $\vec{p}_g$ is the global (or neighbourhood) best, $\vec{v}'_i$ is the particle's velocity, $\vec{x}_i$ is the particle's current position, w is an inertia weight φ_{1i}

and, φ_{2i}, are the acceleration coefficients. Recall that, to ensure convergence, the inertia weight w is conventionally set to 0.729 and φ_1, φ_2 to 1.5 after van den Bergh (2002), consistent with Clerc & Kennedy (2002) and Eberhart & Shi (2000). The velocity will always reduce towards zero with a reduction of 0.729 of its previous value per iteration. As the swarm converges the increase in velocity also becomes minimal.

Assuming for the moment that the *gbest* particle does not change its best, the remaining particles will still converge to the *gbest* point, eventually reaching the same state of zero velocity. This is compounded if the *gbest* is part of its own neighbourhood since the *gbest* particle is only influenced by its own best position. The effect of the *gbest's* inclusion in its own neighbourhood, influence on itself and ability to avoid stagnation are discussed further in Chapter 7, where we develop a dynamic neighbourhood structure based on the contribution made to a particle's improved position. In sum, the term 'convergence' misrepresents the swarm's behaviour: it has been assumed that convergence meant converging on local optima, a point made in Yong-ling Zheng et al. (2003). In fact, convergence should be understood as 'the swarm stops moving in roughly the same area'.

4.1.1 The possible advantage of Vmax

It should be clear from the above that a swarm converges more quickly at the beginning of a run. One of the effects of this is that larger, and potentially optimal, areas of the solution space can be 'flown over' and never exploited. By using $Vmax$ the velocity can be restricted, increasing the probability of optimal areas being found. Shi and Eberhart (1998b) note that not including a $Vmax$ parameter can lead to divergent particle trajectories resulting the swarm failing to converge at all. However, our initial trial experiments demonstrated that Shi & Eberhart's (1998a) suggestion to set $Vmax = Xmax$ also leads to the explosion of a swarm, wherever the solution space is large. Therefore, we use a value of $Vmax$ consistent with the combined observations in van den Bergh (2002), Eberhart & Shi (2000) and Clerc & Kennedy (2002), that of the range of the solution space consistent with implementing a torus boundary condition. Although this will lead to a slower convergence, this is not necessarily detrimental.

In summary, the characteristics of each of the four separate stages of development as the swarm progresses towards a converged state result from the relative influences of the various components of the velocity equation. A closer analysis suggests that PSO has several limitations after the swarm has converged to a point beyond which new improved candidate solutions, some distance from the current best solution, are not found. This was also shown to be a product of the velocity equation.

Innovations

We now introduce three innovatory changes to PSO, designed to meet some of the objections to existing research, as described in Chapter 3. These innovations are tested on real-world data in Chapters 5, 6 and 7 respectively.

Hyperspatial solutions

Rather than training a swarm in which each particle directly represents a possible solution to the optimisation or predictive problem in hand, instead a matrix of weights A is trained using PSO. The aim is for the swarm to find the matrix A in the first equation to yield a solution *r* in the second equation below.

$$s = \sum A \cdot f$$
$$r = \sqrt{s}$$

f is a vector of the (generally real numbered) problem inputs, s is the sum of the components of the vector, the square root is applied to scale the value down, reducing the effect of rounding errors in the initial calculations, and *r* is the proposed solution. A thus provides a hyperspatial solution to the problem, a search space which is abstracted from the problem data. By using a matrix of weights in this way we are better able to model the interactions between the inputs by retaining more degrees of freedom, which introduces greater scope for finding a combination of weights which in turn should improve the accuracy of the result.

This approach is tested in Chapter 5, with a problem of predicting river flow from historical results.

Hierarchical multi-swarms

4.1.2 Principles

The strengths of PSO can be utilised effectively to produce a dynamic optimisation heuristic and apply it to complex datasets, but the convergence process in the generic PSO algorithm, described in above, is such that only a single dataset can be optimised at any one time. However, multiple inter-related subsets are common in real-world data, in particular in chaotic systems analysis. This is usually a result of the initial data collection, but may also result from the need to isolate discrete events such as epileptic seizures or seismic activity from the rest of the data stream, or simply because the data stream is not continuous. One answer to this problem is to optimise more than one swarm.

Multi-swarm implementations were briefly covered in section 0. Our interest in the multi-swarm model is to simultaneously optimise a number of datasets, to improve on the results that could be obtained from a single swarm, or to work with time-series data in which a single swarm would be unrepresentative. For example, with medical data such as EEG[8] scans, several measurements are collected at intervals over a finite time period, and if two scans are conjoined, the model derived is based on an erroneous assumption that the data represents a continuous reading – a flawed model. Therefore, standard PSO can only accurately consider a single scan per patient. The inability to analyse multiple readings within a single dataset is a serious limitation of any optimisation algorithm, as there is no feasible method of aggregating data which has been collected in non-contiguous distinct

[8] Electroencephalography (EEG) is the recording of neuronal electrical activity along the scalp produced from within the brain.

subsets, such as EEG or river flow readings taken on different days. Clearly, treating such data as a single dataset risks distorting the result, since the distinct characteristics of each data subset are merged. It is also clear that to process data from a highly dynamic system, such the human brain, or a glacier, multiple data subsets are needed and the final results must be aggregated in order to produce a statistically meaningful result.

The multi-swarm technique preserves and extends PSO's advantage of being directly connected to the solution space. However, its major benefit is that it gives PSO the ability to work in a data space in which there are discrete subsets within the overall dataset. Multi-swarm PSO also, by its nature, allows different objective functions to be applied to the sub-swarms within the solution space. The objective function used might depend on the level of each swarm within the hierarchy; alternatively, each child swarm could operate with a completely different objective function: for example, each child swarm could optimise a different type of data. A child swarm's processing is then aggregated by the hierarchical super-swarm. One can imagine a multilevel hierarchical swarm applied to climatology modelling, for example, with each level of the swarm integrating a different type of data: for example, atmospheric and oceanographic, with the lower level swarms in the hierarchy producing optimal models for each type. Higher-level swarms would integrate the results, eventually arriving at an overall model of a climate system – the *complete* model being a bottom to top integration of the *gbest* solutions from each swarm. In addition, each parent swarm communicates its children, encouraging them to explore more areas of the fitness landscape. This maintains each swarm's cohesion and allows indirect communication between sibling swarms. This combination of factors improves the solutions found by the child swarms and hence improves the solution found by the whole hierarchy of swarms.

4.1.3 Dynamic hierarchical swarm implementation

To implement this model, we need to consider the behaviour of the proposed extension to PSO further, specifically the actions of the swarm at higher levels of the hierarchy. The canonical PSO velocity equation:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i) \quad \text{(a)}$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i \quad \text{(b)}$$

essentially acts as an attractor, drawing the particles towards the best positions. If the standard velocity equation is applied at each level, the convergence of the child swarms would potentially be hastened to a point within their respective solution spaces which was closest in value to the global best, perhaps the universal best, in the master swarm. This could lead to premature convergence, a factor which is of greater significance in the later stages of convergence. Furthermore, if the master swarm also operates as an attractor, it duplicates the actions of the child swarms and may lead to the swarm converging on a suboptimal solution, negating some of the advantages of using child swarms.

However, in the proposed hierarchical model an adaptation is made to perturb the trajectories of the super swarm particles to have a *repellent* rather than attractive effect. A repulsive implementation in the master swarm forces the *gbest* particle in each child swarm to explore new areas of its own solution space. This directly addresses the stagnation problem identified by van den Bergh (2002) and van den Bergh & Engelbrecht (2002). However van den Bergh’s solution, GCPSO, tackles the problem without being influenced by the nature of the solution space; we consider that every particle should have the opportunity to find improved solutions rather than only the *gbest*. Recall from Chapter 5 that at this stage of convergence the particles are in the same area of the solution space.

A dynamic hierarchical swarm is visualised as a swarm of the *gbests* of each child swarm, with the master swarm applying a repellent force on these, as illustrated in Figure 11. The particle swarm velocity equation needs to be reformulated to reflect this interaction. The modification is given below:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i + \vec{x}_i) - \varphi_{2i} \cdot (\vec{p}_g + \vec{x}_i) \qquad \text{(a)}$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i \qquad \text{(b)}$$

where $\vec{p}_i$ is the particle's previous best, and $\vec{p}_g$ is the global best, $\vec{v}'_i$ is the particle's velocity, $\vec{x}_i$ is the particle's current position, w is an inertia weight φ_{1i} and, φ_{2i}, are the acceleration coefficients. This equation modifies the canonical particle equation by reversing its addition and subtraction operators, leading to large increments in the step size for all the particles in the swarm, with the potential to reverse the sign of the velocity.

By taking the *gbests* from each child, the repellent equation pushes each of them further away from their best positions, encouraging further exploration but without losing the overall coherence of the master and child swarms. The child swarms continue to operate using the standard PSO equations. The master swarm is dependent on the $VMax$ parameter to prevent it from exploding. This has the later advantage of contributing more to the movement in the particle's position when the swarm has converged.

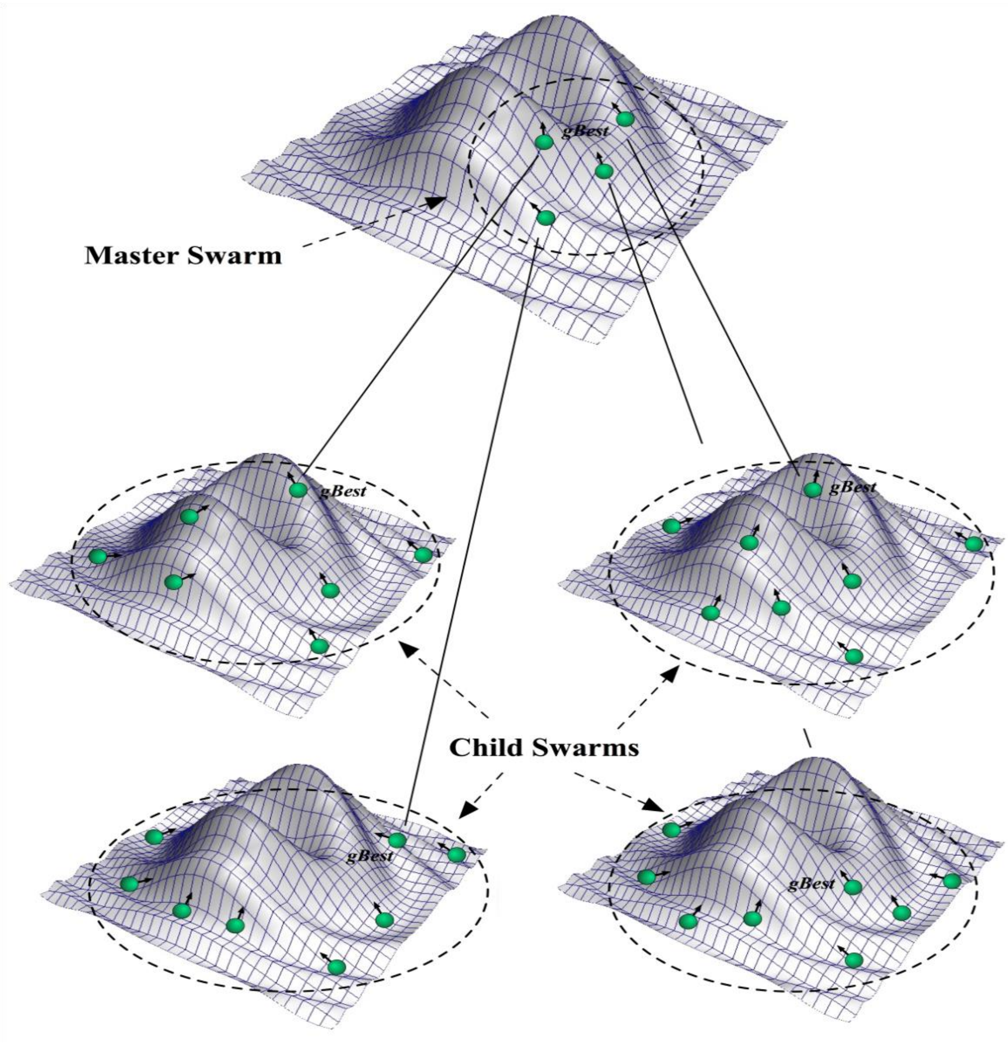


Figure 9: A graphical representation of the interaction between the Master and Child Swarms

If there is more than one level of hierarchy, the repellent force would only be applied to the *gbests* once per iteration in the master swarm, with the canonical velocity equation then being applied to each child swarm.

This amendment to the velocity equation applied to the master perturbs the oscillatory trajectories of the *gbest* particles between the particle's best and the *gbest* child swarms. Unlike van den Bergh's GCPSO solution to premature convergence van den Bergh (2002), which only changes the velocity equation for the *gbest* particle, the proposed solution maintains the particles' interactions within the particle swarm paradigm. It also has the advantage of preserving the stochastic influence characteristic of the original PSO equations, represented by the φ_1 and φ_2 terms, which enhance the exploratory characteristics of the swarm. It also differs from the Hierarchical Structure Poly-Particle Swarm Optimisation (HSPPSO) (Bo Liu et al., 2006) which implements bidirectional transfer of the best positions between the layers.

```
   Initialise child swarm particles to uniformly random
positions and velocities to zero

   Repeat

   Populate master swarm with the bests from all the
   child swarms

   Set best particle as global best

   For each particle in the master swarm do

      Calculate particle's velocity according to the
         equation (a) above

      Update particle's position according to equation (b)
         above

   End Do

      Copy the updated particle positions and velocities
   back to the respective child swarm from where the
   particle originated

   Repeat

         Calculate particle fitness in child swarms, if
      better than previous best change particles previous
      best

      Set best particle as global best

      For each particle do

               Calculate particle's velocity according to
            the canonical velocity equation

               Update particle's position according to
            equation (b) above

      End Do

   Until Termination condition met

Until Termination condition met
```

Figure 10: Pseudo code for Hierarchical Particle Swarm Optimisation

Our hierarchical swarm model is tested in Chapter 6.

Dynamic Neighbourhoods

As discussed in Chapter 2, a particle swarm implementation also supports the notion of neighbourhoods. For the purposes of the discussion that follows, a particle's membership of a neighbourhood is identified by particle identifiers rather than its topographical position on the landscape, meaning that the starting positions of particles in a neighbourhood may be topographically scattered. In our neighbourhood implementations, a particle is not considered to be a member of its own neighbourhood for the purposes of influencing its trajectory; this is in contrast to, for example, van den Bergh & Engelbrecht (2002) and Engelbrecht et al. (2005) where the *gbest* is included in its own neighbourhood. This means that the best particle will only be influenced by its own best position once, and receive a further influence from the best position of the second best particle in a neighbourhood.

A novel neighbourhood topology which is independent of, but can be combined with, the hierarchical swarm technique, has been developed. The aim is to provide a topology that is more responsive to the success of particles than the conventional static topologies, swarms or sub neighbourhoods within a swarm, and is inspired by the dynamic social groupings of humans or other higher species. We argue that particle swarm should have a neighbourhood model based on social groupings: a neighbourhood should be a network, the connections within which relate to the contribution that a particular particle makes to the improvement of other particles over time. This innovation is tested in Chapter 7.

4.1.4 Existing Neighbourhood Topologies

Each of the standard neighbourhoods we reviewed in Chapter 2 is based on the topology of the particles, so that the particles are linked normally by their index position to one or more other particles. For example, the *lbest* neighbourhood topology is one in which adjacent members of the population comprise the neighbourhood In a Fully Informed Particle Swarm (FIPS) every particle directly interacts with every other particle.

Our aim is to develop a neighbourhood that is based on network theory, where a particle's membership of a neighbourhood is based on the value of its contribution to other particles in the neighbourhood.

4.1.5 Network theory

Network theory is a branch of mathematics concerned with the study of the spread of information through an interconnected network. The concept of *small world networks* has been used by Watts & Strogatz (1998) to explore many phenomena, from information flow over the Internet, to the interconnectedness of Hollywood actors. Watts & Strogatz identify two factors which measure the interconnectedness of a network: the *length* and the *clustering coefficient*. When the network is viewed in graph theoretic terms, the length refers to the minimum number of edges from the set of all edges, E , required to connect two vertices, v_1 and v_2 . The clustering coefficient is defined as the average of the local clustering coefficients for all vertices, as described by the following equations Watts (2003):

$$N_i = \{v_j : e_{ij} \in E \wedge e_{ji} \in E\}$$

$$C_i = \frac{2\left|\left\{e_{jk} : v_j, v_k \in N_i, e_{jk} \in E\right\}\right|}{k_i(k_i - 1)}$$

where N_i is the immediately connected neighbours for v_i , e_{ij} is the edge between vertices v_i and v_j , k_i represents the number of neighbours for vertex v_i , $k_i(k_i - 1)$ represents the number of edges that could exist in the network, and C_i is the clustering coefficient for vertex, v_i .

$$\overline{C} = \frac{1}{n}\sum_{i=1}^{n} C_i$$

Interconnectedness is defined as the number of edges connected to a vertex, divided by the number of edges required for all the members of the network to be connected. A small world network is a class of random network which has both short path lengths and a clustering coefficient significantly higher than that of a random network.

Small world networks often occur in social and biological systems (Strogatz, 2004; Watts, 2003) where they are associated with more robust network topologies and rapid communication of signals across the network. As an important mechanism in PSO optimisation is the communication of the *gbest* location to many, but not necessarily all, particles, it seems reasonable to experiment with small world networks to model the neighbourhood topologies in PSO swarms.

Although PSO was originally developed for social interaction modelling by Kennedy & Eberhart (1995), a neighbourhood implementation from the perspective of network theory has never been considered. Our network theory inspired, small world model attempts to capture the group's cohesion in a way which is related to a particle's performance, rather than on a Euclidean distance measure: the neighbourhood does not depend on the particles' indices, but dynamically relates to a particle’s performance within the group. Therefore, particles effectively able to improve the quality of their neighbour’s solutions rather than waiting for those neighbours to improve.

The guiding principles of this new formulation are:

- It is preferable to have connections than not to have connections. To use an analogy, it is better to have friends from whom you can learn, rather than not to have friends.

- It is better to recruit a higher ranked particle than a lower ranked one. Conversely, it is better to drop a lower ranked particle than a higher ranked one.

- After recruiting a new neighbour there should be a period of time before another neighbour is recruited. This is so that initially the influence of the new neighbour can be maximised.

- Conversely, after a particle is recruited there should be an increased probability that it will remain in the neighbourhood. This probability decreases over time.

The dynamic neighbourhood offers a way to approximate interactions between individuals based on the value they place on each other within their local group. It should be stressed, of course, that this is merely an analogy, and not a complete model of social interaction. However, the elements of the equations that follow attempt to represent certain elements we have identified as being important in social interaction.

The model proposed here consists of two formulae – one to recruit, and one to drop, a neighbour, given specific small probabilities. The formula below is used to determine if the particle will join a neighbourhood.

$$\left(\frac{1}{k}\varphi_1 + \left(1 - \frac{1}{t - \tau}\right)\varphi_2\right) > \varphi_3$$

To recruit a new neighbour, a prospective neighbour is selected such that, $p \notin N$, where p is the prospective new neighbour particle and N is the neighbourhood of the current particle.

To drop a particle from a neighbourhood, a random particle is selected such that, $p \in N$, where p is the prospective neighbour to be dropped and N is the neighbourhood of the current particle. The formula below is then used to determine if the particle will leave the neighbourhood:

$$\left(\left(1 - \frac{1}{k}\right)\varphi_1 + \left(1 - \frac{1}{t - \tau_r}\right)\varphi_2\right) > 2\varphi_3$$

where *k* is determined by the fitness of the particle to be recruited or dropped, *t* is the current iteration, τ_r is the iteration at which the particle considered for removal was recruited as a neighbour, and τ is the iteration at which the recruiting particle last recruited a neighbour. The three random values ϕ_1, ϕ_2, ϕ_3 are selected uniformly from the range 0..1 influence the probability of a neighbour being dropped or recruited. In the formula to drop a particle, ϕ_3 is multiplied by 2 to reduce the probability that the result of evaluating the equation will result in a particle being removed from the neighbourhood. Conversely, in the formula to recruit a particle, ϕ_3 is not multiplied by 2 in order to increase the probability that the evaluation of the formula will result in a new neighbour being recruited.

Each of the two formulae above has two main elements: the first element to select a particle based on its ranking, and the second element to introduce a time delay to slow down the frequency of recruitment of a particle to, or removal from, the neighbourhood. The three random coefficients in these equations allow for a small, but increasing over time, probability of recruiting or dropping a neighbour. The neighbourhood of a given particle is updated on each iteration of the PSO process, and the particle's neighbourhood is then used to update the velocity equation before the formula to remove the worst performing particle from the particle's neighbourhood is applied. If there is only one particle left in the particle's neighbourhood, then there are two implementation possibilities: one is for the last remaining particle in a neighbourhood always to be retained; the second is that the *gbest* particle could be used in the place of the neighbourhood best in the velocity equation. The option of retaining a minimum number of particles in a neighbourhood has the advantage of retaining the neighbours which have previously influenced the particle, and this is the option we have implemented in our experiments.

Such a dynamic neighbourhood model should improve the exploitation potential of the swarm by enhancing the information flows between interconnected sub-networks. The existing PSO neighbourhood topologies, with some exceptions – for example: Janson & Middendorf (2005) and Clerc's TRIBES (2006) – implement static topologies. In social networks individuals make choices based on the value they place on another individual's contribution to their network, and so connections between individuals are made and broken over time (Watts, 2003, 2004; Watts et al., 2002).

To summarise, for the dynamic neighbourhood, once per iteration the formulae for the dynamic neighbourhood were applied as follows:

```
For each particle do

   Probabilistically add particle to the
   particle's neighbourhood

      Retrieve a copy of the particle's
   neighbourhood

      Probabilistically remove a particle from
   the particle's neighbourhood

      Return the copy of the neighbourhood to
   update the velocity equations

End Do
```

The above illustrates the process of identifying a particle’s neighbourhood per iteration. After this has been completed, the new neighbourhood is used to update the velocity equations, using the inertia weight variant given below:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i)$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i$$

Our dynamic neighbourhood model is a significantly different topology to that of the dynamic neighbourhoods in Emara (2009), Hu & Eberhart (2002) and Akat & Gazi (2008), because we allow a particle to select its neighbours based on the performance of the prospective neighbour. Although Janson & Middendorf (2005) suggest an adaptive hierarchy of particles, their neighbourhoods are defined by particle index, not location in the fitness landscape. The notion of hierarchy should be treated as if the levels in the hierarchy were analogous to separate social networks, introducing an element of interaction between the particles beyond the particle’s own neighbourhood, and using the notion of *scale* introduced by Watts et al. (2002). An analogy would be the difference between a social and work life. The interactions between both areas influence each other, but happen on different scales with different degrees of interaction.

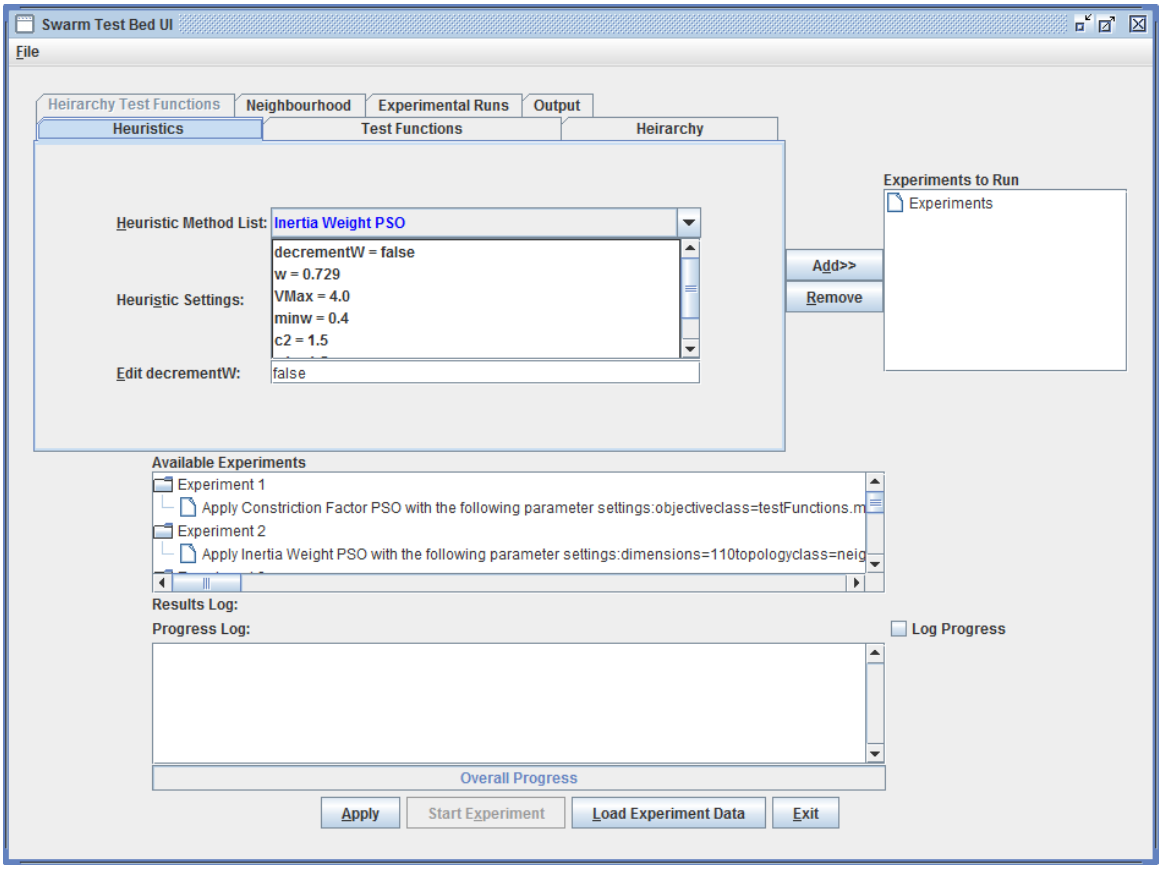
Testing the innovations

In the following chapter, we demonstrate a model for the hydrological flow prediction using River Severn data and demonstrate a way to abstract the problem landscape from the landscape searched by PSO. Our analysis of the behaviour of the velocity equation in the course of PSO’s convergence is extended in Chapter 6 into a hierarchical model to address issues of premature convergence. Finally, Chapter 7 tests out our suggested neighbourhood model.

We base our experiments on real-world data which by its very nature includes noise. The experiments that follow address the question as to whether our techniques, discriminate between noise and descriptive data successfully.

Experimental methods

All of the experiments completed during the course of our research were run on a custom designed test bed written in Java. The test bed allows flexibility in the configuration of PSO, which is used and allows the dataset to which the swarm as applied to be selected. The following figure shows a screenshot of the test bed:



Heuristic Method List: Inertia Weight PSO

Heuristic Settings:

decrementW = false
w = 0.729
VMax = 4.0
minw = 0.4
c2 = 1.5

Edit decrementW: false

Available Experiments

Experiment 1
Apply Constriction Factor PSO with the following parameter settings:objectiveclass=testFunctions.m
Experiment 2
Apply Inertia Weight PSO with the following parameter settings:dimensions=110topologyclass=neig

Figure 11: Screen capture of test bed written to complete our experiments

After an iteration is completed, the test bed allows the results from the best performing particle or all particles to be stored in XML format for subsequent analysis.

All swarms used the hyperspatial transfer functions described in section 4.5 above, except for the highly complex Parkinson’s data in section 7.1.

Unless otherwise stated, all experiments used 1000 steps for the PSO to find the solution. Each experiment was repeated for 20 runs, with the swarm giving the best results from training data used for generating the presented results using the testing data.

Each PSO model uses the canonical inertia weight variant of the equation for all swarms:

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i - \vec{x}_i) + \varphi_{2i} \cdot (\vec{p}_g - \vec{x}_i) \quad \text{(a)}$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i \quad \text{(b)}$$

In chapter 6, the master swarms in the hierarchy use the modified equations from section 4.6 above, reproduced below.

$$\vec{v}'_i = w \cdot \vec{v}_i + \varphi_{1i} \cdot (\vec{p}_i + \vec{x}_i) - \varphi_{2i} \cdot (\vec{p}_g + \vec{x}_i) \quad \text{(a)}$$
$$\vec{x}'_i = \vec{x}_i + \vec{v}'_i \quad \text{(b)}$$

The parameter values used in all experiments are given in Table 6, after Shi & Eberhart (1998b) and van den Bergh (2002), to ensure the swarm converges. These parameters are used for all swarms, including both the master and child swarms in the hierarchical PSO experiments of Chapter 6.

Parameter	**Value**
w	0.729
φ_1	1.5
φ_2	1.5
$VMax$	4.0
Particles	40

Table 6: PSO parameter values used.

In addition, to the parameters given above and consistent with Helwig et al. (2013), which shows that no boundary condition implementation outperforms another on a suite of test functions, we implement a boundary condition consistent Infinite model, in which the search space effectively wraps as if toroidal. Our reasoning for this is that we effectively abstract the search space from the solution space in our model, and therefore by wrapping the search space in a continuous loop, we avoid making assumptions about the characteristics of the search space and retain our problem independence.

In addition to the PSO parameters, we need to introduce some parameters for the dynamic neighbourhood, providing a framework for the recruitment of particles into a neighbourhood. For our experiments the dynamic neighbourhood parameters were set as follows:

Parameter	**Value**
Initial number of particles in each neighbourhood	4
Minimum number of particles in a particle's neighbourhood	5
Maximum number of particles in a particle's neighbourhood	10

Table 7: Dynamic neighbourhood parameter settings used in the experiments.

These settings were derived by experiment, in order to maintain sufficient influence for a particle to converge whilst restricting the maximum neighbourhood size. We recognise that this is a pragmatic decision but the aim of our experiments is to explore the effect of neighbourhood membership, a restriction on the maximum neighbourhood size avoids the neighbourhood growing to include every particle, a potential drawback of FIPS, and the neighbourhood topology being used for comparison.

Chapter 5

Hydrographical flow prediction of the River Severn using PSO

This chapter presents experimental results from an implementation of PSO used to predict hydrographical flow in the River Severn. This implementation uses the hyperspatial transfer function introduced in section 0. The aim is to produce projections at least as good as simple multiple linear regression, as proof of concept for a generalised method of modelling hydrographical flow using PSO. The data used is from the Centre for Ecology and Hydrology Plynlimon Research Catchments (Kirby et al., 1991). The results represent a significant statistical improvement on the results presented previously using the same data.

Flow Data

The Centre for Ecology & Hydrology in the UK, www.ceh.ac.uk, has been recording the flows from the Severn River and its three tributaries, the Hore, Hefren and Tanllwyth – referred to as the Plynlimon Research Catchments – over a period of years. The purpose of the research supported by the Centre has been to collect data for comparison between forested and grassland catchment areas, based on data collected from these between 1972 and 2004 (Kirby et al., 1991; Mount & Abrahart, 2011). In this chapter we present the results of our application of PSO using a subset of this dataset covering 1980 to 1990, the purpose of which is to develop a predictive hydrological flow model for the River Severn based on the historical data. Figure 12 shows a map of the catchment area.

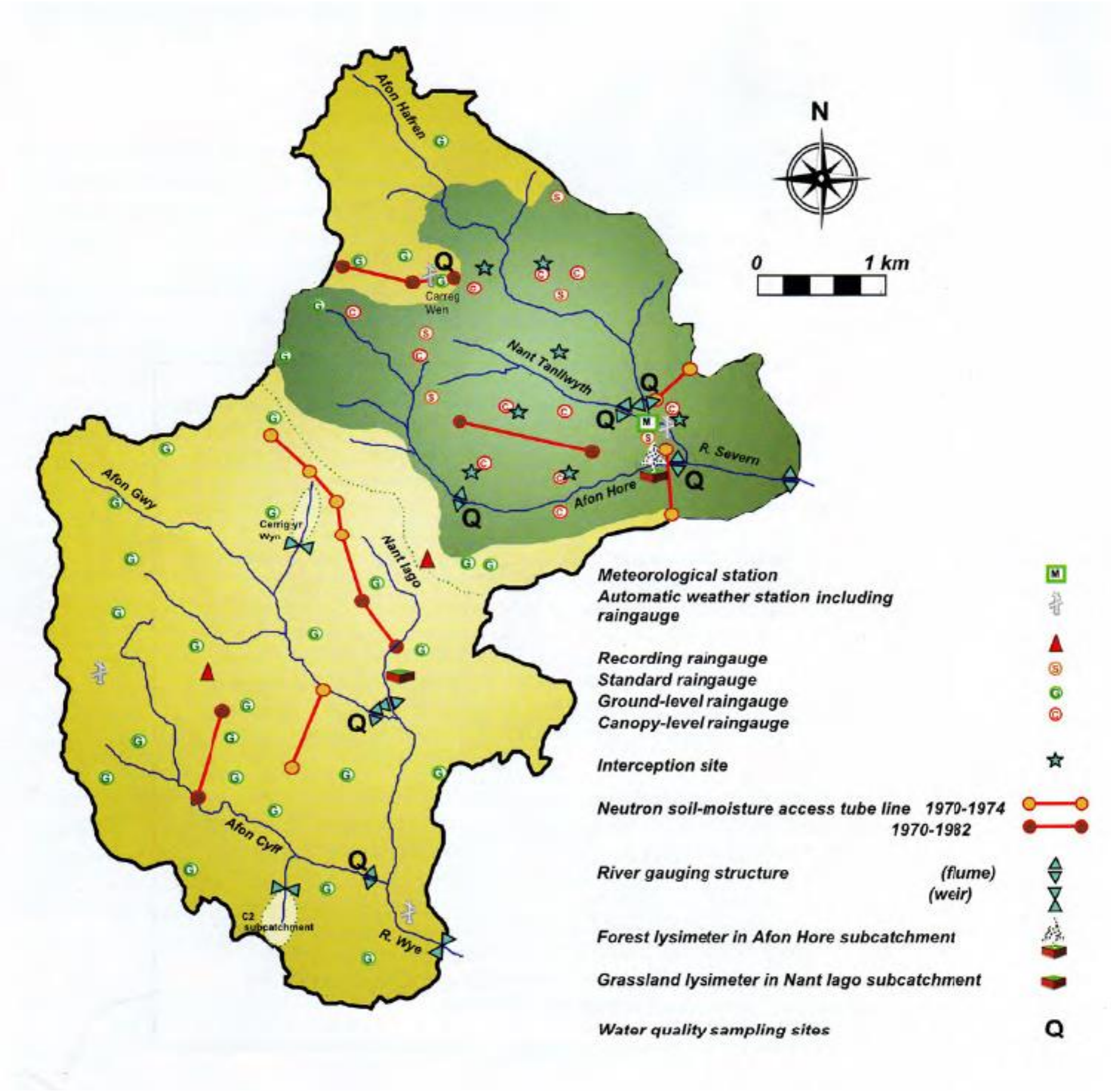


Figure 12: Plynlimon catchment, shown in darker Green

First, it is necessary to introduce some technical terms which are used below. In Hydrology a *catchment* refers to the drainage basin or area of land where surface water, rain or snow melt converges at the exit of the basin. *Run-off* refers collectively to all of the water which enters the river system from the surrounding hills. *Lag* refers to the time after a rainfall event: for example, until the run-off has entered the river system and has an influence on the hydrological flow. *Dry bulb temperature* refers to the air temperature taken with thermometers shielded from moisture. In addition, we will refer to certain events as occurring at time *t*. In order to be an effective predictive model the model must be able to predict events at time *t* using data available before time *t*. Much of the existing hydrographical predictive modelling using artificial intelligence (AI) techniques do not achieve this: rather, they model a specific event using the data values taken at the time the event occurred. There have been no attempts to model River Severn flow using either AI or non-AI techniques and, interestingly, no other attempts to model the River Severn using the data we have from the Pylynlimon research data. Obtaining a successful model would mean being able to predict flood events. Of the non-AI hydrographical flow models, the following are of interest: Domniţa et al. (2009), Ward et al. (2009) and Shin-jen Cheng (2010). These papers all develop specific models for each river basin being modelled and do not attempt to make predictions, choosing instead to model a particular event. The models are evaluated by how closely the model fits the actual event.

There have been a number of experiments which attempt to model hydrographical flow using artificial intelligence techniques other than PSO. These mainly rely on Artificial Neural Networks (ANN) and model the outflow from the basin at time *t* only on the basis of data from the run-off at time *t*, without allowing for accumulation or lag. For example, Turan & Yurdusev (2009), Kerh & Lee (2006) and Imrie et al. (2000) depend on predicting hydrological flow at time *t* using recorded values also from time *t*. This has no *predictive* value since the data values are taken from the time at which the event occurred.

A predictive model needs to be able to predict events ahead of time so that action can be taken to mitigate the effects of the predicted event. In this chapter, we develop a model using input data at *t* to *predict* the out flow at *t + 1* and *t+2*. In our models *t* refers to the day, so *t + 1* refers to the following day, and so on for *t + n*. This requires making predictions using data from readings taken before the event. Three significant recent papers in this area are: Tapoglou et al. (2013), Kuok et al. (2010), and Chau (2007), which use PSO to optimise a neural network were discussed in Chapter 3. However, Kuok et al. (2010) use data from time *t* only; we argue this has the same limitation with regard to the flood prediction as is the case with Turan & Yurdusev (2009), Kerh & Lee (2006) and Imrie et al. (2000). In addition, as we have previously argued in Chapter 2, for Eberhart & Kennedy (2004) PSO only acts to support another technique, in this case ANNs. Whilst this is a valid goal, it fails to exploit PSO's potential to find optimal solutions using more straightforward mathematical techniques.

A recent review of artificial intelligence techniques used to model hydrographical flow has been conducted by Mount & Abrahart (2011), which calls into question the validity of using such techniques to predict hydrographical flow. Obviously it is a significant challenge to the computational community as a whole to justify the use of AI techniques, in preference to linear regression, for example. As identified above, a shortcoming the techniques discussed in Mount & Abrahart, and the models developed there, is that they also use data taken at time *t* to predict events at time *t*, modelling a particular event rather than predicting future events. This is also a shortcoming of Kuok et al. (2010), the implicit criticism made by Mount & Abrahart (2011) of Turan & Yurdusev (2009). In their reply to Mount & Abrahart, Turan & Yurdusev (2011) argue that the purpose of their earlier work was to compare the techniques considered rather than produce a model which can be used to make predictions. However, our aim is to predict *future* hydrographical events. The application of AI techniques seek to predict well by including more degrees of freedom, which in turn allows the model to produce acceptable predictions on new datasets which are subsequently presented to the model. However, multiple linear regression is not effective in the context of prevention when there is a high degree of interaction between the independent variables (Keskin et al., 2013). Although we also use a linear model, we make the assumption that our model reduces overfitting. Any prediction model also assumes that the system on which prediction is being made is not random, and in this case, we assume that hydrological systems are chaotic in nature.

Hydrological Implementation

In order to predict hydrological flow for the Severn a matrix of weights is multiplied by the vector representing earlier flow readings, providing a hyperspatial solution in the form of the matrix, A (see section 4.5). The aim is for the swarm to find the matrix A in the first equation to minimise the error. A is also derived from a search space which is abstracted from the hydrological data:

$$s = \sum A \cdot f$$
$$r = \sqrt{s}$$

$$err = \sum_{1}^{n} |actualflow - r|$$

f is a vector of the real numbered hydrographical flow values from the River Severn tributaries, s is the sum of the components of the vector, square root is applied to scale the value down, reducing the effect of rounding errors in the initial calculations. r is the estimated flow for the River Severn. By using a matrix of weights we are better able to model the interactions between the inputs through retaining more degrees of freedom, thus introducing greater scope for the discovery of a combination of weights, and in turn improving the accuracy of the predictions. This resulting values of *r* are then combined to give a yearly sum, which is then subtracted from the sum of the real value where err the error is and n is the number of observations. The absolute difference is used to ensure that the minimum is zero; this of course means that a difference of -1 is treated as equivalent to a difference of +1. Each particle is given a random starting position in the range +2 to -2. The value of ±2 is the smallest integer required to allow the inputs to be mapped to the output. Once per iteration, the particles' candidate solutions are compared against the historical data. In the current implementation, a single swarm concentrates on a single year, with the swarm thus being run once for each year. Developing a model on a year's worth of data is useful, because it allows each year's model to be compared in subsequent analysis.

Discussion and Results

In assessing how predictable the hydrographical flow in the River Severn is, initially several models were considered. The available data from the hydrographical readings taken from the Plynlimon research (Kirby et al., 1991) were augmented by dry bulb and rainfall data. However, this led to inaccuracies in the predictions for the hydrographical flow through the river. It is reasonable to assume that this was because the weather data recorded the environmental conditions at the time the measurements were taken and did not take into account the time lag required for these to have an effect on the river system. Therefore, the final model was only based on the hydrographical flow through the River Severn's tributaries, ignoring weather and temperature data.

The hydrographical flow readings for a given day are averaged to provide a daily figure. The calibration of the readings from the Tanllwyth was reported as problematic by Marc & Robinson (2007). With reference to Figure 7 in Marc & Robinson, reproduced below, the discrepancies between the actual flow and the measurement recorded are characterised by a steady increase in flow for the Tanllwyth.

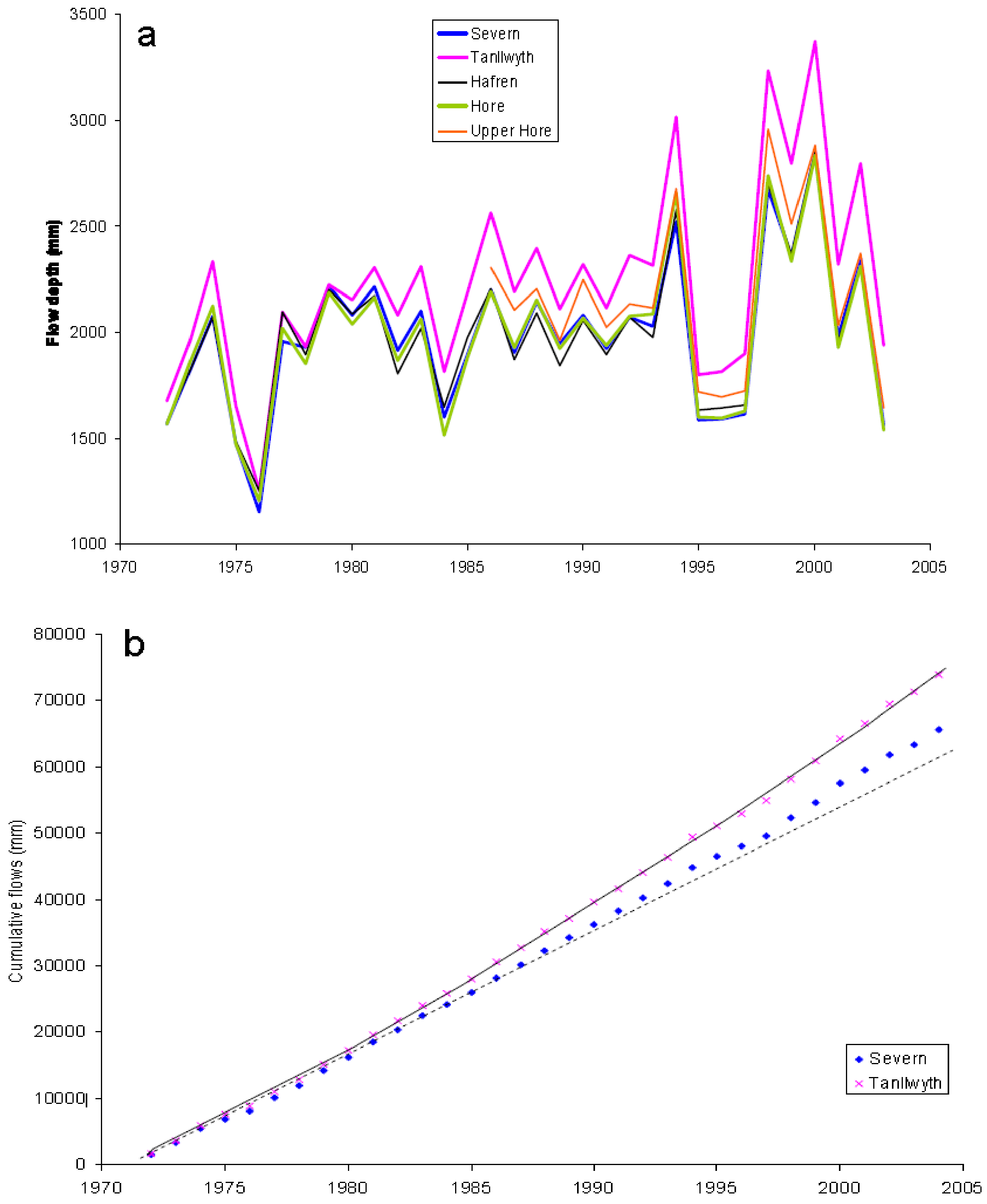


Figure 13: Reproduction of Figure 7 from Marc & Robinson (2007); (a) Time series of annual flows for the Severn and its sub-basins. (b) Cumulative annual flows for the Severn and the Tanllwyth.

Figure 13b shows the flows for the River Severn and Tanllwyth, the blue dotted line is the River Severn flow and the pink dotted line the Tanllwyth. The Tanllwyth flow is shown to be steadily increasing relative to that of the River Severn. According to Marc & Robinson, this was originally attributed to tree felling which started at about the same time. However, the flow continued to increase, pointing to erroneous measurements from the gauge. In addition, the Tanllwyth is enclosed between the Hore and Hefren rivers within the catchment (see Figure 12); thus the possibility of erroneous rainfall measurements or leakage can be eliminated, as this would also be seen in the Hore and Hefren data – see Marc & Robinson (2007). The discrepancies due to the erroneous flow measurements are more apparent in the period after the decade covered by these experiments. We took the view that, although measurements from this river were problematic, it was likely that they remained useful in producing a predictive model. Our rationale for this is that although the readings were incorrect, this was due to an incorrect calibration of the instrumentation and the readings retained a consistent pattern, which reflects elements of the actual flow. For this reason, they were included as inputs to produce the model. The data represents daily averaged flows for the River Severn tributaries. The input data was rearranged to allow the prediction of the current flow using input measurements taken at an earlier time. Each unit of time represents the flow for a day. Computer runs were completed using input measurements from *t* to predict flow at *t + 1* and *t + 2* respectively, therefore allowing predictions to be made ahead of time based on previous readings. The data presented in Appendix A gives sample results for *t + 2*.

Figure 14 shows a sample plot of the converging swarm for 1988, included to show the typical convergent behaviour of a swarm. The figure shows the position of the particle oscillating before converging on the optimum. This is a strength of the PSO paradigm, allowing a particle to 'fly' past its previous best to explore farther regions of the solution space.

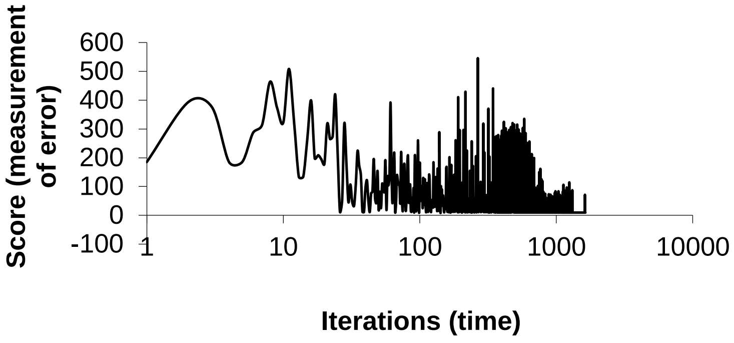


Figure 14: Sample convergence to optima for 1988.

Appendix A shows the results for the experiments using the hydrological data collected from the River Severn. All of the tables as referenced below can be found in this Appendix. The tables show the R-squared statistic values, the R-squared is a statistical measure of how well the model approximates the actual data points in the range 0..1; the closer the R-squared value is to 1, the closer the fit to the data. Table 12 shows the R-squared value for each year, using the matrix calculated for the corresponding year. The rows indicate the years, and the columns indicate the matrix used. The best performing matrix, that is the matrix with the least mean squared error, is shown as white text on a black background in Table 12, Appendix A. It is interesting to note that the matrix produced for a given year is not necessarily the one to produce the best result for that year: for example, 1980 and 1989 are not the best performers for any year. This suggests that there is a local optimum within the dataset, which prevents convergence on the true optimum. The matrix produced using the 1981 data, for example, produces the best results for 1981 and also for 1980. This is also true for 1989, which is outperformed by the 1988 matrix. It is important to recognise that the model has no knowledge of the data's origin, characteristics or accuracy. Therefore, the finding that a model developed from the data from another year produces better results than the native model for that year suggests the hydrographical flows in the river system are very stable, with external environmental factors having little effect on the hydrographical flow. The possibility that the solution matrices for the years that perform poorly are converging to locally optimal points suggests that using a hierarchical swarm model and sharing sample solutions between swarms during a training run may help find the true optimum for that year. Table 13 shows a comparison between the R squared results produced by PSO and MLR in every case the model is sufficiently accurate to predict the flows so the flow properties of the River Severn are stable over time. The model trained now should continue to be valid in years to come.

Table 14 shows the matrix which produced the R-squared values produced from our model and multiple linear regression, for each year. As already remarked, certain years do not produce the best performing matrix for that year. For further verification, the experiment was run twice more on 1980, to see if the matrix produced improved results. Neither of the runs improved, either for 1980 or another year, leading us to speculate that this is due to characteristics of the data for that year, which are yet to be identified. Table 12 presents a statistical analysis of the accuracy of each model for a given year. Appendix A presents a summary of the error statistics for each year. The years 1983 and 1986 are less predictable than the other years for which data was provided. The R-squared values for these years in Table 12 are lower, indicating that the flow is less predictable, when using the same model. From the matrices for these years in Table 14, it is difficult to conclude anything about the nature of the data. However, the result would indicate that this may be a result of the interdependencies between the variables for 1983 and 1986 being more complicated, or dependent on factors which are not included in the model, leading to a reduction in the convergence.

The size of the matrix by which the vector is multiplied arises mainly from historical development of the model. Originally, the dry bulb value and the rainfall data were included, providing five input values. As already stated, the number of rows represents the number of variables available to map a single value in the result vector; initially this was set as a 5 x 5 matrix. As with any regression, the number of variables has an influence on the achievable accuracy. When the model was reduced to the tributaries only, we initially ended up with a 5 x 3 matrix; later reducing this to a 3 x 3, to improve computational efficiency; this resulted in a typical loss of accuracy at 10^{-3} in the R squared value. With any model, there is a trade-off between accuracy and computational efficiency. Of course, once the matrix for a given year has been found, it is possible to combine the rows into a single weighting factor for each tributary. The matrix size determines the degrees of freedom available to the model: experimentation shows an average reduction in predictive accuracy of 0.7% between the 5 x 3 and 3 x 3 matrixes respectively.

The values in the matrix probably have no physical significance – the matrix is just an abstract agent of transformation. However, the values do model the effects of lag, evaporation, etc., inherent in the system. If this were not the case the model would not have the predictive accuracy demonstrated. Although the results are based on daily averaged flows, and probably do not model daily lag, the success of the model suggests that the time taken for the run-off to enter the river system, particularly that resulting from rainfall, is represented. One would expect a reduction in accuracy after a rainfall event if run-off was not being modelled. Indeed, this may have been a contributory factor to the reduction of model accuracy when the rainfall and dry bulb temperature were included, due to the time lag before the effects of evaporation and runoff enter the watercourse. Further, the results in Table 12 for these years are representative of every experimental run. This indicates the existence of features in the data which cause the swarm to converge to a local optimum. It is unclear why a model trained on one dataset would perform worse than a model trained on a second dataset when that model is applied to the first dataset, and when all the data comes from the same catchment area. A possible reason for this would be that a particular dataset contains local optimum values which cause the swarm to over fit. There are also differences in the numerical components of the matrices developed for each year. A suggested area of further investigation is therefore to analyse the shape of the data landscapes of each year and the factors influencing the convergence. Although, the components of the matrix do not represent physical features of the landscape or rivers, comparative analysis of matrices derived from different river systems might be illuminate hydrographical characteristics indicative of the *physical* landscape surrounding the river system.

The initial model is successful at improving on the results of existing hydrographical models, but the model should be further developed to enhance its applicability to a wider time range, this is further examined in Chapter 6. In addition to avoiding convergence to locally optimal values within a given year, a two layer hierarchical swarm version of the model is likely to lead to faster and possibly more accurate convergence, due to the exchange of data between the swarms which allows each to be guided by the experience of other swarms on datasets collected in different years.

5.1.1 Predicting floods

The data presented above shows an R-squared value of about .98 for the entire group of datasets. Q95 refers to the probability that a river's flow is exceeded 95% of the time, and therefore measuring the top 5% of flow, where flood events occur. The model was rerun using a dataset consisting of only the Q95 values.

The data was divided into training and test sets. Interestingly, and encouragingly, the model maintains an R-squared value of about .958 for each of the test datasets modelled.

This provides an improvement of .02 over a multiple linear regression on the training data, a statistic which was subsequently maintained on the test data. The improvement of 0.02 over the multiple linear regression was reported in a private communication by Nick Mount, Associate Professor of Hydroinformatics, School of Geography, University of Nottingham. The details of the model used to produce these results were not forthcoming. We therefore reproduced our own R squared model to verify this result. Using the Q 95 data for *t-2*, we were able to verify that they are more at least as good as the multi-linear regression model with an R squared value of .93. This is consistent with the results reported by Professor Mount, and indicates that our model produces sustainable improvement over other techniques even for the Q 95 data which in turn improves the ability to predict flood events.

As a final validation check, we compared the distribution of residuals (errors) from predictions made by multiple linear regression and the PSO predictors. An R-squared test on the residuals gave a result of 0.99, indicating that the residuals from both predictors were highly similar. We used a Kolmogorov-Smirnov test to quantify this intuition. If the distribution of residuals from both predictors were similar, we could say that the two predictors were behaving similarly. The Kolmogorov-Smirnov test is used to determine if two distributions are significantly different. (We considered using Student's t-test, but this test assumes that the distributions being compared are approximately normal, and the distributions generated by the predictors are not normally distributed.) Figure 15 shows the K-S results for the River Severn data:

Year	KS statistic	Critical value	Different? (95% confidence)	PSO	MLR
1980	0.112022	0.191379	Same	0.997	0.998
1981	0.123288	0.191379	Same	0.998	0.998
1982	0.134247	0.191379	Same	0.992	0.992
1983	0.142466	0.191379	Same	0.99	0.991
1984	0.155738	0.191379	Same	0.998	0.998
1985	0.021918	0.191379	Same	0.995	0.995
1986	0.29863	0.191379	Different	0.994	0.994
1987	0.060274	0.191379	Same	0.996	0.996
1988	0.180328	0.191379	Same	0.996	0.996
1989	0.320548	0.191379	Different	0.995	0.995
1990	0.128767	0.191379	Same	0.997	0.997

Figure 15: K-S River Severn Results Summary

Following best practice for the K-S statistic, we separated the residuals into equally sized buckets. The results show that the residuals are drawn from the same distribution and therefore support the R squared statistic analysis that PSO performs similarly to MLR on the River Severn data.

The results reported demonstrate the robustness of our method for abstracting the search space from the physical data to be modelled. Significantly, the results show an improvement over Mount & Abrahart (2011) and also demonstrate the robustness of our model, which uses PSO directly to produce a transformation matrix with better predictive validity than the artificial neural-network-based model in Kuok et al. (2010).

It should be noted that the technique for predicting flood values used by hydrologists could be improved. The hydrologists' models only include the values in which are interested in, rather than the complete dataset of measurements taken over the period of time. The R-squared statistic is also arguably the wrong measure to be using in order to develop a predictive model. Used in this way, the R squared value only indicates best fit for the selection of data used – in this case the flood values. In this case, it would be better to treat the problem as a classification problem with inputs classified as either flood or non-flood values. If successful, this would produce a model for predicting flood independent of the values on which it had been trained. Nevertheless, we have continued to use the R squared statistic in the fashion in which hydrologists use it in order to present comparable results.

Summary of River Severn Experiments

The aim of this chapter has been to present an initial account of the PSO techniques used to predict the hydrological flows of the River Severn from historical data. Essentially the results constitute a proof of concept which later chapters will develop. Key to this model was the abstraction of the search space from the solution space. We achieve this through the use of a matrix of weights which is then multiplied by the input vector of previous flow values. We demonstrated, through the use of the R squared statistic results, that our method outperforms previous results presented by Mount & Abrahart (2011). In particular, we draw the reader's attention to the results in section 5.1.1 which demonstrate a significant improvement on the ability to predict flood within the River Severn catchment. We also demonstrated that our model was able to find matrices capable of making accurate predictions on different datasets. This finding supports our hypothesis that abstracting the search space from the data space is of benefit when using PSO. We were able to show that the accuracy of our predictions held for *t+1* and *t+2*, improving on the results of other models such as that of Kuok et al. (2010).

The benefits of the model presented above are that it is relatively easy to implement and produces accurate predictions of the flow rate from historical data within approximately 300 iterations, completing the process of optimisation in a third of the time available. The result is a matrix for each year which generates a series of predictions which can be compared to the predictions made by other techniques. Furthermore the matrix is able to make accurate predictions on new datasets. However, a shortcoming of the model is that it is not very sensitive to extremes; this is possibly because the training dataset was too small and lacked the necessary variability in the data. These extrema normally result from periods of heavy rainfall which are not represented explicitly in the model. Early experiments showed that including rainfall and temperature data directly made the model less accurate, although clearly rainfall is a factor in the river flows

The results of the River Severn experiments demonstrate that PSO can outperform multiple linear regression. It also showed that the model developed on one training set can perform better than another model trained on another dataset. The analysis of the velocity during the convergence of the swarm also identifies transitions during the convergence process which can cause the swarm to stagnate prematurely (see Chapter 4). With these findings in mind, in the following chapter we develop a hierarchical model using a more complex dataset.

Chapter 6

Dynamic Hierarchical Particle Swarm Optimisation

The previous chapter demonstrated an improvement over the R squared values of multiple linear regression on the hydrological flow for the River Severn. This chapter introduces experimental results for a hierarchical extension to PSO. The generic particle swarm equations only allow one dataset to be considered concurrently. In this chapter we test a hierarchical particle swarm to accommodate discrete data sets that are part of a larger whole. The hierarchical swarm was described in section 0. The experiments in this chapter use two data sets: a glacier outflow data first presented in Stott & Mount (2007) and Mount & Stott (2008); and gas spectrometry data (Fonollosaa et al., 2015). We are using different datasets from the River Severn hydrographical dataset, because the experiments on this dataset show that a PSO implementation performed at least as well as multiple linear regression, and we therefore wanted a dataset which would enable us to show a clear difference, either better or worse, between the single and hierarchical swarm models. Although the results presented show an improvement, they do not outperform those produced by multiple linear regression. Although this may be considered a shortcoming of our research, especially as the results are from a single dataset, the significant improvement in the hierarchical results over those of the single swarm demonstrates the value of our technique on datasets with complex characteristics to which PSO may be applied.

Glacier Data

We shall first describe the data from Stott & Mount (2007) and Mount & Stott (2008), introducing some of the terminology. Figure 16 reproduced from Mount & Stott (2008) shows the area of the Ecrins National Park, France where the data was collected.

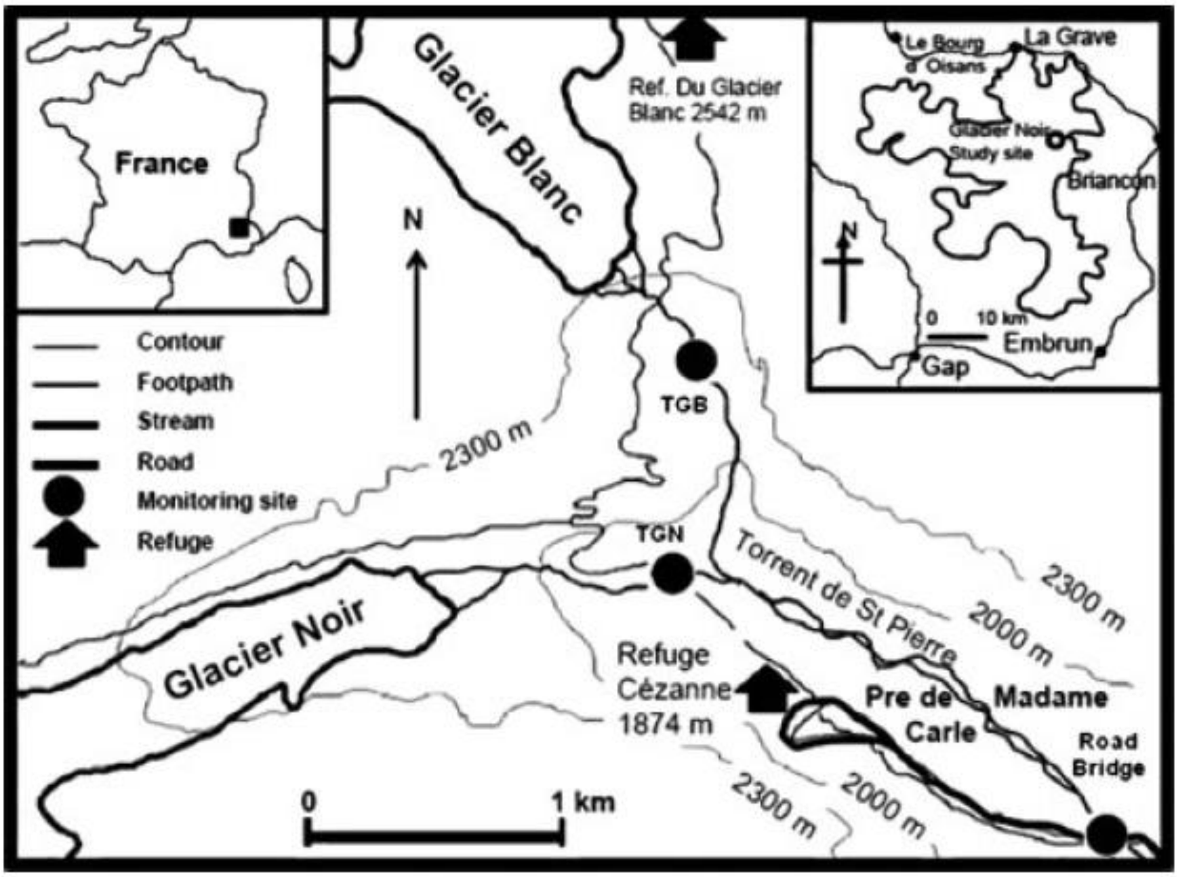


Figure 16: The glaciers Noir and Blanc and the Pre de Mme Carle. Reproduced from Mount & Stott (2008)

Our experiments use a subset of the glacial flow data originally published by Stott & Mount (2007, 2008), with the data covering 10th – 18th July 2005. The data was collected at two monitoring stations downstream of each glacier. Each monitoring station recorded the *suspended sediment concentration (SSC)* and *discharge (Q)* at 10 minute intervals. Suspended sediment is defined as sediment that moves in suspension in water and is maintained in suspension; discharge refers to the quantity of sediment transported in suspension (Colby et al., 1953). The air temperature was also recorded. Rainfall data was not included in the data provided, and therefore air temperature is the only independent environmental variable in the data. The emission of rainfall data is potentially important as its inclusion would provide a direct measurement of the amount of water falling as rain in the catchment, which was independent of the amount of water held within the glacier. The SSC discharge at the road bridge was also recorded. The aim of the study was to predict the discharge at the road bridge using the values taken at the monitoring stations as inputs. We also define *proglacial* as the area immediately in front of the glacier into which the melt drains.

As we argued in section 0, PSO, unlike most other optimisation techniques, is influenced by the landscape of the solution space being searched. Pulled in the direction of better solutions, the flight across the landscape is informed by the topography of that landscape. Extending this strength of PSO to a higher-level swarm means that discrete data subsets can be processed separately. The relationships between the subsets can be exploited to improve solutions within each subset.

6.1.1 The Need for a Hierarchical Approach?

Inter-related subsets are common throughout real-world data, in particular in chaotic systems analysis. The strengths of PSO can be utilised effectively to produce a dynamic optimisation heuristic and apply it to complex datasets, but the convergence process in the generic PSO algorithm, described in section 0, is such that only a single dataset can be optimised at any one time in order to maintain the separation of discrete measurements over a finite period. Thus our interest in the multi-swarm model is to simultaneously optimise different datasets, to improve on the results of a single swarm. For example, with medical data such as EEG[9] scans, measurements are collected over a finite time period. If two scans are conjoined, the model derived is based on an erroneous assumption that the data represents a continuous reading: treating such data as a single dataset risks distorting the result, since the distinct characteristics of each data subset are merged. Therefore simple PSO can only accurately consider a single scan per patient. This is a limitation in terms of being able to analyse multiple readings within a single dataset. It is not feasible to develop a generic method of aggregating data in cases where the data is collected in distinct, non-contiguous subsets, such as EEG or glacier readings. But it is also clear that in order to process data from a highly dynamic system, such as the human brain, or a glacier, multiple data subsets are needed and the final results must be aggregated in order to produce a statistically meaningful result. The multi-swarm technique preserves and extends the advantage PSO gains from being directly connected to the solution space, while giving PSO the ability to work in more generic forms of data space, in which there are discrete subsets within the overall dataset. The technique also, by its nature, allows different objective functions to be applied to the sub-swarms within the solution space. The objective function used might depend on the level of each swarm within the hierarchy; alternatively, each child swarm could operate with a different objective function: for example, each child swarm could be optimising a different type of data. A child swarm's processing is then aggregated by the hierarchical super-swarm. One can imagine a multilevel hierarchical swarm applied to climatology modelling, for example, with each level of the swarm integrating a different type of data – for example atmospheric and oceanographic. At the lower levels of the hierarchy the swarms produce optimal models for each type of data. At higher levels of the hierarchy the swarms work on integrations of the results from lower levels. As the sub-models are communicated up the hierarchy

[9] Electroencephalography (EEG) is the recording of neuronal electrical activity along the scalp produced from within the brain.

they are then integrated and further processed by the child swarms higher up the hierarchy in order to produce an overall model for a climate system; the *complete* model will be a bottom to top integration of the *gbest* solutions from each swarm. The technique proposed also maintains each swarm's cohesion. By applying this behaviour across the child swarms, the better particles within each child swarm are shown to successfully inform improvements to the best solution found within each child swarm.

6.1.2 Glacier Flow Implementation

The description of how hierarchical swarm works can be found in section 0. The experimental parameters and methods are described in section 4.9.

The experiments in Chapter 5 show that PSO converged very well on the River Severn hydrological flow data. Therefore a further improvement will be difficult to achieve, even with our new technique. In order to demonstrate the advantage of the hierarchical swarm technique, data that require subsets to be considered independently is needed. The glacial data satisfies this criterion, since the data collected is non-contiguous.

A relevant question to ask here is why glacier data is difficult to analyse. Recall the aim is to predict the discharge at the road bridge using the values taken at the monitoring stations as inputs. Predicting glacial flow is difficult because we lack a complete understanding of the factors affecting how such flows develop, combined with the difficulties of accurately recording the data. This often results in inconsistencies and incomplete datasets. However, there is significant value in modelling the dataset to obtain a prediction model from which a subsequent extreme events model could be developed.

As with ANN training (Haykin, 1998), using weights from multiple linear regression is prone to overfitting, and proves a poor forecast model due to the Mean Squared Error elimination techniques used (Tetko et al., 1995; Kuok et al., 2010). The convergence to optima is influenced by the previous best experiences rather than attempting to reduce the error explicitly.

6.1.3 Results

A transfer function similar to that used with the hydrographical flow data reported in Chapter 5. As with the River Severn data model, the implementation uses a position vector, where the number of dimensions is defined by 5 multiplied by the number of input variables, the vector is then reorganised into a matrix defined as $5 \times n$, where n is the number of input variables. The choice of 5 as a multiplier was chosen due to the success on the hydrographical data, giving 5 degrees of freedom per input given the loss of accuracy when a multiplier of 3 was used, (see Chapter 5). With the glacier data, we are trying to predict the SSC, given glacial discharge as the inputs, rather than the flow. We used the same hyperspatial transfer function as described in section 4.5, reproduced below:

$$r = \sum_{1}^{n} A \cdot f$$

deriving the predicted suspended sediment concentration, where A is the matrix defined above from the values of the particle’s current position – the values the swarm is optimising, r is the predicted SSC and f is the vector representing glacier discharge. This is a similar scoring function to the equation shown below except that n now represents the number of observations taken during a day. Unlike the River Severn model, the air temperature is included, as it is the only factor within the available data which represents a change in the environment. In Stott & Mount (2007), the authors comment that rainfall is not included because their aim was to record the SSC in the glacier's outflow. However, the rainfall value is not being measured in the SSC directly and therefore forms an independent indicator of environmental changes. The glacier flow measured includes the effects of rainfall, as we shall see, but this is combined with melt and therefore not independent of the SSC output measured.

In order to demonstrate the improvement in the predictive model produced by the hierarchical swarm compared with the single swarm model we again use the R-squared statistic, the single swarm model was run using the data from each day to produce benchmark results for 1000 iterations. Our River Severn experiments showed 1000 iterations to be sufficient for convergence on a good solution. The results were compared with running a hierarchical swarm of eight child swarms, one swarm for each day, and a master swarm, as described above, also for 1000 iterations. Although the number of iterations was set to 1000, both experiments found an optimal value significantly earlier. The solution space is also modelled such that it wraps in every dimension. The implementation has a boundary of $-20 < x_i > 20$ where x is the position vector.

Recall from Chapter 5 that the R-squared value is a measure of the goodness of fit between the predicted values and the actual values. An R-squared value closer to 1 is a better fit between the predicted and actual values and therefore indicates a better predictive model overall. The following table shows a summary of the results obtained for each day for both the hierarchical and single swarm models together with the R-squared results obtained from Multiple Linear Regression:

R-squared Summary

Date	Single	Hierarchy	Regression
10/07/2005	0.55	0.51	0.60
11/07/2005	0.27	0.27	0.43
12/07/2005	0.02	0.78	0.80
13/07/2005	0.74	0.74	0.79
14/07/2005	0.22	0.43	0.68
15/07/2005	0.71	0.72	0.77
16/07/2005	0.48	0.49	0.74
17/07/2005	0.48	0.48	0.61
18/07/2005	0.60	0.74	0.78

Table 8: Representative sample of results from hierarchical and single swarm experimental runs for each day.

Table 8 shows the R squared values produced from runs of the hierarchical swarm model and the single swarm model for each day. As with the River Severn results, we are using the R-squared statistic for comparison with hydrographical results, so the same caveats in Chapter 5 apply. Although only a single result is given for each day per swarm model, the relationships between the results are representative of all results. The hierarchal swarm results are better than the single swarm results however there is still a significant difference between the hierarchical swarm and multiple linear regression R squared results. This is a disappointing result, as we were expecting a closer match between the hierarchical R-squared results and those of multiple linear regression. However, it is interesting to note that the multiple linear regression results are also relatively low scoring. One of the reasons for this may be that the dataset does not include environmental measurements other than the air temperature. This would indicate that the glacier melt is affected by other variables which are not included in the dataset – for example rainfall, wind speed, light intensity etc. Nevertheless, our experiments do show that the hierarchical swarm variant consistently outperforms the single swarm variant on the dataset tested. The purpose of this experiment was to demonstrate that sharing information between swarms, using a hierarchical model produced a better outcome for all swarms. This observation builds on our findings in Chapter 5 that data sets which were not included in the training data for that swarm were outperformed by the model produced by the swarm. The glacier dataset is a more complex dataset to optimise, arguably because it omits environmental variables which may contribute to the SSC in the melt.

We completed a K-S statistic analysis. The results are given below. Again, the statistical results show that the errors for MLR and PSO are drawn from the same probability distribution, indicating that their performance is comparable on this data set.

Date	KS statistic	Critical value	Different? (95% confidence)
10	0.25	0.277608838	Same
11	0.295238095	0.13272241	Different
12	0.066666667	0.185072558	Same
13	0.38317757	0.131476163	Different
14	0.112149533	0.113333333	Same
15	0.271028037	0.113333333	Different
16	0.112149533	0.113333333	Same
18	0.341176471	0.147512711	Different

Table 9: K-S statistics results on the glacier data

6.1.4 Discussion

Of interest are the results for the 13th, 12th and 10th July 2005. In all experiments completed for the 13th and the 10th the single swarm slightly outperforms the hierarchical swarm model. It is instructive to consider what is causing this slight deterioration in performance, given that there are more influences trying to improve the values used to populate the transformation matrix. A. The hierarchical swarm performs less well on these two days when compared with the single swarm. However, the overall performance of the hierarchical swarm model produces better R squared values than the single swarm model. In order to understand why the hierarchical swarm performs less well, recall that the master swarm reverses the arithmetic operations in the velocity equation and that this is only applied to the *gbest* particles in the child swarms once per iteration. It is perhaps logical to conclude that there is a particularly variable characteristic to the data space for these two days, such as seeing by a sudden change in weather. This would imply that the influence of the master swarm is to 'distract' the improvement in the respective child swarm *gbest*. Thus preventing the child swarm *gbest* particle from reducing to the smaller scale required to converge on the optimum for that day's data. In the early stages, convergence is slowed due to the perturbation of the *gbest* trajectory within the child swarm. Other particles within the swarm still converge on the *gbest* position: we consider this an advantage of the hierarchical technique, as it reduces overfitting and increases the predictive power of the model. A plot of the data for 10th – 13th July 2005 is shown in Figure 17. The inputs 'in2' and 'in3' are the melt for the glaciers Noir and Blanc respectively, and inputs 'in4' and 'in5' are the sedimentary load for glaciers Noir and Blanc respectively. The 12th July 2005 predicts poorly using single swarm; according to Stott & Mount (2007), a rainfall event occurred on 11th July 2005. However, in a hierarchical swarm model, there is a significant increase in predictive success.

Table 8 only shows a single result for each day, the result shown for 12th July 2005 is one of the higher R-squared values produced over a series of runs. Given the increase in predictability, as shown by the R-squared statistic, it is reasonable to conclude that the data collected from the other 24-hour periods is influencing the convergence, since the effect of a rainfall event will have a time lag on the glacier flow. Extending this logic, the days which score least well in comparison with their single swarm equivalents, are 10th and 13th July 2005. These are characterised by a decreasing temperature, presumably before and after the weather system passes.

Another interesting observation that can be made from the graphs in Figure 17 is that there seems to be a correlation between the air temperature value and the value of ‘in4’. This input is identified in Stott & Mount (2007) and Mount & Stott (2008) as the suspended sediment load at the Glacier Noir measuring station. This measuring station is at a lower elevation than Glacier Blanc and we conclude a higher sediment load is being registered, due to the greater volume of melt and associated proglacial discharge following an increase in temperature. Nevertheless, the correlation is quite strong between this measure and the air temperature. We conclude that the hierarchical results show some improvement over that of the single swarm based on a single dataset. It is too early to determine whether or not the improvement holds for other applications or datasets.

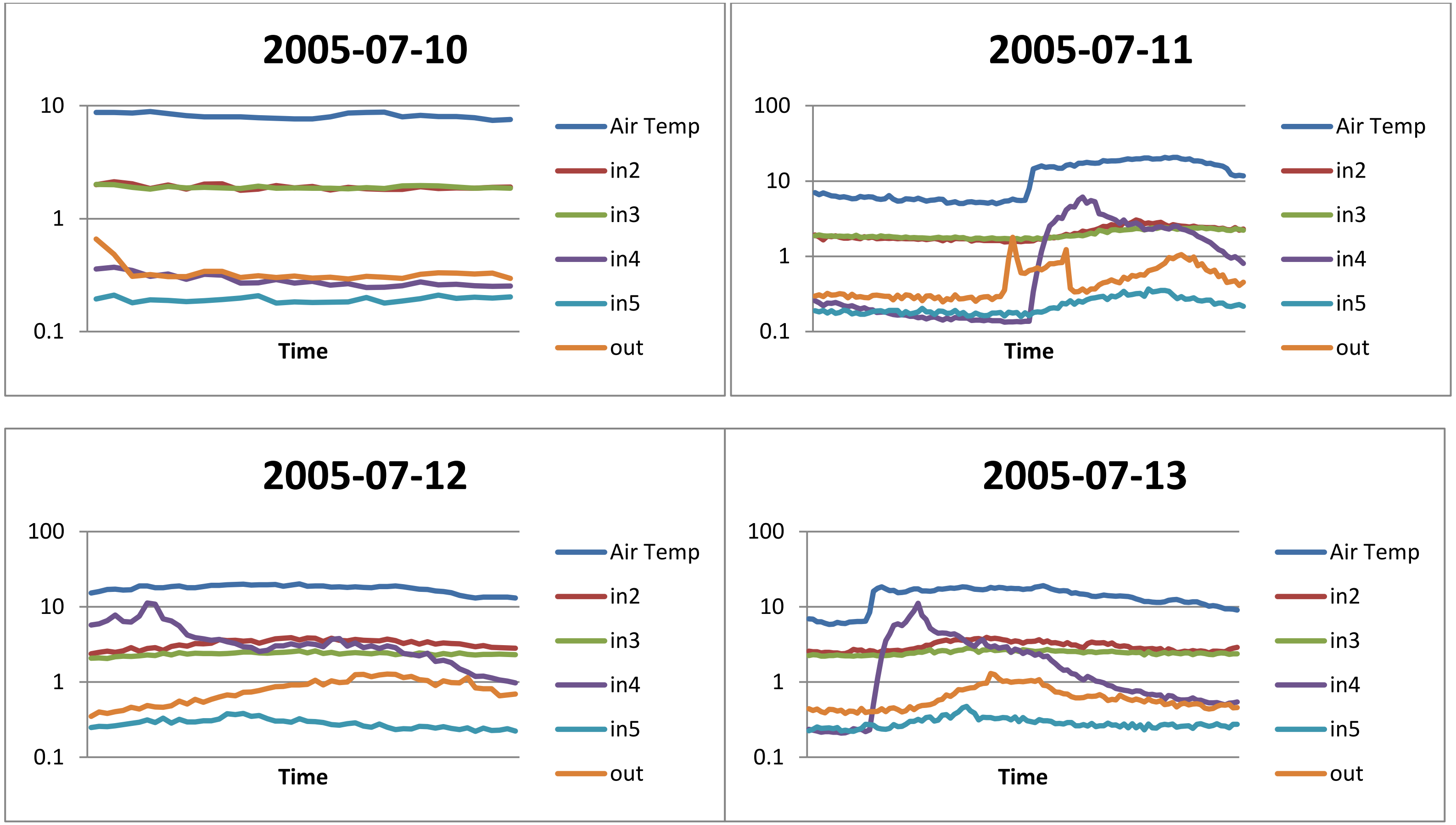


Figure 17: Data plot for 10th – 13th July 2005. The values for 'in2' – 'in5' are the melt and Suspended Sediment measurements upstream of the basin at 'out', see Stott & Mount Stott & Mount (2007).

Stott & Mount (2007, 2008) refer to the use of random variables: we take this to be a reference to modelling the chaotic nature of glacier behaviour. Mount & Stott model suspended sediment concentration using a Bayes Network which uses random values to infer probabilities from which the predicted SSC is derived. It is important that we recognise that randomness and chaos are not the same thing, chaotic systems can be deterministic. We propose that glacier behaviour is chaotic and not random therefore it is our view that a model should endeavour to reflect its chaotic characteristics. However, the authors suggest, especially in Mount & Stott (2008), that a way to model these chaotic processes is to introduce additional random variables into the chosen model. We suggest that this is not a complete solution to modelling unknown data, since although the glacial system is chaotic there are other unmeasured processes, which are not random, contributing to the glacier's behaviour. Introducing more random variables has the effect of adding more degrees of freedom with which to find the factors which map the inputs to the expected outputs. An alternative would be to use known data, for example the statistical *residual* values from Multiple Linear Regression, to provide information about the unknown factors or factors which are not recorded. Here we are using the term *residual* to refer to the error difference between the expected value and the predicted value produced by the model. As the values in the candidate model converge, producing improved predictions, there remains a pattern in the residuals which we suggest is indicative of the unmeasured data. The current dataset for example, includes a direct measure of rainfall rather than the effect of rainfall measured as part of the melt. The hierarchical PSO model demonstrates the possible strength of this idea through the way it finds the degrees of freedom represented in matrix A. The process is analogous to back propagation in ANNs Haykin (1998) where the differences between the predicted and actual values, the residuals, are propagated back through the ANN. Our suggested technique has the benefit that the residuals are subsequently used as an input introducing another input which is responsive to changes in the system. If the mean residual value is used for subsequent iterations, rapid fluctuations in value of the residual would be reduced and, at least in part, the error would approximate the pattern of the unmeasured variables.

Gas Spectrometry

While a hierarchical PSO can successfully predict glacial outflow, we need to apply the hierarchical PSO to other data sets. The identification of a gas within a mix of gases is an interesting scientific problem, particularly in the field of health and safety (Brown et al., 2014; Haitao Li et al., 2014). We used the dataset available from https://archive.ics.uci.edu/ml/datasets/Gas+sensor+array+under+dynamic+gas+mixtures, further details of this dataset are available in Fonollosaa et al. (2015). The dataset is a time series dataset collected from 16 chemical sensors exposed to gas mixtures at various concentrations, in particular two gases, ethylene and methane in air and ethylene and CO in air. The aim is to predict the identities and concentrations the gases.

This is a suitable data set for testing the hierarchal PSO approach as each time series dataset can be used by a separate swarm with the parent swarm used to coordinate them and find a globally optimal solution.

6.1.5 Method and Results

In these experiments, we use the same technique as we did with the glacier data that of the hierarchical swarm technique. As before, the data was also split into test and training sets. The data set is extremely large, with about 8.2 million rows of data, so we used a random subset of

1500 rows for each category for both training and testing sets. We chose this size pragmatically, as it was the largest data set that could reliably fit in memory in the PSO test bed.

The data classification was categorical (gas name and concentration), but the hyperspatial transfer function produced by the PSO give real-valued results. Therefore, we encoded the categories by using the ASCII codes of the sample names. This may appear slightly unusual, however, it is an effective way to produce a numeric value which it does not assume any inherent order to the gas concentrations.

We used seven child swarms, one for each different combination of gases and concentrations in the dataset (including one for no test gas being present in the sample). Each child swarm optimised a transfer matrix from the input data to an indicator value. A presented data set was classified by comparing the output from each swarm's transfer matrix to that swarm's target value, with the closest match indicating the assigned classification.

For comparison, we trained a standard multiple linear regression predictor on the data, taking the output classification as being the class whose encoded value was closest to the MLR output.

This was a challenging data set, as the differences between the sample classes were small. The classifiers could detect differences in gas composition, but neither was reliably able to differentiate between concentrations of the same gas. Therefore, we used just sample gas composition when evaluating the results.

As the classification is categorical, evaluation of the results is simple: we counted the data rows where the predictor made the correct prediction for gas composition, regardless of the gas concentrations. The hierarchical swarm correctly identified 65% of the samples, while the MLR predictor correctly identified 70% of the samples.

While this result is disappointing, the results show that the hierarchical swarm can optimise multiple datasets and perform approximately as well as conventional statistical analysis.

Conclusion

This chapter has considered the implementation of a dynamic hierarchical particle swarm with the intention of providing a technique to process multiple datasets, while preserving the separate identity of each dataset. The underlying mapping model has been successfully applied using a single swarm, to River Severn hydrographical flow prediction in Chapter 5. We extended it to a multi-swarm model and applied it to a statistically less predictable dataset. We have shown that such a model produces an extension of PSO which is able to find an improved solution over the single swarm. In our example a transformation matrix is found, without the requirement to implement a second technique, for example in the paper by Imrie et al. (2000) PSO is used in conjunction with an ANN to make hydrographical predictions. The hierarchical system provides an effective aggregation of the discrete results from the lower level swarms. Our method has an advantage over other techniques that use aggregation, such as ANNs, in that each swarm contributes to the convergent potential, which is the momentum and guided direction of travel to find improved solutions. The aggregation used in our technique does not impede the potential for a single swarm to converge, and information is not lost for use in later stages of the convergence process. Further, by using separate swarms to process each dataset and then aggregating the results in a master swarm of the *gbests* from each child swarm, an improvement is seen over the results generated from each swarm on its own.

However the hierarchal swarm results on the glacier data are still less accurate than those produce by Multiple Linear Regression. In section 6.1.3 and section 6.1.5, we observed that the R squared results for multiple linear regression were also relatively poor and attributed this to the dataset, not including variables which contribute to the amount of SSC measured within glacier melt. A possible solution to this is to optimise a set of data transformations, see section 0 on further work. From the point of view the hierarchical particle swarm, we have to conclude that although our experiments, on both the glacier and gas spectrometry datasets, demonstrate the benefits of our technique in improving the performance of PSO to problems, where PSO is well-suited. We have also shown that there continue to be limitations in PSOs ability to find the optimum on complex datasets. Understanding why an improvement is seen, but without achieving optimum is a possible area of future research as achieving an understanding would be useful for developing complex predictive models.

Chapter 7

Network Theory Inspired Neighbourhoods for PSO

In the previous two chapters, we have demonstrated techniques in applying PSO to datasets, with complex interactions between the inputs and demonstrated a stable improved prediction using the techniques we developed. Using inspiration from the hierarchical swarm model we realised that it was possible that the structure of the neighbourhood was contributing to convergence on local optima. In this chapter we test our new neighbourhood model, as described in section 0. In this model, neighbours are determined by their influence on a particle's performance.

The dynamic neighbourhood PSO used in this chapter shows a large improvement on the results produced from the fully informed particle swarm neighbourhood and an improvement over the *lbest* neighbourhood.

In order to demonstrate the dynamic neighbourhood, given our success with a hydrographical data and the desire to test more complex datasets, we turned to medical and cosmological data for these experiments. These data sets are larger and more complex than the data sets described earlier in the dissertation. The use of these data sets also illustrates the applicability of PSO to other domains enables the model to be developed independently from the hydrological and glacier data sets.

All experiments in this chapter use the same experimental setup and parameters as before, as described in section 0.

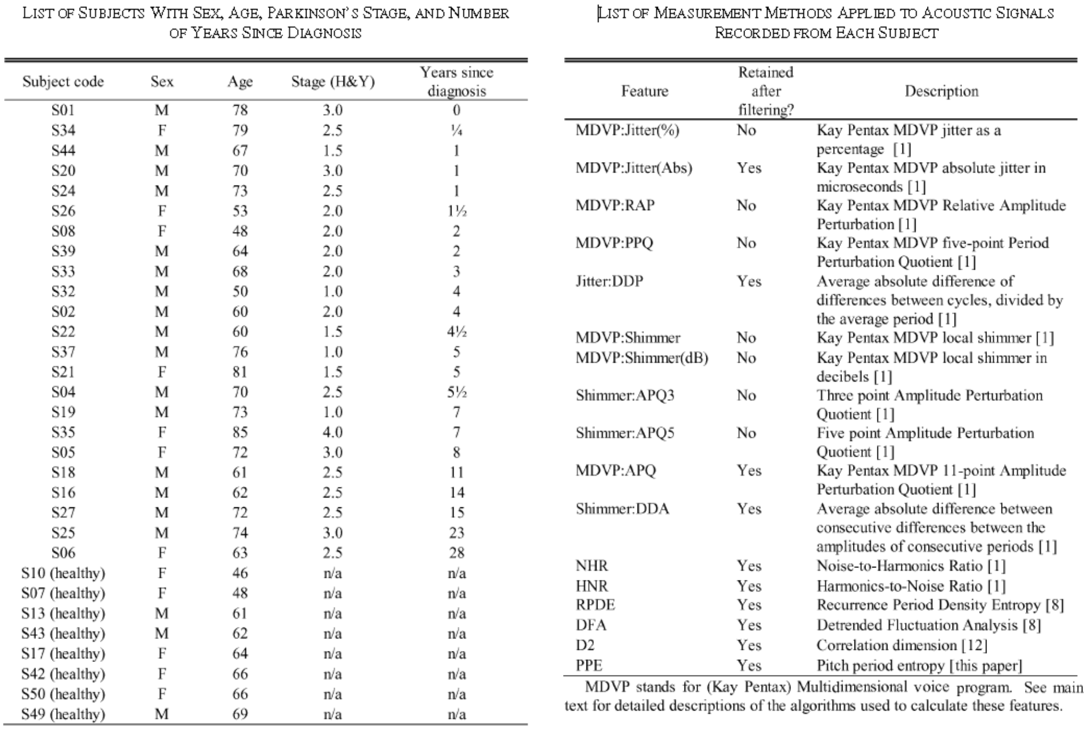
Parkinson's Speech Data

Our experiments using the hydrographical data in the previous two chapters showed that the development from a single swarm to a hierarchical swarm improved the accuracy of the predictions made by our model. We concluded that due to the relatively low R squared values from multiple linear regression on the glacier data was of poor quality. By poor quality we are referring to the way and types of data collected. With this in mind we sought a new dataset. Our rationale in selecting the data set was to apply the dynamic neighbourhood to a completely different dataset with more complex interactions between the inputs and with a known data quality from previous results (2009). Our reasoning behind this was that a complex dataset would benefit from a more adaptive neighbourhood structure. With this in mind we turned to medical data, which is notoriously difficult to optimise because of its time series nature and complex interactions between the inputs. To test our neighbourhood model we used the Parkinson's speech data available from the machine learning repository at http://archive.ics.uci.edu/ml/datasets/Parkinsons (Retrieved: 27 July 2012). This data was originally published in Little et al. (2007) and contains speech data from patients with Parkinson's disease and from a control group of subjects without the disease. The Parkinson data was selected in particular because voice recognition represents a complex dataset and the ability to differentiate between a patient with Parkinson's and an individual without Parkinson's presents a computationally difficult task to develop a model which allows the progression of Parkinson's to be monitored through the use of voice patterns. The Parkinson's datasets also is a high quality dataset, the data was collected in a clinical environment, with the advantage of having published results to use as comparison.

For many people with Parkinson's disease, it is necessary to make several trips to hospital in order to assess the progression of the disease. In recent years there have been attempts to produce a predictive model to assess the severity of the patients' symptoms based on their speech patterns (Little et al., 2007). Successful application of the technique would allow for telemonitoring of the patient using Internet-based telecommunications, therefore reducing the number of hospital visits needed.

The Parkinson data was selected in particular because of the complexity of the data in the data set. Developing a model to differentiate between a patient with Parkinson's and an individual without Parkinson's is a computationally difficult task. There have been several models for the detection of dysphonia in Parkinson's disease. Many of the existing techniques using conventional analytical tools were unreliable in distinguishing between Parkinson's patients and healthy controls (Little et al., 2009). We suggest that one of the reasons for this, based on our own experience in using speech recognition software, is that presentation of dysphonic characteristics in a given patient is highly variable throughout the day, and therefore the speech presentation is inconsistent.

The approach we have adopted with our transfer function, see the equation below, is similar to that of Little et al. (2009). However, the purpose of this chapter is to demonstrate the value of a dynamic neighbourhood in comparison to FIPS. Our experiments use the standard 10 inputs used by Little et al and we did not attempt to find the optimal set of characteristics for our method. Further improvements to our results may be achievable. The data used contains the standard voice measures of *pitch* of vocal oscillation, absolute sound pressure level (indicating the relative loudness of speech), *jitter* (the extent of variation in speech from vocal cycle to vocal cycle), *shimmer* (the extent of variation in speech amplitude from cycle to cycle) and *noise-to-harmonics ratios* (the amplitude of noise relative to tonal components in the speech). Little, et al. also added new statistical measures referred to as *pitch period entropy*; we include this measurement in our experiments. Table 10, reproduced here, shows details of the data used in our experiments.



LIST OF SUBJECTS WITH SEX, AGE, PARKINSON'S STAGE, AND NUMBER OF YEARS SINCE DIAGNOSIS

Subject code	Sex	Age	Stage (H&Y)	Years since diagnosis
S01	M	78	3.0	0
S34	F	79	2.5	¼
S44	M	67	1.5	1
S20	M	70	3.0	1
S24	M	73	2.5	1
S26	F	53	2.0	1½
S08	F	48	2.0	2
S39	M	64	2.0	2
S33	M	68	2.0	3
S32	M	50	1.0	4
S02	M	60	2.0	4
S22	M	60	1.5	4½
S37	M	76	1.0	5
S21	F	81	1.5	5
S04	M	70	2.5	5½
S19	M	73	1.0	7
S35	F	85	4.0	7
S05	F	72	3.0	8
S18	M	61	2.5	11
S16	M	62	2.5	14
S27	M	72	2.5	15
S25	M	74	3.0	23
S06	F	63	2.5	28
S10 (healthy)	F	46	n/a	n/a
S07 (healthy)	F	48	n/a	n/a
S13 (healthy)	M	61	n/a	n/a
S43 (healthy)	M	62	n/a	n/a
S17 (healthy)	F	64	n/a	n/a
S42 (healthy)	F	66	n/a	n/a
S50 (healthy)	F	66	n/a	n/a
S49 (healthy)	M	69	n/a	n/a

LIST OF MEASUREMENT METHODS APPLIED TO ACOUSTIC SIGNALS RECORDED FROM EACH SUBJECT

Feature	Retained after filtering?	Description
MDVP:Jitter(%)	No	Kay Pentax MDVP jitter as a percentage [1]
MDVP:Jitter(Abs)	Yes	Kay Pentax MDVP absolute jitter in microseconds [1]
MDVP:RAP	No	Kay Pentax MDVP Relative Amplitude Perturbation [1]
MDVP:PPQ	No	Kay Pentax MDVP five-point Period Perturbation Quotient [1]
Jitter:DDP	Yes	Average absolute difference of differences between cycles, divided by the average period [1]
MDVP:Shimmer	No	Kay Pentax MDVP local shimmer [1]
MDVP:Shimmer(dB)	No	Kay Pentax MDVP local shimmer in decibels [1]
Shimmer:APQ3	No	Three point Amplitude Perturbation Quotient [1]
Shimmer:APQ5	No	Five point Amplitude Perturbation Quotient [1]
MDVP:APQ	Yes	Kay Pentax MDVP 11-point Amplitude Perturbation Quotient [1]
Shimmer:DDA	Yes	Average absolute difference between consecutive differences between the amplitudes of consecutive periods [1]
NHR	Yes	Noise-to-Harmonics Ratio [1]
HNR	Yes	Harmonics-to-Noise Ratio [1]
RPDE	Yes	Recurrence Period Density Entropy [8]
DFA	Yes	Detrended Fluctuation Analysis [8]
D2	Yes	Correlation dimension [12]
PPE	Yes	Pitch period entropy [this paper]

MDVP stands for (Kay Pentax) Multidimensional voice program. See main text for detailed descriptions of the algorithms used to calculate these features.

Table 10: Reproduced from Little et al. (2009); lists the subjects and measurements.

Little et al. performed some pre-processing to identify the attribute subsets which are more effective at predicting which group the subject is in a Parkinson's patient or control subject without Parkinson's. Their findings, shown in Table 11, show that the derived characteristics which they introduce are better at predicting which group the subject is in than the raw measured data.

7.1.1 Experiments

The description of dynamic neighbourhood swarms can be found in section 0. The experimental parameters and methods are described in section 4.9.

As stated above, we used all 10 variables identified as 'yes' in Table 10 as the input vector to the model. In order to compare the dynamic neighbourhood, we also ran experiments using the FIPS neighbourhood. The results for each neighbourhood are given in Table 15: below.

Due to the complexity of the interactions between the input variables, we modified the scoring function used for the hydrological data – the modified function is shown in the equation below. The rational for the change is to reduce the relative influence of the dimensions with a larger value range to the same scale as base dimensions with a smaller range values. To achieve this, the equation below takes the square root of the value of the first element in the vector and then alternately the cosine and sine of the vector values, **Epps & Ambikairajah (2005)**. As with the hydrological flow data presented in Chapter 5, we use a matrix of weights multiplied by the vector representing the speech characteristics, providing a hyperspatial solution in the form of the matrix, A. The aim in this experiment is for the swarm to find A and B to minimise the error, e, in the third equation below:

$$e = \sum_{1}^{n} \left| expected - r \right|$$

$$s = A \cdot f + B$$

$$r = \sum_{i=1}^{n} \begin{cases} \sqrt{|s_i|} & \text{if } i = 0 \\ \cos(s_i) & \text{if } i \bmod 2 = 1 \\ \sin(s_i) & \text{if } i \bmod 2 = 0 \end{cases}$$

$$e = \sum_{1}^{n} \left| expected\ - r \right|$$

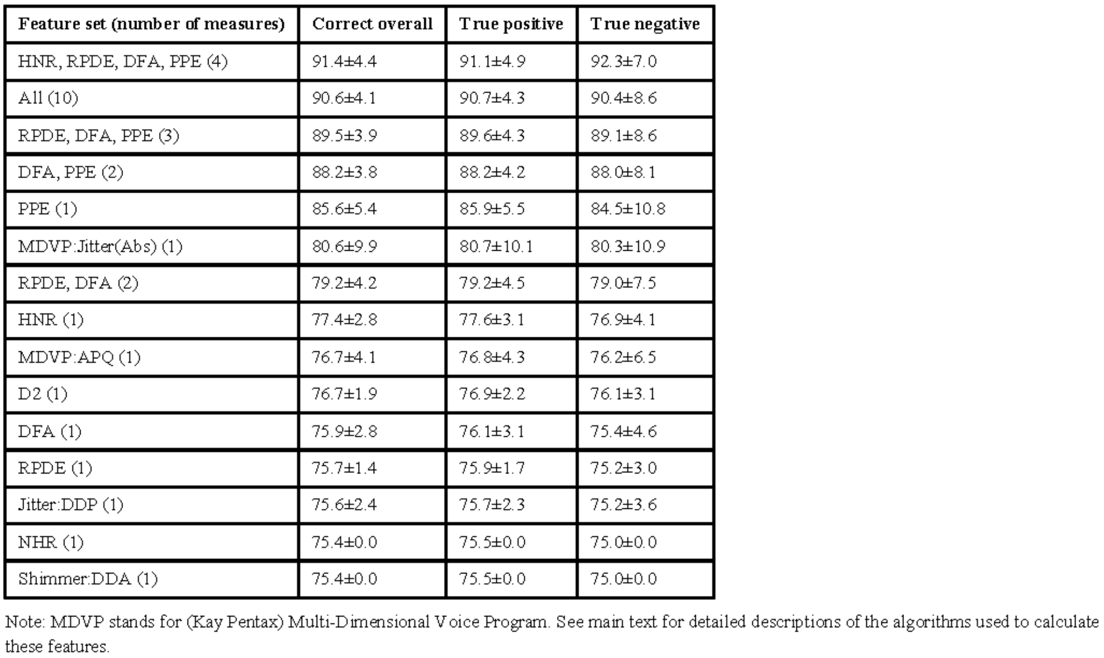
A is a matrix, B is a vector; PSO is used to find the component values of A and B, f is a vector representing the real numbered voice characteristic input values, s is the result vector from the matrix multiplication. The vector B is used to allow for the convergence on a translation element to the model. r is the sum of the result of the conditionally determined function for each of the components in the vector and is the value which predicts if the subject is a Parkinson's patient or not. The function used to aggregate the result vector s was modified to make the result more responsive to changes in the vector, suggested by Epps & Ambikairajah (2005). The effect of this technique is to reduce the relative influences of each dimension to the same scale, and therefore avoid a dimension with a larger range having a bigger effect for a relatively small change in value. As with the hydrographical data, by using a matrix of real numbers we are better able to model the interactions between the inputs through retaining more degrees of freedom, thus introducing greater scope to find a combination of weights which in turn improve the accuracy of the predictions. The third equation aggregates the errors between the predicted and expected values to give an overall score for the particle.

As with the other experiments in this thesis to guarantee that the swarm converges, we used the parameter settings for the swarm given in section 4.9.

7.1.2 Results and Discussion

Our experimental aim is to use the data to explore the effectiveness of a dynamic neighbourhood by comparing it to the performance of the FIPS neighbourhood on the same dataset. Accordingly, we used the same filtered subset of 10 inputs identified in Table 10 used in

Little et al. (2009). The voice patterns in the data are complex and have multiple interactions between the measured variables, making the characteristics of the particles' convergence erratic. This makes the data suitable for demonstrating the relative effectiveness of the different neighbourhood topologies.



Feature set (number of measures)	Correct overall	True positive	True negative
HNR, RPDE, DFA, PPE (4)	91.4±4.4	91.1±4.9	92.3±7.0
All (10)	90.6±4.1	90.7±4.3	90.4±8.6
RPDE, DFA, PPE (3)	89.5±3.9	89.6±4.3	89.1±8.6
DFA, PPE (2)	88.2±3.8	88.2±4.2	88.0±8.1
PPE (1)	85.6±5.4	85.9±5.5	84.5±10.8
MDVP:Jitter(Abs) (1)	80.6±9.9	80.7±10.1	80.3±10.9
RPDE, DFA (2)	79.2±4.2	79.2±4.5	79.0±7.5
HNR (1)	77.4±2.8	77.6±3.1	76.9±4.1
MDVP:APQ (1)	76.7±4.1	76.8±4.3	76.2±6.5
D2 (1)	76.7±1.9	76.9±2.2	76.1±3.1
DFA (1)	75.9±2.8	76.1±3.1	75.4±4.6
RPDE (1)	75.7±1.4	75.9±1.7	75.2±3.0
Jitter:DDP (1)	75.6±2.4	75.7±2.3	75.2±3.6
NHR (1)	75.4±0.0	75.5±0.0	75.0±0.0
Shimmer:DDA (1)	75.4±0.0	75.5±0.0	75.0±0.0

Note: MDVP stands for (Kay Pentax) Multi-Dimensional Voice Program. See main text for detailed descriptions of the algorithms used to calculate these features.

Table 11: List of SVM classification performance results, reproduced from Table III in Little et al. (2009)

The second row of Table 11 contains the results presented by Little et al. (2009) for the combination of features we tested. Each row of the table identifies the group of features tested and the results achieved. It is unclear from Little et al. from where the ±4.1 confidence is derived: one percentage point equates to the misclassification of two subjects. Table 15: shows the results from our experiments for the overall correct, true positives and true negatives. Our initial experiments showed an overall correct for the FIPS neighbourhood of 74.26% and 88.72% overall correct for the dynamic neighbourhood. We then carried out a series of additional experiments to demonstrate the range in accuracy; these results are shown in Table 15:. In contrast to the confidence interval presented for all the results in this table, our results show an average accuracy of 89.90% using the dynamic neighbourhood compared with an average 76.99% for the FIPS neighbourhood with a similar range for both neighbourhoods. The improvement in the accuracy of the dynamic neighbourhood is due to more true positives, although the percentage of true negatives is still relatively low.

	FIPS			**Dynamic**		
	% Correct	**True Positive**	**True Negative**	**% Correct**	**True Positive**	**True Negative**
	85.64%	93.20%	62.50%	91.19%	97.96%	68.75%
	87.69%	94.56%	66.67%	90.26%	94.56%	72.92%
	70.36%	96.60%	16.67%	89.23%	94.56%	68.75%
	74.26%	100.00%	0.00%	88.72%	93.20%	62.50%
	71.97%	100.00%	2.08%	90.26%	96.60%	64.58%
	70.50%	97.96%	27.08%	89.74%	97.28%	68.75%
Mean	76.74%	97.05%	29.17%	89.90%	95.69%	67.71%
Standard Deviation	0.08	0.03	0.29	0.01	0.02	0.04

Table 15: Summary of Results

Taken together, the percentage correct and the percentage of false predictions demonstrate it is possible to predict whether a subject has Parkinson's from the characteristics of their voice pattern, see Table 15:. However our model demonstrates a relatively low true negative value which would need more work in order to be useful in a clinical setting. Prior to the work of Little, et al. (2007), using voice pattern analysis to monitor the clinical progression of Parkinson's Disease had not been considered. The subset of voice parameters used in our experiments, and those of Little et al., includes measurements derived from the voice data. The reason that these derived measurements produce improved accuracy needs to be better understood, in order to reduce the false-positives produced when processing data. Our results indicate that the reason for not attaining 100% accuracy with the dynamic neighbourhood is largely due to false positives:

that is, subjects without Parkinson's were being treated as if they had Parkinson's. Understanding the reason for this will be important if the results are to be used to inform the clinical management of a patient's presentation[10]. The high degree of accuracy seen in these results is interesting, but we suggest that further research is needed to find improved analysis before our technique is used for the telemonitoring of Parkinson's patients. Telemonitoring of Parkinson's patients has been achieved by Little et al. (2009), albeit with a relatively small group of subjects. Recently the Parkinson's Voice Initiative http://www.parkinsonsvoice.org was established with the aim of gathering voice data to develop the authors' algorithm.

Since we completed this experimental work, two additional papers have been published (Tsanas et al., 2012a, 2012b) using an enhanced technique and a larger dataset than used in Little et al. (2009). The published results show an improvement over our results however the dataset or details of the technique have not been published. Therefore a comparison to the technique we use has not been possible.

[10] In a clinical setting a patient's presentation refers to the constellation of symptoms which taken together leads to a diagnosis

Iteration	*Neighbourhood*	*Iteration*	*Neighbourhood*
0	23,11,26,13	199	2,38,9,12,13
1	1,2,33,6,27,28	299	2,38,7,10,26,12,13
2	34,23,6,26,28,14	399	34,2,38,7,12,13
3	18,5,6,7,13	499	0,19,38,7,12,13
4	20,39,11,13,28	599	19,32,3,7,13,14
5	20,39,22,11,13,28	699	19,38,37,7,12,13,14,30
7	20,39,22,11,29,13	799	2,19,38,7,13,14
8	0,20,22,11,29,13	899	34,19,3,38,6,8,12,13,14
9	0,20,22,11,13	999	19,38,9,12,13,14
99	32,2,9,10,13,14,15		

Table 16: A sample neighbourhood of the *gbest* particle over a series of iteration when processing Parkinson's speech data. The second column lists the identities of the particles in the neighbourhood.

An examination of the detail of the dynamic neighbourhoods shows how the swarm organises around the particles that have found the best solutions. Table 16: shows the neighbourhood membership of the *gbest* particle, for the first 10 iterations, then in steps of 100 over the 1000 iterations. The second column lists the identities of the particles in the *gbest's* neighbourhood. It is important to bear in mind that the identity of the *gbest* is not the same from iteration to iteration. In the earlier stages of convergence, the recruitment, see 95, results in an increased probability of a particle recruiting more neighbours and therefore more influences on its trajectory. As the convergence progresses, each particle's neighbourhood becomes a stable set of other particles. For example, particles 13, 14, 19 and 38 become stable members of the *gbest's* neighbourhood. A characteristic suggesting the adoption of key influencers based on these particles' continuing success. This is consistent with the way social networks develop and stabilise around a few key influencers (Watts et al., 2005). We find it intriguing that our neighbourhood model seems to have some of the characteristics of a social network, yet by selecting members based on the value of their contribution, the swarm outperforms one in which the neighbourhood connects all particles to every other particle (FIPS).

The emergence of a stable group of key influencers, and frequent changes in the neighbourhood for particles outside this group, occurred in all the experiments we completed. This suggests the equations used to recruit and remove neighbours are effective in finding the improving particles. A note of caution is needed, however: the key influencers in a particle swarm may be non-improving particles centred on local optima, rather than particles which are continuing to improve to the global optimum. However, our results consistently demonstrate an improvement over the comparable results presented in Table 11 which would suggest that the global optimum is being found.

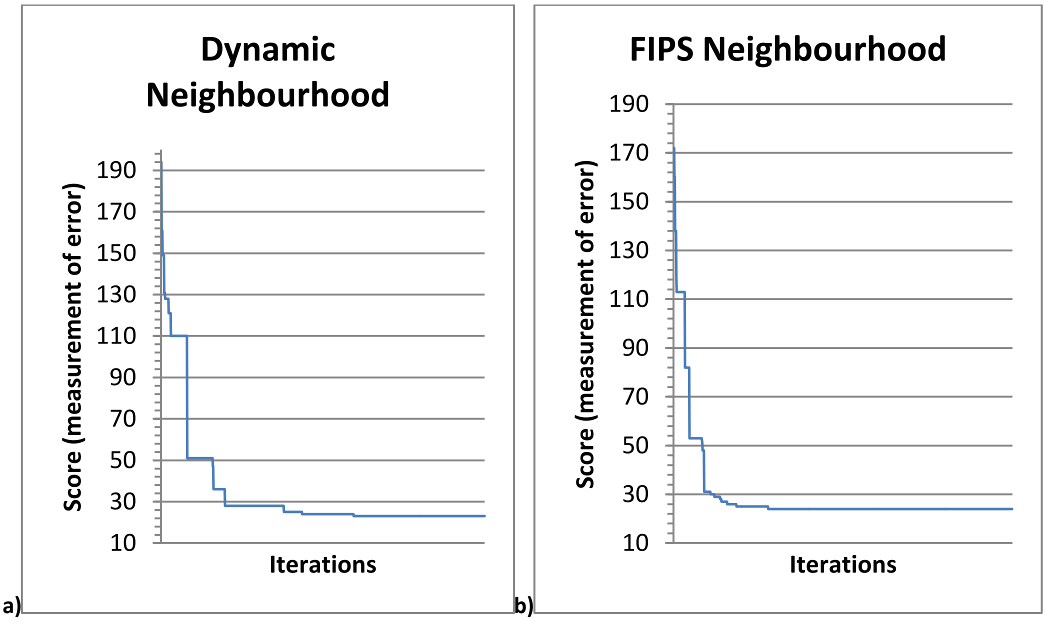


Figure 18: a) Convergent pattern for a Dynamic Neighbourhood b) Convergent pattern for a FIPS Neighbourhood

Figure 18a shows the convergent pattern of the *gbest* particle over the same iterations for the dynamic neighbourhood. Figure 18b, shows the results using the FIPS neighbourhood. The convergence is slower in the dynamic neighbourhood, resulting from the reduction in the number of influences on the particle; this is seen in the more stepped line. The dynamic neighbourhood generates more diverse influences on a particle's velocity, therefore enhancing a particle's capacity for exploration. This is demonstrated by the slightly later convergence to a flat line in Figure 18a, when compared with Figure 18b. The changing influences on a particle are demonstrated through the changing membership of the *gbest*'s neighbourhood.

Lbest Neighbourhood Results

We completed five experiments using the *lbest* neighbourhood. The *lbest* neighbourhood is typically characterised by setting $n = 3$. We were expecting that this neighbourhood topology would perform similarly to the dynamic neighbourhood, anticipating that the reason the dynamic neighbourhood performed better was due to the smaller neighbourhood size. However the *lbest* neighbourhood performed statistically less well than the dynamic neighbourhood, with a mean of 81.35%±1.88% predictions correct. This suggests the dynamic neighbourhood's method of selecting neighbours improves convergence on the global best for the Parkinson's voice data.

Cosmology Data

There are significant amounts of cosmology data available from the 2dF-SDSS LRG and QSO (2SLAQ) website, http://www.2slaq.info/. In order to further demonstrate the usefulness of our dynamic neighbourhood concept we decided, following advice, to predict the redshift using the data set as true labels of each objects' redshift. Using the following datasets combined on the object identifier – Croom et al. (2009) and Collister et al. (2007) – to obtain the redshift value z photo; this was the value we wanted to predict from a subset of measurements taken from distant galaxies. We used the following subset of data values with which to predict z photo value, [20 Petrosian Mag u], [21 Petrosian Mag g], [22 Petrosian Mag r], [23 Petrosian Mag i], [24 Petrosian Mag z]. The meaning of these attributes is defined in Blake et al. (2007). Briefly, the parameters represent the Petrosian Magnitude in each band. For further discussion on how the

Petrosian Magnitude for each band is calculated, see Blake et al. (2007) and Blanton et al. (2001).

The scientific aim is to predict the redshift. This parameter set was selected to see if it was possible to improve the prediction of the redshift value using a smaller dataset then the full set of variables which include derived statistics. Due to the time constraints, we have the outcome therefore is whether or not PSO, using the dynamic neighbourhood, can improve over MLR on this dataset for the R squared statistic. Our aim was to show that the dynamic neighbourhood performed better than the multiple linear regression on the same dataset. A similar model has not been published on this data, which is why we made the comparison multiple linear regression. Machine Learning has previously been used to identify interstellar bubbles; Beaumont et al. (2014) is included here is a sample of the work using machine learning in the field of cosmology.

7.1.3 Method and Results

As with the Parkinson's data, we constructed a model by using the parameters given in section 0 for the dynamic neighbourhood. This is to see if we can obtain good results using a proven convergent set of parameter values.

After a series of five runs, due to available computing time, the average R squared statistic for the dynamic neighbourhood was a value of 0.025, with a range of 0.0152 to 0.029 compared with the multiple linear regression of 0.008. This is clearly an improvement on the multiple linear regression value. But this is not a scientifically useful prediction technique.

In order to validate whether or not it was the dynamic neighbourhood topology led to the poor predictions we ran equivalent experiments, but this time using the FIPS and *lbest* neighbourhoods, the latter with five neighbours included in each neighbourhood. The results were similarly disappointing, with the R squared values of 0.024 for both neighbourhoods. This demonstrates that the dataset is not highly predictable, and that the topology used does not statistically affect the predictions. However, the dynamic neighbourhood PSO gave the best results of all the PSO alternatives, indicating that it is able to extract the most useful information from this small selection of parameters.

Conclusion

We have tested a new dynamic neighbourhood topology inspired by research in network theory. The aim was to explore the potential of particle swarm with the use of the standard velocity equation to model certain characteristics of social network interaction.

The dataset on which we chose to test our neighbourhood contains complex relationships between the inputs. Our results show a significant improvement over a fully connected network, and a smaller but nevertheless significant improvement over the *lbest* neighbourhood, compared with our dynamic neighbourhood on the same data which in turn showed an improvement on the results presented by Little et al. (2009). Our experiments show that the dynamic neighbourhood demonstrates improved solutions over all other neighbourhoods tested. The difference between the dynamic neighbourhood and the other topologically based neighbourhoods is that the selection of neighbours is dynamic and based on a particle's rank. We therefore conclude that the dynamic neighbourhood is able to converge to an improved final solution because the particles' trajectories are being influenced by particles that have solutions which already have the potential to improve. The improvement over the *lbest* neighbourhood suggests that the quality of the information shared, rather than the quantity, is also important in facilitating improved convergent potential. This is a significant research finding in terms of the application of neighbourhood topologies and PSO. Although the classical PSO neighbourhoods lead to information dissemination which is roughly equal throughout the swarm, our model results in the particles receiving a more directed path to convergence. We describe the information dissemination as roughly equal because although some of the classical neighbourhood topologies support local neighbourhoods the whole swarm remains interconnected in a linear structure, resulting in changes to the identity of particles influencing a given particle. We suggest that a network based neighbourhood is also better suited for modelling real-world social networks, and therefore could be used in modelling the spread of epidemics (Watts et al., 2005). Watts et al. recognise that their model is too simplistic; they model a population as a nested hierarchy of subpopulations that interact within local contexts: schools, workplaces etc. However the Watts et al. model assumes the level of interaction within a subpopulation is uniform. We suggest that our dynamic neighbourhood is a suitable model to introduce an element of variability into the interactions within the subpopulation.

We note the results presented in this chapter show a significant variation in the membership of a particle's neighbourhood. We see this as a strength of our model, which leads to greater diversity in the influences on a particle's velocity. Further development of the formulas used to update a neighbourhood may lead to a more flexible social network model for the dissemination of information to solve complex combinatorial problems. Finally, although our results show a significant improvement, only one voice pattern dataset was tested and it should be noted that we did not test on the variety of parameter subsets used by the original authors. Further study is required to verify the results on a wide variety of datasets.

Chapter 8

Conclusion

In this thesis we considered the performance of several enhancements to PSO on five real-world datasets. For each of these datasets our aim was to demonstrate the performance of PSO, with this in mind we used a parameter configuration guaranteed to converge. Using the River Severn dataset we demonstrated the principle of our technique showing an improvement on flood prediction over multiple linear regression. We introduced innovations to the PSO heuristic: hierarchical swarm organisation and dynamic neighbourhoods, and tested these innovations on large, complex, real-world data sets, with varying success. However we conclude that our enhancements improve the performance of PSO on all datasets tested

Summary

We identified the following issues with existing PSO techniques:

- PSO has a tendency to converge prematurely to local optima.
- Many of the real-world applications of PSO required some combination of a problem-specific implementation of the PSO heuristic, pre-processing of the data, or additional techniquest which PSO supplemented
- The original PSO can only optimise a single dataset at a time. This makes it difficult to optimise a group of discrete but related datasets

- The standard neighbourhoods used by PSO are based on a particle's index position, rather than a measure of its contribution. This is important because the index-based neighbourhoods are essentially random, unchanging groupings of particles.

The contributions to knowledge in this thesis are as follows:

- The use of a problem-independent model to find the objective function values for problems which involve prediction or modelling of a system. This was introduce in section 0 and tested in Chapter 5, using hydrographical flow data. The introduction of our model enabled PSO, in its single swarm variant, to find a predictive model with accuracy comparable to multiple linear regression, but with the advantage of being applicable to datasets which had not been not used to train the model.

- We identified four stages in the swarm's convergent behaviour characterised by the dominant influence on the particle's velocity. Through the identification of these stages we found a limitation of the PSO velocity equation that becomes apparent when the swarm reaches the stage where improved solutions are not found some distance from the current *gbest*. Without any increase in velocity at this stage, the velocity of all particles reduces, and with it the swarm's capacity for further exploration. This insight informed our subsequent development of hierarchical PSO.

- The hierarchical form of PSO was described in section 0 and tested in Chapter 6. Each swarm can optimise a subset of data and influence the other swarms' convergence. The significant feature of this version of PSO, which gives it an advantage over competing variants, is that it continues to respect the interactions between particles in both the master and child swarms.

We demonstrated the value of dynamic, hierarchical PSO with glacier outflow data and gas spectrometry data. The results of both sets of experiments showed that the hierarchal swarm was consistently able to outperform a single swarm. However, rather disappointingly, the technique was outperformed by the MLR results, on both glacier and gas spectrometry datasets. We therefore conclude that our hierarchical technique improves PSO's performance on problems that are represented by discrete data sets which need to be processed simultaneously.

- We developed a dynamic neighbourhood topology for PSO in section 0. In this variant, particles select their neighbours based on the contribution made to the particle's success. This variant was tested in Chapter 7. The advantage of such a topology is that the particles' neighbourhoods adapt to the value each particle places on the other particles. When applied to the Parkinson's voice data, the dynamic neighbourhood demonstrated a significant improvement over the FIPS neighbourhood model. Not only did the dynamic neighbourhood converge to a lower optimal value and was able to do so consistently. These results were repeated when the dynamic neighbourhoods were applied to the cosmological data. Although this data set produced much lower quality results than the Parkinson's data, the dynamic neighbourhoods performed the best of the approaches we used. The results presented in Chapter 7 represent a significant contribution to the application of neighbourhood topologies with particle swarm optimisation.

Further Work

The ideas for further research which have presented during our research are briefly summarised below:

Understanding the Benefits of Hierarchical Particle Swarm Optimisation

We saw in Chapter 6 that hierarchical particle swarm optimisation was able to improve on the results produced by a single swarm. Future development of this PSO variant will include testing on different complex datasets, and the development of different objective functions to produce a comparison with which to benchmark the hierarchical variant against other variants of PSO. A key element of this work will be determine the general properties of the data set and fitness landscape that control the hierarchy's ability to identify the global maximum in a complex fitness landscape and with disjoint data sets.

Convergent Behaviour of PSO on Real-World Data

An analysis of the convergence behaviour of PSO using real-world data, so that the relationship between the particle bests and the particle's current position can be further understood, especially where the data is noisy as with most real-world data. In this thesis we have deliberately used a parameter configuration guaranteed for convergence van den Bergh (2002). Further analysis may lead reformulation of the velocity equation. Although Blackwell (2012) summarises the recent research on Bare Bones PSO development of Kennedy (2004) by Krohling (2005), Wang et al. (2008), and Krohling & Mendel (2009), the paper concentrates on test functions. Blackwell points out that the Bare Bones implementation is problem-specific and that classes of problems remain unexplored. The Bare Bones variant of

PSO uses probability distribution sampling as its velocity update rule. An improved understanding of convergence to solutions in real-world problems is needed in order to better understand how convergence is achieved by the inertia weight and constriction factor variants of PSO. This research should also consider the influence of larger distances between particle's positions, especially during the early stages of a run, and the effect of this on the particle's cumulative velocity. Further consideration needs to be given to the relative influences of the existing velocity, the change in velocity and the position of the particles' bests.

Advantage of the Dynamic Neighbourhood

An interesting effect of the dynamic neighbourhood, demonstrated in Chapter 7, is that particles improve to better solutions without necessarily having the *gbest* in their neighbourhood. The neighbourhood memberships have shown that the dynamic neighbourhood performed better without *gbest*. The convergent behaviour of particles without the influence of *gbest* in their neighbourhood needs to be explored, so that the dissemination of information within the swarm can be better exploited. Our experiments establish stable groups which are analogous to the way social networks develop. We believe this to be a significant observation, which could lead to a better understanding of the dissemination of information through a social network. It would be interesting to apply our dynamic neighbourhood to the work done by Watts et al. (2005), to explore the spread of information and disease through social networks. Current research in this area lacks an effective model of an individual's interaction within the group. The dynamic neighbourhood offers a way to model interactions between individuals based on the value they place on each other within their local group.

Function interdependence

The current implementation of PSO uses a single vector for a particle's velocity and a single vector for its position. This means that PSO is unable to optimise a series of inter-dependent functions. One example of this would be an iterated function system Barnsley (2000). An iterated function system (IFS) is a closed set of functions which are iteratively applied according to the probability associated with the function. Each function is a matrix of values and an associated probability. In addition, each function is inter-dependant. PSO could be extended such that the velocity and position vectors are replaced by matrices so that an IFS for a given dataset may be found. Specifically, an extension could be developed to make use of the collage theorem Barnsley (2000), to compare the characteristics in two or more natural phenomena, such as weather systems. The collage theorem states that for any given dataset it is possible to find a set of mathematical transformations which approximate the dataset Barnsley (2006). The theorem describes a distance measure, the Hausdorff distance, a measure of the similarity of two datasets. If a smaller set of transformations can be identified, this may be used as a reference set from which a comparison can be made. There is currently no universal method for finding iterated function systems for a given dataset. It is expected that solving this problem would allow the models we use in Chapter 6 and Chapter 7 to be extended to model more complex interactions. This in turn would mean that a dynamic comparison between historical and current data could be made to perhaps an early warning system or to add a degree of adaption to an existing model.

Adaptive Vmax

Finally, in Chapter 4, we briefly touched on the potential advantage of using *Vmax* to enhance exploitation in the early stages. Although PSO has had some notable real-world successes – for example: Slade et al. (2004), Eberhart & Kennedy (2004) and Lei & Liying (2005), a more adaptive velocity may lead to stronger exploitation potential and therefore slower convergence, which in turn may provide better convergence as there would be an improved probability of finding better solutions. We suggest that a possible enhancement is to allow *Vmax* to adapt based on the clustering coefficient of the swarm and the elapsed iterations. Such a methodology could be explored to allow the swarm to be more responsive to the solution space as improved solutions are found.

This thesis has considered the application of swarm computation, specifically, particle swarm optimisation to several real-world computational optimisation problems. We have shown that PSO is able to improve on other optimisation techniques when suitable abstractions are made from the problem domain.

Glossary

Ant Colony Optimisation (ACO): a population based, socially inspired, optimisation technique. The technique uses the generation of an artificial pheromone trail to influence the discovery of improved solutions.

Candidate Solution: a member of a population of possible solutions that satisfy all constraints.

Convergence: the point at which the candidate solutions, in a population of solutions, stop making further significant improvement.

Degrees of Freedom: the number of values in the final calculation of a statistic that are free to vary.

HydroTest (www.hydrotest.org.uk) is a free statistical analysis site for comparing observed (i.e., expected) and modelled (i.e. forecast or predicted) time series data produced by hydrological systems. This was used to validate the hydrological results.

Mean Squared Error (MSE): a measure of the average error between the actual and predicted values produced by a particular mathematical model.

Objective Function: the function to be optimised in an optimisation problem.

Particle Swarm Optimisation (PSO): a population based, socially inspired, optimisation technique. The technique uses the positions of the particles in the population to influence the discovery of improved solutions.

R squared: a statistical measure of how well predicted data approximates real data points. R squared is a descriptive measure between zero and one, indicating how good one term is at predicting another, a value nearer one indicates a better fit. R squared needs to be applied carefully since it can produce erroneous readings, for example, it cannot be used for scoring a model directly rather a measure of the goodness of fit between the predicted and real datasets. R squared is a standard measure used for hydrological data.

Residual: an observable estimate of an unobservable error. In this thesis it is equivalent to the squared statistical error.

Bibliography

Abdelbar, A.M. & Abdelshahid, S., 2004. Instinct-based PSO With Local Search Applied To Satisfiability. *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, 3, pp.2291- 2295.

Abraham, A., Das, S. & Roy, S., 2008. *Swarm Intelligence Algorithms for Data Clustering. In: Soft Computing for Knowledge Discovery and Data Mining.* Springer US.

Akat, S.B. & Gazi, V., 2008. Particle Swarm Optimization With Dynamic Neighborhood Topology: Three Neighborhood Strategies And Preliminary Results. *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE , vol., no., pp.1-8, 21-23 Sept. 2008*, pp.1-8.

Angeline, P.J., 1998a. Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy And Performance Differences. *Evolutionary Programming Vii, 601-610. Berlin: Springer*, pp.601-10.

Angeline, P.J., 1998b. Using Selection To Improve Particle Swarm Optimization. *Proceedings Of The IEEE Congress On Evolutionary Computation (CEC 1998), Anchorage, Alaska, USA.*, pp.84-89.

Antoniou, P. et al., 2013. Congestion control in wireless sensor networks based on bird flocking behavior. *Computer Networks*, 57(5), pp.1167-91.

Ashabani, M. & Mohamed, Y.-R.I., 2011. Multiobjective Shape Optimization of Segmented Pole Permanent-Magnet Synchronous Machines With Improved Torque Characteristics. *Magnetics, IEEE Transactions on*, 47(4), pp.795-804.

Bala Krishna, M. & Doja, M.N., 2011. Swarm intelligence-based topology maintenance protocol for wireless sensor networks. *Wireless Sensor Systems, IET*, 1(4), pp.181-90.

Barnsley, M.F., 2000. *Fractals Everywhere*. Morgan Kaufmann Publishers Inc,US.

Barnsley, M.F., 2006. *SuperFractals*. 1st ed. Cambridge University Press; 1 edition (7 Sep 2006).

Baskar, S. & Suganthan, P.N., 2004. A Novel Concurrent Particle Swarm Optimization. *Evolutionary Computation, 2004. CEC2004. Congress on Volume 1, 19-23 June 2004*, pp.792 – 796.

Beaumont, C.N. et al., 2014. The Milky Way Project: Leveraging Citizen Science and Machine Learning to Detect Interstellar Bubbles. *Astrophysical Journal Supplement*.

Bhattacharya, R. & Bhattacharyya, T., 2012. Position Mutated Hierarchical Particle Swarm Optimization and its Application in Synthesis of Unequally Spaced Antenna Arrays. *Antennas and Propagation, IEEE Transactions on* , 60(7), pp.3174-81.

Blackwell, T., 2012. A Study of Collapse in Bare Bones Particle Swarm Optimization. *Evolutionary Computation, IEEE Transactions on* , 16(3), pp.354-72.

Blackwell, T.M. & Branke, J., 2004. Multi-Swarms, Exclusion and Anti-Convergence in Dynamic Environments. *IEEE Transactions on Evolutionary Computation*, pp.459-72.

Blackwell, T. & Branke, J., 2006. Multiswarms, Exclusion, and Anti-Convergence. *IEEE Transactions on Evolutionary Computation*, 10(4), pp.459-72.

Blake, C., Collister, A., Bridle, S. & Lahav, O., 2007. Cosmological baryonic and matter densities from 600,000 SDSS Luminous Red Galaxies with photometric redshifts. *MNRAS*, pp.1527--1548.

Blanton, M.R. et al., 2001. The Luminosity Function of Galaxies in SDSS Commissioning Data. *The Astronomical Journal*, 121(5), pp.2358-80.

Blum, C. & Dorigo, M., 2005. Search Bias In Ant Colony Optimization: On The Role Of Competition- balanced Systems. *Evolutionary Computation, IEEE Transactions On volume 9, Issue 2, April 2005*, 9(2), pp.159-74.

Bo Liu et al., 2006. Directing Orbits Of Chaotic Systems By Particle Swarm Optimization. *Chaos, Solitons & Fractals*, 29(2), pp.454-61.

Bonabeau, E., 1999. Editor's Introduction: Stigmergy. *Special issue of Artificial Life on Stigmergy*, 5(2), pp.95-96.

Bonabeau, E. & Théraulaz, G., 2000. Swarm Smarts. *Scientific American*, pp.72-79.

Brown, V.M., Crump, D.R., Plant, N.T. & Pengelly, I., 2014. Evaluation of the stability of a mixture of volatile organic compounds on sorbents for the determination of emissions from indoor materials and products using thermal desorption/gas chromatography/mass spectrometry. *Journal of Chromatography A*, 1350, pp.1-9.

Carlisle, A. & Dozier, G., 2000. Adapting Particle Swarm Optimization To Dynamic Environments. *Proceedings, 2000 ICAI, Las Vegas, NV, Vol. I, pp.429-434*, I, pp.429-34.

Carlisle, A. & Dozier, G., 2001. Tracking Changing Extrema With Particle Swarm Optimizer. *Technical Report CSSE01-08. 2001. Auburn University.*.

Changhe Li & Shengxiang Yang, 2012. A General Framework of Multipopulation Methods With Clustering in Undetectable Dynamic Environments. *Evolutionary Computation, IEEE Transactions on*, 16(4), pp.556-77.

Changhe Li, Shengxiang Yang & Trung Thanh Nguyen, 2012. A Self-Learning Particle Swarm Optimizer for Global Optimization Problems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* , 42(3), pp.627-46.

Chau, K., 2007. A Split-Step Particle Swarm Optimization Algorithm In River Stage Forecasting. *Journal of Hydrology (2007) 346*, 346(3-4), pp.131-35.

Chen, S., 2009. Locust Swarms – A New Multi-Optima Search Technique. *in Proceedings of the IEEE Congress on Evolutionary Computation*, pp.1745–52.

Cheng-Chi Wu, Wen-Li Lee, Yung-Chang Chen & Kai-Sheng Hsieh, 2013. Evolution-Based Hierarchical Feature Fusion for Ultrasonic Liver Tissue Characterization. *Biomedical and Health Informatics, IEEE Journal of*, 17(5), p.967976.

Chen, S. & Montgomery, J., 2011. Selection Strategies for Initial Positions and Initial Velocities in Multi–optima Particle Swarms. *in Proceedings of the Genetic and Evolutionary Computation Conference*, pp.53–60.

Chen, A.B., Zhang, P. & Yokota, H., 2013. Evaluating treatment of osteoporosis using particle swarm on a bone remodelling mathematical model. *Systems Biology, IET* , 7(6), pp.231-42.

Clearwater, S., Hogg, T. & Huberman, B.A., 1992. Cooperative Problem Solving. *In Computation: The Micro and Macro View, Singapore: World Scientific*, pp.33-70.

Clerc, M., 2006. *Particle Swarm Optimization*. ISTE.

Clerc, M. & Kennedy, J., 2002. The Particle Swarm - Explosion, Stability, And Convergence In A Multidimensional Complex Space. *Evolutionary Computation, IEEE Transactions On*, 6(1), pp.58-73.

Coello Coello, C.A. & Maestria, M.S., 2002. MOPSO : A Proposal For Multiple Objective Particle Swarm Optimization. *Proceedings Of The IEEE Congress On Evolutionary Computation (CEC 2002), Honolulu, Hawaii USA. 2002*.

Coello Coello, C.A., Pulido, G.T. & Lechuga, M.S., 2004. Handling Multiple Objectives With Particle Swarm Optimization. *Evolutionary Computation, IEEE Transactions On, Volume 8, Issue 3, June 2004*, 8(3), pp.256 - 279.

Colby, B.R., Hembree, C.H. & Jochens, E.R., 1953. Chemical quality of water and sedimentation in theMoreau River Drainage Basin, South Dakota. *U.S. Geological Survey Circular 270, 53*.

Collister, A. et al., 2007. MegaZ-LRG: A photometric redshift catalogue of one million SDSS Luminous Red Galaxies. 375, pp.68-76. Available at: http://www.2slaq.info/readme_megazlrg.

Corne, D., Dorigo, M. & Glover, F., 1999. *New Ideas In Optimisation (Advanced Topics In Computer Science)*. Mcgraw-Hill.

Croom, S.M. et al., 2009. The 2dF-SDSS LRG and QSO Survey: The spectroscopic QSO catalogue. 392, pp.19-44. Available at: http://www.2slaq.info/query/2slaq_LRG_webcat_hdr.txt.

Cui-Ru Wang et al., 2005. A Modified Particle Swarm Optimization Algorithm And Its Application In Optimal Power Flow Problem. *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on , 18-21 Aug. 2005,* 5, pp.2885- 2889.

de Castro, L.N. & Von Zuben, F.J., 2004. *Recent Developments in Biologically Inspired Computing*. Idea Group Publishing.

de Oca, M.A.M., Stutzle, T., Birattari, M. & Dorigo, M., 2009. Frankenstein's PSO: A Composite Particle Swarm Optimization Algorithm. *Evolutionary Computation, IEEE Transactions on*, 13(5), pp.1120-32.

Delgado Saa, J.F. & Cetin, M., 2013. Discriminative Methods for Classification of Asynchronous Imaginary Motor Tasks From EEG Data. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* , 21(5), pp.716-24.

Domniţa, M., Crăciun, A.I. & Haidu, I., 2009. GIS In Determination Of The Discharge Hydrograph Generated By Surface Runoff For Small Basins. *Geographia Technica.*, 8(1), pp.11-22.

Eberhart, R. & Kennedy, J., 2004. Human Temour Analysis With Particle Swarm Optimisation. *Proceedings Of The 2003 IEEE Swarm Intelligence Symposium. Sis'03 (cat. No.03ex706)*, pp.1927-30.

Eberhart, R.C. & Shi, Y., 1998. Comparison Between Genetic Algorithms And Particle Swarm Optimization. *Evolutionary Programming VII: Proceedings of the Seventh Annual Conference on Evolutionary Programming, San Diego, CA 1998*.

Eberhart, R.C. & Shi, Y., 2000. Comparing Inertia Weights And Constriction Factors In Particle Swarm Optimization. *Evolutionary Computation, 2000. Proceedings Of The 2000 Congress On volume 1, 16-19 July 2000*, 1, pp.84-88.

Eberhart, R.C. & Shi, Y., 2001. Particle Swarm Optimization: Developments, Applications And Resources. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2001), Seoul, Korea.*.

Emara, H.M., 2009. Adaptive Clubs-Based Particle Swarm Optimisation. *American Control Conference, 2009. ACC '09. 10-12 June 2009*, pp.5628 – 5634.

Engelbrecht, A.P., Masiye, B.S. & Pampard, G., 2005. Niching Ability Of Basic Particle Swarm Optimization Algorithms. *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE, 8-10 June 2005*, pp.397- 400.

English, T.M., 2000. Optimization Is Easy and Learning Is Hard in the Typical Function. *Proceedings of the 2000 Congress on Evolutionary Computation: CEC00*, pp.924-31.

Epps, J. & Ambikairajah, E., 2005. Visualisation of reduced-dimension microarray data using Gaussian mixture models. *proceedings of the 2005 Asia-Pacific symposium on Information visualisation* , 45.

Fodorean, D., Idoumghar, L. & Szabo, L., 2013. Motorization for an Electric Scooter by Using Permanent-Magnet Machines Optimized Based on a Hybrid Metaheuristic Algorithm. *Vehicular Technology, IEEE Transactions on*, 62(1), pp.39-49.

Fonollosaa, o., Sheika, S., Huertaa, R. & Marcob, S., 2015. Reservoir computing compensates slow response of chemosensor arrays exposed to fast varying gas concentrations in continuous monitoring. *Sensors and Actuators B: Chemical*, 215, pp.618–29.

Franken, N. & Engelbrecht, A.P., 2004. PSO Approaches To Coevolve IPD Strategies. *Evolutionary Computation, 2004. CEC2004. Congress on , vol.1, no.pp. 356- 363 Vol.1, 19-23 June 2004*, 1, pp.356-63.

Franken, N. & Engelbrecht, A.P., 2005. Particle Swarm Optimization Approaches To Coevolve Strategies For The Iterated Prisoner's Dilemma. *Evolutionary Computation, IEEE Transactions on , vol.9, no.6, Dec. 2005*, 9(6), pp.562- 579.

Goldberg, D.E., 1989. *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison-Wesley.

Haitao Li, Xiaoyi Mu, Yuning Yang & Mason, A.J., 2014. Low Power Multimode Electrochemical Gas Sensor Array System for Wearable Health and Safety Monitoring. *Sensors Journal, IEEE*, 14(10), pp.3391-99. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6851860.

Hardin, C.T., 2005. WoSP: A Multi-optima Particle Swarm Algorithm. *in Proceedings IEEE Congress on Evolutionary Computation*, pp.727- 734.

Haykin, S., 1998. *Neural Networks: A Comprehensive Foundation (International Edition)*. 2nd ed. Prentice Hall.

Helwig, S., Branke, J. & Mostaghim, S.M., 2013. Experimental Analysis of Bound Handling Techniques in Particle Swarm Optimization. *Evolutionary Computation, IEEE Transactions on* , 17(2), pp.259-71.

Higashi, N. & Iba, H., 2003. Particle Swarm Optimisation With Gaussian Mutation. *Proceedings Of The IEEE Swarm Intelligence Symposium 2003 (SIS 2003), Indianapolis, Indiana, USA.*, pp.72-79.

Holland, J., 2000. *Emergence: From Chaos To Order*. Oxford University Press.

Hölldobler, B. & Wilson, E.O., 2008. *The Superorganism: The Beauty Elegance and Strangeness of Insect Societies*. W.W. Norton.

Huang, T. & Mohan, A.S., 2005. A Hybrid Boundary Condition For Robust Particle Swarm Optimization. *Antennas and Wireless Propagation Letters, IEEE*, 4, pp.112 - 117.

Huang, T. & Mohan, A.S., 2005. Significance Of Neighborhood Topologies For The Reconstruction Of Microwave Images Using Particle Swarm Optimization. *Microwave Conference Proceedings, 2005. APMC 2005. Asia-Pacific Conference Proceedings , vol.1, no.*, 1, pp.4 pp.-.

Hu, X. & Eberhart, R., 2002. Multi-objective Particle Swarm Optimisation Using Dynamic Neighbourhoods. *Proceedings Of The IEEE Congress On Evolutionary Computation (CEC 2002), Honolulu, HawaiI USA.*, pp.1677-81.

Hu, W. & Tan, Y., 2015. Prototype Generation Using Multiobjective Particle Swarm Optimization for Nearest Neighbor Classification. *IEEE Transactions on Cybernetics*, PP(99), pp.1-13. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7300413.

Iacoban, R., Reynolds, R.G. & Brewster, J., 2003a. Cultural Swarms: Assessing The Impact Of Culture On Social Interaction And Problem Solving. *Proceedings Of The 2003 IEEE Swarm Intelligence Symposium. SIS'03 (cat. No.03ex706)*, pp.212-19.

Iacoban, R., Reynolds, R.G. & Brewster, J., 2003b. Cultural Swarms: Modelling The Impact Of Culture On Social Interaction And Problem Solving. *Proceedings Of The 2003 IEEE Swarm Intelligence Symposium. SIS'03 (cat. No.03ex706)*, pp.205-11.

Imrie, C.E., Duncan, S. & Korre, A., 2000. River Flow Prediction Using Artificial Neural Networks: Generalisation Beyond The Calibration Range. *Journal of Hydrology*, 233(1-4), pp.138-53.

Ince, T., Kiranyaz, S. & Gabbouj, M., 2009. A Generic and Robust System for Automated Patient-Specific Classification of ECG Signals. *Biomedical Engineering, IEEE Transactions on*, 56(5), pp.1415-26.

Ismail, A. & Engelbrecht, A.P., 1999. Training Product Units In Feedforward Neural Networks Using Particle Swarm Optimization. *In: Development and Practice of Artificial Intelligence Techniques, VB Bajic, D Sha (eds), Proceedings of the International Conference on Artificial Intelligence, Durban, South Africa*, pp.#36-40.

Janson, S. & Middendorf, M., 2003. A Hierarchical Particle Swarm Optimizer. *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on , vol.2, no.pp.770- 776 Vol.2, 8-12 Dec. 2003*, 2, pp.770- 776 Vol.2.

Janson, S. & Middendorf, M., 2005. A Hierarchical Particle Swarm Optimizer And Its Adaptive Variant. *Systems, Man and Cybernetics, Part B, IEEE Transactions on , vol.35, no.6pp. 1272- 1282, Dec. 2005*, 35(6), pp.1272-82.

Jian, W., Yun-can Xue & Ji-xin Qian, 2004. An Improved Particle Swarm Optimization Algorithm With Neighborhoods Topologies. *Machine Learning And Cybernetics, 2004. Proceedings Of 2004 International Conference On volume 4, 26-29 Aug. 2004*, 4, pp.2332 - 2337.

Jing Liu, Wenbo Xu & Jun Sun, 2005. Quantum-behaved Particle Swarm Optimization With Mutation Operator. *Tools with Artificial Intelligence, 2005. ICTAI 05. 17th IEEE International Conference on , vol., no.pp. 237- 240, 14-16 Nov. 2005*, pp.237- 240.

Johnson, S., 2002. *Emergence: The Connected Lives Of Ants, Brains, Cities And Software*. Penguin Books.

Junfeng Chen, Ziwu Ren, Z. & Xinnan Fan, 2006. Particle Swarm Optimization With Adaptive Mutation And Its Application Research In Tuning Of PID Parameters. *Systems and Control in Aerospace and Astronautics, 2006. ISSCAA 2006. 1st International Symposium on , vol., no.pp. 990-994, 19-21 Jan. 2006*, pp.990- 994.

Jun-Jie, X. & Zhan-Hong, X., 2005. An Extended Particle Swarm Optimizer. *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, pp.5-.

Kachroudi, S., Grossard, M. & Abroug, N., 2012. Predictive Driving Guidance of Full Electric Vehicles Using Particle Swarm Optimization. *Vehicular Technology, IEEE Transactions on* , 61(9), pp.3909-19.

Kadirkamanathan, V., Selvarajah, K. & Fleming, P.J., 2006. Stability Analysis Of The Particle Dynamics In Particle Swarm Optimizer. *Evolutionary Computation, IEEE Transactions on , June 2006*, 10(3), pp.245- 255.

Kennedy, J., 1999. Small Worlds And Mega-minds: Effects Of Neighborhood Topology On Particle Swarm Performance. *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on Volume 3, 6-9 July 1999*, 3, pp.1931-38.

Kennedy, J., 2004. Bare Bones Particle Swarm. *Proceedings Of The 2003 IEEE Swarm Intelligence Symposium. SIS'03 (cat. No.03ex706)*, pp.pp.80-87.

Kennedy, J., 2005. Why Does It Need Velocity? *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE 8-10 June 2005*, pp. 38-44.

Kennedy, J. & Eberhart, R., 1995. Particle Swarm Optimisation. *Neural Networks, 1995. Proceedings. IEEE International Conference On , Volume: 4 , 27 Nov.-1 Dec. 1995*, 4, pp.pp.1942-1948.

Kennedy, J., Eberhart, R.C. & Shi, Y., 2001. *Swarm Intelligence*. Morgan Kaufmann.

Kennedy, J. & Mendes, R., 2002. Population Structure and Particle Swarm Performance. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2002), Honolulu, Hawaii USA.*.

Kennedy, J. & Mendes, R., 2003. Neighborhood Topologies In Fully-Informed And Best-of-Neighborhood Particle Swarms. *Proceedings Of The 2003 IEEE International Workshop On Soft Computing In Industrial Applications 2003 (SMCIA/03)*, pp.pp.445-450.

Kenny, I., 2005. Differential Evolution Verses Particle Swarm Optimisation Or The Search For Differential Swarm Optimisation. *Technical Report*.

Kentzoglanakis, K. & Poole, M., 2012. A Swarm Intelligence Framework for Reconstructing Gene Networks: Searching for Biologically Plausible Architectures. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on* , 9(2), pp.358-71.

Kerh, T. & Lee, C.S., 2006. Neural Networks Forecasting of Flood Discharge at an Unmeasured Station Using River Upstream Information. *Advances in Engineering Software 37*, 37, pp.533-43.

Keskin, M.E., Taylan, D. & Kucuksille, E.U., 2013. Data mining process for modeling hydrological time series. *Hydrology Research*, 44(1), pp.78-88.

Khan, A.A. & Brown, A.K., 2012. Intelligent z-plane boundary condition-particle swarm optimiser for small array pattern synthesis. *Microwaves, Antennas & Propagation, IET* , 6(14), pp.1598-607.

Ki-Baek Lee & Jong-Hwan Kim, 2013. Multiobjective Particle Swarm Optimization With Preference-Based Sort and Its Application to Path Following Footstep Optimization for Humanoid Robots. *Evolutionary Computation, IEEE Transactions on*, 17(6), pp.755-66.

Kirby, C., Newson, M.D. & Gilman, K., 1991. Plynlimon research: The first two decades. *Institute of Hydrology*.

Kit Yan Chan, Dillon, T.S. & Chang, E., 2013. An Intelligent Particle Swarm Optimization for Short-Term Traffic Flow Forecasting Using on-Road Sensor Systems. *Industrial Electronics, IEEE Transactions on*, 60(10), pp.4714-25.

Krohling, R.A., 2005. Gaussian particle swarm with jumps. *in Proc. IEEE CEC*, pp.1080–85.

Krohling, R.A. & Mendel, E., 2009. Bare bones particle swarm optimization with Gaussian or Cauchy jumps. *in Proc. 11th Conf. CEC*, pp.3285–91.

Kulkarni, R.V. & Venayagamoorthy, G.K., 2011. Particle Swarm Optimization in Wireless-Sensor Networks: A Brief Survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 41(2), pp.262-67.

Kuok, K.K., Harun, S. & Shamsuddin, S.M., 2010. Particle Swarm Optimization Feedforward Neural Network for Modeling Runoff. *International Journal of Environmental Science and Technology*, 7(1), pp.67-78.

Lei, X. & Liying, J., 2005. Global Optimal ICA And Its Application In Brain MEG Data Analysis. *Neural Networks and Brain, 2005. ICNN&B '05. International Conference on , vol.1, no., 13-15 Oct. 2005*, 1, pp.353-57.

Li, X., 2004. Adaptively Choosing Neighbourhood Bests Using Species In A Particle Swarm Optimizer For Multimodal Function Optimization. *Proceedings of the Genetic and Evolutionary Computation Conference 2004 (GECCO 2004), Seattle, WA, USA.*, pp.105-16.

Liang, J.J., Qin, A.K., Suganthan, P.N. & Baskar, S., 2006. Comprehensive Learning Particle Swarm Optimizer For Global Optimization Of Multimodal Functions. *Evolutionary Computation, IEEE Transactions on , vol.10, no.3pp. 281- 295, June 2006*, 10(3), pp.281- 295.

Liang, J.J. & Suganthan, P.N., 2005. Dynamic Multi-swarm Particle Swarm Optimizer With Local Search. *Evolutionary Computation, 2005. The 2005 IEEE Congress on , vol.1, no.pp. 522- 528 Vol.1, 2-5 Sept. 2005*, 1, pp.522- 528 Vol.1.

Lin Lu, Qi Luo, Jun-yong Liu & Chuan Long, 2008. An Improved Particle Swarm Optimization For Reconfiguration Of Distribution Network. *Natural Computation, 2008. ICNC '08. Fourth International Conference on , vol.4, no.*, 4, pp.453-57.

Little, M.A. et al., 2009. Suitability of dysphonia measurements for telemonitoring of Parkinson's disease. *IEEE Transactions on Biomedical Engineering.*, 56(4), pp.1015-22.

Little, M.A. et al., 2007. Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection. *BioMedical Engineering OnLine 2007, 6:23 (26 June 2007)*.

Lixin Tang & Xianpeng Wang, 2013. A Hybrid Multiobjective Evolutionary Algorithm for Multiobjective Optimization Problems. *Evolutionary Computation, IEEE Transactions on*, 17(1), pp.20-45.

Li, H., Yi, Y., Li, X. & Guo, Z., 2013. Human activity recognition based on HMM by improved PSO and event probability sequence. *Systems Engineering and Electronics, Journal of* , 24(3), pp.545-54.

Lloyd., S.P., 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2), pp.129–37.

Lloyd, S.P., 1957. Least square quantization in PCM. *Bell Telephone Laboratories Paper*.

Lorenz, E.N., 1963. Deterministic nonperiodic flow. *J. Atmos. Sci. 20*, 20, pp.130–41.

Macready, W.G. & Wolpert, D.H., 1995. What Makes An Optimization Problem Hard?.

Maltese, J., Ombuki-Berman, B. & Engelbrecht, A., 2015. Co-operative Vector-Evaluated Particle Swarm Optimization for Multi-objective Optimization. *Computational Intelligence, 2015 IEEE Symposium Series on*, pp.1294-301. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7376761.

Mandal, B. & Si, T., 2015. Opposition based Particle Swarm Optimization with exploration and exploitation through gbest. *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, pp.245-50. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7275616.

Marc, V. & Robinson, M., 2007. The Long-Term Water Balance (1972-2004) Of Upland Forestry And Grassland At Plynlimon, Mid-Wales. *Hydrology & Earth System Sciences*, 11(1), pp.44-60.

Matsumura, Y. et al., 2013. A (μ, λ) evolutionary and particle swarm hybrid algorithm, with an application to dinosaur gait optimization. *Computational Intelligence & Applications (IWCIA), 2013 IEEE Sixth International Workshop on*, pp.89 - 93.

Mendes, R., Kennedy, J. & Neves, J., 2004. Watch Thy Neighbor Or How The Swarm Can Learn From Its Environment. *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706)*, pp.88-94.

Meng-xin, Gao-junru, Li-peng & She-na, 2014. The application of dynamic multi-swarm particle swarm optimization in the optimial dispatch of loads of thermal power unit. *Electronics, Computer and Applications, 2014 IEEE Workshop on*, pp.573-76. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6845685.

Messerschmidt, L. & Engelbrecht, A.P., 2004. Learning To Play Games Using A PSO-based Competitive Learning Approach. *Evolutionary Computation, IEEE Transactions On volume 8, Issue 3, June 2004*, pp.pp.280 - 288.

Michalewicz, Z., 1996. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag Berlin And Heidelberg Gmbh & Co. K.

Michalewicz, Z. & Fogel, D.B., 2004. *How To Solve It: Modern Heuristics*. Springer-Verlag Berlin And Heidelberg Gmbh & Co. K.

Mikki, S.M. & Kishk, A.A., 2007. Hybrid Periodic Boundary Condition for Particle Swarm Optimization. *Antennas and Propagation, IEEE Transactions on* , 55(11), pp.3251-56.

Mount, N.J. & Abrahart, R.J., 2011. Discussion Of ' River Flow Estimation From Upstream Flow Records By Artificial Intelligence Methods' By M. E. Turan, M. A. Yurdusev [J. Hydrol. 369 (2009) 71-77]. *Journal of Hydrology, 5 January 2011*, 396(1-2), pp.193-96.

Mount, N. & Stott, T., 2008. A Discrete Bayesian Network To Investigate Suspended Sediment Concentrations In An Alpine Proglacial Zone. *Hydrological Processes Volume 22, Issue 18, 30 August 2008*, 22(8), pp.3772-84.

Munlin, M.A. & Anantathanavit, M., 2015. Hybrid K-means and Particle Swarm Optimization for symmetric Traveling Salesman Problem. *Industrial Electronics and Applications (ICIEA), 2015 IEEE 10th Conference on*, pp.671-76. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7334194.

Murphy, K.P., 2012. *Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning Series)*. MIT Press.

Ning Li, Yuan-Qing Qin, De-Bao Sun & Tong Zo, 2004. Particle Swarm Optimization With Mutation Operator. *Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on*, 4, pp.2251- 2256.

Peng, T. et al., 2014. Modeling Cell–Cell Interactions in Regulating Multiple Myeloma Initiating Cell Fate. *Biomedical and Health Informatics, IEEE Journal of* , 18(2), pp.484-91.

Peram, T., Veeramachaneni, K. & Mohan, C.K., 2003. Fitness-distance-ratio Based Particle Swarm Optimization. *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE 24-26 April 2003*, pp.174 – 181.

Po-Hung Chen, 2012. Two-Level Hierarchical Approach to Unit Commitment Using Expert System and Elite PSO. *Power Systems, IEEE Transactions on*, 27(2), pp.780-89.

Poli, R., Kennedy, J. & Blackwell, T., 2007. Particle swarm optimization: An overview. *Swarm Intelligence*, 1(1), pp.33-57.

Price, K.V., 2005. Differential Evolution: A Practical Approach To Global Optimization. *Springer-Verlag Berlin and Heidelberg GmbH & Co. K (Jun 2005) Proceedings. 2006 IEEE International Symposium on*, pp.137- 140.

Qais, M. & AbdulWahid, Z., 2013. A new method for improving particle swarm optimization algorithm (TriPSO). *Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference on* , pp.1-6.

Qu, B.Y., Suganthan, P.N. & Das, S., 2013. A Distance-Based Locally Informed Particle Swarm Model for Multimodal Optimization. *Evolutionary Computation, IEEE Transactions on*, 17(3), pp.387-402.

Ratnaweera, A., Halgamuge, S. & Watson, H.C., 2004. Self-organizing Hierarchical Particle Swarm Optimizer With Time-Varying Acceleration Coefficients. *Evolutionary Computation, IEEE Transactions On* , 8(3), pp.240-55.

Rezaei, H. & Azadi, S., 2009. Nonrigid Medical Image Registration Using Hierarchical Particle Swarm Optimization. *Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control, 2009. ICSCCW 2009. Fifth International Conference on , vol., no., pp.1-4, 2-4 Sept. 2009 doi: 10.1109/ICSCCW.2009.5379433*, pp.1-4.

Roberge, V., Tarbouchi, M. & Labonte, G., 2013. Comparison of Parallel Genetic Algorithm and Particle Swarm Optimization for Real-Time UAV Path Planning. *Industrial Informatics, IEEE Transactions on*, 0(1), pp.132-41.

Röhler, A.B. & Chen, S., 2011. An Analysis of Sub-swarms in Multi-swarm Systems. *in Lecture Notes in Artificial Intelligence, Vol. 7106: Proceedings of Joint Australasian Conference in Artificial Intelligence, D. Wang and M. Reynolds Eds. Springer-Verlag*, pp.271–80.

Röhler, A.B. & Chen, S., 2012. Multi-swarm hybrid for multi-modal optimization. *in Proceedings of the IEEE Congress on Evolutionary Computation*, pp.1759-66.

Rubio-Largo, A., Vega-Rodriguez, M.A., Gomez-Pulido, J.A. & Sanchez-Pérez, J.M., 2012. A Comparative Study on Multiobjective Swarm Intelligence for the Routing and Wavelength Assignment Problem. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(6), pp.1644-55.

Sarikhani, A. & Mohammed, O.A., 2011. Multiobjective Design Optimization of Coupled PM Synchronous Motor-Drive Using Physics-Based Modeling Approach. *Magnetics, IEEE Transactions on*, 47(5), pp.1266-69.

Scheutz, M., 2007. Real-Time Hierarchical Swarms For Rapid Adaptive Multi-Level Pattern Detection And Tracking. *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE , vol., no., pp.234-241, 1-5 April 2007 doi: 10.1109/SIS.2007.367943*, pp.234-41.

Scriven, I., Lu, J. & Lewis, A., 2010. Electromagnetic Noise Source Approximation for Finite-Difference Time-Domain Modeling Using Near-Field Scanning and Particle Swarm Optimization. *Electromagnetic Compatibility, IEEE Transactions on Compatibility, IEEE Transactions on*, 52(1), pp.89-97.

Secrest, B.R. & Lamont, G.B., 2003. Visualizing Particle Swarm Optimization -Gaussian Particle Swarm Optimization. *Swarm Intelligence Symposium 2003. Sis '03. Proceedings Of The 2003 IEEE*.

Senthilnath, J., Omkar, S.N., Mani, V. & Karthikeyan, T., 2013. Multiobjective Discrete Particle Swarm Optimization for Multisensor Image Alignment. *Geoscience and Remote Sensing Letters, IEEE* , 10(5), pp.1095-99.

Senthilnath, J. et al., 2012. Hierarchical Clustering Algorithm for Land Cover Mapping Using Satellite Images. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of* , 5(3), pp.762-68.

Shi, Y. & Eberhart, R.C., 1998a. Parameter Selection In Particle Swarm Optimization. *Evolutionary Programming Vii: Proceedings Of The Seventh Annual Conference On Evolutionary Programming, New York*, pp.591-600.

Shi, Y. & Eberhart, R.C., 1998b. A Modified Particle Swarm Optimizer. *Proceedings Of The IEEE Congress On Evolutionary Computation (CEC 1998), Piscataway, NJ*, pp.69-73.

Shin-jen Cheng, 2010. Inferring Hydrograph Components From Rainfall And Streamflow. *Journal Of The American Water Resources Association*, 46(6), pp.1171-91.

Siano, P. & Mokryani, G., 2013. Assessing Wind Turbines Placement in a Distribution Market Environment by Using Particle Swarm Optimization. *Power Systems, IEEE Transactions on*, 28(4), pp.3852-64.

Slade, W.H., Ressom, H.W., Musavi, M.T. & Miller, R.L., 2004. Inversion Of Ocean Color Observations Using Particle Swarm Optimization. *Geoscience and Remote Sensing, IEEE Transactions on Volume 42, Issue 9, Sept. 2004*, 42(9), pp.1915 - 1923.

Sneyers, R., 1997. Climate Chaotic Instability: Statistical Determination and Theoretical Background. *Environmetrics*, 8(5), pp.517–32.

Srivastava, L. & Singh, H., 2015. Hybrid multi-swarm particle swarm optimisation based multi-objective reactive power dispatch. *IET Generation, Transmission & Distribution*, 9(8), pp.727-39. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7106085.

Stacey, A., Jancic, M. & Grundy, I.., 2003. Particle Swarm Optimization With Mutation. *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on Volume 2, 8-12 Dec. 2003,* 2, pp.425 - 1430.

Storn, R. & Price, K., 1995. Differential Evolution - A Simple And Efficient Adaptive Scheme For Global Optimization Over Continuous Spaces. *Technical Report Tr-95-012*.

Stott, T. & Mount, N., 2007. Alpine Proglacial Suspended Sediment Dynamics In Warm And Cool Ablation Seasons: Implications For Global Warming. *Journal of Hydrology, Volume 332, Issues 3-4, 15 January 2007*, 332(3-4), pp.259-70.

Strogatz, S.H., 2004. *Sync: The Emerging Science Of Spontaneous Order*. Penguin Books.

Sun, J., Palade, V., Wu, X. & Fang, W., 2014. Multiple Sequence Alignment with Hidden Markov Models Learned by Random Drift Particle Swarm Optimization. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 11(1), pp.243-57.

Tabor, M., 1989. *Chaos and Integrability in Nonlinear Dynamics: An Introduction*. New York: Wiley.

Tapoglou, E. et al., 2013. Groundwater level forecasting under climate change scenarios using an artificial neural network trained with particle swarm optimization. *Hydrological Sciences Journal*, 58(7), Available at: http://dx.doi.org/10.1080/02626667.2013.838005.

Tetko, I.V., Livingstone, D.J. & Luik, A.I., 1995. Neural network studies. 1. Comparison of overfitting and overtraining. *J. Chem. Inf. Comput. Sci.*, 35, pp.826-33.

Tsanas, A., Little, M.A., McSharry, P.E. & Ramig, L.O., 2012b. Using the cellular mobile telephone network to remotely monitor Parkinson's disease symptom severity. *IEEE Transactions on Biomedical Engineering (submitted)*.

Tsanas, A. et al., 2012a. Novel speech signal processing algorithms for high-accuracy classification of Parkinson’s disease. *IEEE Transactions on Biomedical Engineering*, 59(5), pp.1264-71.

Tsujimoto, T., Shindo, T., Kimura, T. & Jin'no, K., 2012. A relationship between network topology and search performance of PSO. *IEEE Congress on Evolutionary Computation, Brisbane, QLD, 2012*, pp.1-6.

Turan, M.E. & Yurdusev, M.A., 2009. River Flow Estimation from Upstream Flow Records By Artificial Intelligence Methods. *Journal of Hydrology 369*, 369.

Turan, M.E. & Yurdusev, M.A., 2011. Reply to discussion on “River flow estimation from upstream flow records by artificial intelligence methods” by M.E. Turan and M.A. Yurdusev [J. Hydrol. 369 (2009) 71–77]. *Journal of Hydrology*, 409(1-2), pp.580-81.

van den Bergh, F., 2002. An Analysis Of Particle Swarm Optimizers. *Ph.D. Dissertation, Dept. Comput. Sci., Univ. Pretoria, Pretoria, South Africa*.

van den Bergh, F. & Engelbrecht, A.P., 2002. A New Locally Convergent Particle Swarm Optimizer. *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics 2002 (SMC 2002),* pp.96-101.

van den Bergh, F. & Engelbrecht, A.P., 2004. A Cooperative Approach To Particle Swarm Optimization. *Evolutionary Computation, IEEE Transactions On volume 8, Issue 3, June 2004*, 8(3), pp.225 - 239.

van der Merwe, D.W. & Engelbrecht, A.P., 2003. Data Clustering Using Particle Swarm Optimisation. *Proceedings Of The 2003 IEEE Swarm Intelligence Symposium. SIS'03 (cat. No.03ex706)*, pp.215-20.

Veeramachaneni, K., Peram, T., Mohan, C.K. & Osadciw, L.A., 2003. Optimization Using Particle Swarms With Near Neighbor Interactions. *Lecture Notes in Computer Science (LNCS) No. 2723: Proceedings of the Genetic and Evolutionary Computation Conference 2003 (GECCO 2003), Chicago, IL, USA.*, 2723, pp.110-21.

Wachowiak, M.P. et al., 2004. An Approach To Multimodal Biomedical Image Registration Utilizing Particle Swarm Optimization. *Evolutionary Computation, IEEE Transactions On*, 8(3), pp.289-301.

Wang, X. & Li, J., 2004. Hybrid Particle Swarm Optimization With Simulated Annealing. pp.pp.2402 - 2405.

Wang, H. et al., 2008. An improved particle swarm optimization with adaptive jumps. *in Proc. IEEE CEC*, pp.392–97.

Ward, N.D., Gebert, J.A. & Weggel, J.R., 2009. Hydraulic Study of the Chesapeake and Delaware Canal. *Journal of Waterway, Port, Coastal & Ocean Engineering. Jan/Feb2009*, 135(1), pp.24-30.

Watts, D.J., 2003. *Small Worlds: The Dynamics of Networks between Order and Randomness*. Princeton University Press (24 Nov 2003).

Watts, D.J., 2004. *Six Degrees: The Science of a Connected Age*. Vintage.

Watts, D.J., Dodds, P.S. & Newman, M.E.J., 2002. Identity and Search in Social Networks. *Science* , 296 (5571), pp.1302–05.

Watts, D.J., Muhamad, R., Medina, D.C. & Dodds, P.S., 2005. Multiscale, Recurrent Epidemics In A Hierarchical Compartment Model. *Proceedings of the National Academy of Sciences, 102(32)*, pp.11157-62.

Watts, D.J. & Strogatz, S.H., 1998. Collective dynamics of 'small-world' networks. *Nature 393*, pp.440-42.

Weibo Wang & Quanyuan Feng, 2010. A Hierarchical Particle Swarm Optimization Algorithm Combined With Chaotic Search. *Computer and Communication Technologies in Agriculture Engineering (CCTAE), 2010 International Conference On , vol.3, no., pp.435-438, 12-13 June 2010*, 3, pp.435-38.

Wenhua, H., Ping, Y. & Haixia, R., 2011. Application of Damping-Boundary-based PSO to MFL Signal Inversion. *Measuring Technology and Mechatronics Automation (ICMTMA), 2011 Third International Conference on* , 1, pp.532-35.

Wen-Jun, Z. & Xiao-Feng, X., 2003. DEPSO: Hybrid Particle Swarm With Differential Evolution Operator. *Systems, Man And Cybernetics, 2003. IEEE International Conference On, Volume: 4, 5-8 Oct. 2003*, 4, pp.3818-21.

Wen-Tsai Sung, Jui-Ho Chen & Kung-Wei Chang, 2014. Mobile Physiological Measurement Platform With Cloud and Analysis Functions Implemented via IPSO. *Sensors Journal, IEEE*, 14(1), pp.111-23.

Wilson, E.O., 1971. *The Insect Societies.* Belknap Press.

Wilson, E.O., 1975/2000. *Sociobiology: The New Synthesis*. Belknap Press.

Wolpert, D.H. & Macready, W.G., 1995. No Free Lunch Theorems For Search.

Wolpert, D.H. & Macready, W.G., 1997. No Free Lunch Theorems For Optimization. pp.pp.67 - 82.

Wu, Z. & Chow, T.W.S., 2013. Binary neighbourhood field optimisation for unit commitment problems. *Generation, Transmission & Distribution, IET* , 7(3), pp.298-308.

Wu, Q., Yan, H.-S. & Yang, H.-B., 2008. A Forecasting Model Based Support Vector Machine and Particle Swarm Optimization. *Power Electronics and Intelligent Transportation System, 2008. PEITS '08. Workshop on* , pp.218-22.

Xiangtao Li & Minghao Yin, 2013. Multiobjective Binary Biogeography Based Optimization for Feature Selection Using Gene Expression Data. *NanoBioscience, IEEE Transactions on* , 12(4), pp.343-53.

Xiaodong Li & Xin Yao, 2012. Cooperatively Coevolving Particle Swarms for Large Scale Optimization. *Evolutionary Computation, IEEE Transactions on*, 16(2), pp.210-24.

Xiao, X. et al., 2003. Gene clustering using self-organizing maps and particle swarm optimization. *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*, p.10.

Xiao-Feng Xie, Wen-Jun Zhang & Zhi-Lian Yang, 2002. Hybrid Particle Swarm Optimizer With Mass Extinction. *Communications, Circuits And Systems And West Sino Expositions, IEEE 2002 International Conference On volume 2, 29 June-1 July 2002*, 2, pp.1170 – 1173.

Xu, R., Cai, X. & Wunsch, D.C., 2006. Gene Expression Data for DLBCL Cancer Survival Prediction with A Combination of Machine Learning Technologies. *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, pp.894-97.

Xue Wang & Sheng Wang, 2011. Hierarchical Deployment Optimization for Wireless Sensor Networks. *Mobile Computing, IEEE Transactions on*, 10(7), pp.1028-41.

Xu, S. & Rahmat-Samii, Y., 2007. Boundary Conditions in Particle Swarm Optimization Revisited. *Antennas and Propagation, IEEE Transactions on* , 55(3), pp.760-65.

Yan-Liang Li, Wei Shao, You, L. & Bing-Zhong Wang, 2013. An Improved PSO Algorithm and Its Application to UWB Antenna Design. *Antennas and Wireless Propagation Letters, IEEE*, 7(3), pp.1236-39.

Yen, G.G. & Wen-Fung Leong, 2009. Dynamic Multiple Swarms in Multiobjective Particle Swarm Optimization. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39(4), pp.890-911.

Ying Li, Hongli Gong, Dagan Feng & Zhang, Y., 2011. An Adaptive Method of Speckle Reduction and Feature Enhancement for SAR Images Based on Curvelet Transform and Particle Swarm Optimization. *Geoscience and Remote Sensing, IEEE Transactions on*, 49(8), pp.3105-16.

Yong-ling Zheng, Long-Hua Ma, Zhang Li-yan & Ji-xin Qian, 2003. On The Convergence Analysis And Parameter Selection In Particle Swarm Optimization. *Machine Learning and Cybernetics, 2003 International Conference on Volume 3, 2-5 Nov. 2003*, 3, pp.1802 - 1807.

Yuwono, M., Su, S.W., Moulton, B.D. & Nguyen, H.T., 2012. Optimization strategies for rapid centroid estimation. *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp.6212-15. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6347413.

Yuwono, M., Su, S.W., Moulton, B.D. & Nguyen, H.T., 2014. Data Clustering Using Variants of Rapid Centroid Estimation. *Evolutionary Computation, IEEE Transactions on*, 18(3), pp.366-77.

Zeng, X., Zhang, Y. & Guo, Y., 2011. Polyphase coded signal design for MIMO radar using MO-MicPSO. *Systems Engineering and Electronics, Journal of* , 22(3), pp.381-86.

Zhan, Z.-H., Zhang, J., Li, Y. & Chung, H.S.-H., 2009. Adaptive Particle Swarm Optimization. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(6), pp.1362-81.

Zhao, S.Z., Liang, J.J., Suganthan, P.N. & Tasgetiren, M.F., 2008. Dynamic Multi-Swarm Particle Swarm Optimizer with Local Search for Large Scale Global Optimization. *in Proceedings IEEE Congress on Evolutionary Computation*, pp.3845–52.

Zhi-Hui Zhan, Jun Zhang, Yun Li & Yu-hui Shi, 2011. Orthogonal Learning Particle Swarm Optimization. *Evolutionary Computation, IEEE Transactions on*, 15(6), pp.832-47.

Zhi, X.H. et al., 2004. A Discrete PSO Method For Generalized TSP Problem. *Machine Learning and Cybernetics, Proceedings of 2004 International Conference on , 26-29 Aug. 2004*, 4, pp.2378-2383.

Appendix A River Severn Results

	Matrix										
Year	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990
1980	0.996839	0.997227	0.996854	0.996452	0.996240	0.994221	0.993638	0.993361	0.995548	0.996922	0.994534
1981	0.997534	0.997834	0.997698	0.997094	0.996478	0.994002	0.994384	0.994446	0.995948	0.997574	0.995318
1982	0.990806	0.989359	0.991478	0.985667	0.986877	0.981749	0.983913	0.986615	0.985281	0.990348	0.985454
1983	0.982645	0.985706	0.983107	0.989361	0.982568	0.980592	0.970462	0.967039	0.976651	0.979486	0.971078
1984	0.997678	0.998180	0.996824	0.997005	0.997889	0.995788	0.997286	0.997348	0.997705	0.997553	0.997512
1985	0.990863	0.992703	0.988377	0.994015	0.994761	0.995298	0.994228	0.993163	0.994218	0.990811	0.993491
1986	0.988676	0.990308	0.987268	0.990377	0.992413	0.993403	0.993315	0.992345	0.993096	0.990003	0.992862
1987	0.994598	0.995321	0.993491	0.994744	0.996213	0.995347	0.996180	0.996437	0.996143	0.995265	0.996250
1988	0.992434	0.993526	0.991118	0.993025	0.995091	0.995103	0.995523	0.995075	0.995512	0.993502	0.995400
1989	0.993406	0.993923	0.992656	0.993460	0.994315	0.993412	0.993892	0.993894	0.994345	0.993933	0.994187
1990	0.993094	0.994124	0.992269	0.992635	0.995338	0.994882	0.996370	0.995802	0.996337	0.994608	0.996430

Table 12: Shows the R-squared value for each year, using the matrix calculated for the corresponding year. The rows indicate the years, whereas the columns indicate the matrix used. The best performing matrix for each year (row), is shown as white text on a black background.

	Matrix											
Year	1980		1981		1982		1983		1984		1985	
	PSO	MLR	PSO	MLR	PSO	MLR	PSO	MLR	PSO	MLR	PSO	MLR
1980	0.997	0.998	0.997	0.997	0.997	0.997	0.996	0.995	0.996	0.997	0.994	0.997
1981	0.998	0.998	0.998	0.998	0.998	0.997	0.997	0.997	0.996	0.998	0.994	0.998
1982	0.991	0.991	0.989	0.991	0.991	0.992	0.986	0.984	0.987	0.991	0.982	0.991
1983	0.983	0.985	0.986	0.987	0.983	0.978	0.989	0.991	0.983	0.986	0.981	0.987
1984	0.998	0.998	0.998	0.998	0.997	0.998	0.997	0.996	0.998	0.998	0.996	0.998
1985	0.991	0.995	0.993	0.995	0.988	0.994	0.994	0.993	0.995	0.995	0.995	0.995
1986	0.989	0.994	0.990	0.994	0.987	0.994	0.990	0.992	0.992	0.994	0.993	0.994
1987	0.995	0.996	0.995	0.996	0.993	0.996	0.995	0.994	0.996	0.996	0.995	0.996
1988	0.992	0.996	0.994	0.996	0.991	0.996	0.993	0.993	0.995	0.996	0.995	0.996
1989	0.993	0.995	0.994	0.995	0.993	0.995	0.993	0.993	0.994	0.995	0.993	0.995
1990	0.993	0.996	0.994	0.996	0.992	0.996	0.993	0.991	0.995	0.996	0.995	0.996

	Matrix									
Year	1986		1987		1988		1989		1990	
	PSO	MLR	PSO	MLR	PSO	MLR	PSO	MLR	PSO	MLR
1980	0.994	0.997	0.993	0.997	0.996	0.997	0.997	0.997	0.995	0.997
1981	0.994	0.998	0.994	0.997	0.996	0.998	0.998	0.998	0.995	0.997
1982	0.984	0.992	0.987	0.992	0.985	0.992	0.990	0.992	0.985	0.992
1983	0.970	0.985	0.967	0.979	0.977	0.983	0.979	0.985	0.971	0.978
1984	0.997	0.998	0.997	0.998	0.998	0.998	0.998	0.998	0.998	0.998
1985	0.994	0.995	0.993	0.995	0.994	0.995	0.991	0.995	0.993	0.994
1986	0.993	0.994	0.992	0.994	0.993	0.994	0.990	0.994	0.993	0.994
1987	0.996	0.996	0.996	0.996	0.996	0.996	0.995	0.996	0.996	0.996
1988	0.996	0.996	0.995	0.996	0.996	0.996	0.994	0.996	0.995	0.996
1989	0.994	0.995	0.994	0.995	0.994	0.995	0.994	0.995	0.994	0.995
1990	0.996	0.996	0.996	0.996	0.996	0.996	0.995	0.996	0.996	0.997

Table 13: PSO and MLR R-squared value for each year, using the matrix calculated for the corresponding year for the River Severn data.

1980

0.9020003	0.2258762	-0.0854694
1.0944389	-0.5518632	0.5336288
0.6702101	0.7906727	0.2933398
0.0082921	0.5325794	-0.1875138
0.7079931	1.1027036	1.0074612

1981

-0.1388691	0.2967863	1.0017402
0.6853794	0.2295014	0.354013
1.0373192	0.3912027	-0.7237264
-0.4901823	0.7174507	0.5141762
0.7120208	0.1442158	-0.1036047

1982

0.8321009	0.4835061	-0.0701966
-0.5671766	0.7972709	0.7893756
0.4516507	-0.0098882	0.9244932
-0.5146601	3.0009955	-0.9377654
-1.1271209	0.8984909	1.0875029

1983

-0.4299736	1.2318652	0.0471777
0.8380171	0.1159662	0.8155107
0.1139792	0.2956568	-0.3303402
0.3916767	1.1350075	0.2977395
-0.6509684	0.378837	-0.1824708

1984

0.95477	0.2355716	-0.0758961
-0.5855584	1.4229962	0.3453872
0.9850552	-0.0156539	-0.0221437
0.9469464	1.0458464	-0.4724834
0.548535	0.2127135	1.3791959

1985

-0.2964126	0.9906163	0.3738213
0.292582	0.8056453	0.2099769
0.3378603	-1.2356507	1.4281871
0.979089	0.7522917	-1.0892461
-0.3223854	0.7779104	0.3959591

1986

0.5409889	0.0436789	0.7458592
-0.7763194	0.61663	0.8696689
1.3573117	-0.0058835	0.2561779
-0.0365886	-0.2661736	0.6693748
-0.0107959	1.5350743	0.2345025

1987

0.1372002	0.791111	0.2502757
0.8805642	0.1514472	-0.4326657
0.5096498	1.059988	0.9564073
0.9892667	-0.0498011	-1.7321703
-0.4052139	-1.2543292	0.3359835

1988

0.4302483	0.7864755	0.1063979
-1.0734431	0.7203413	1.2172224
-0.1009305	0.679235	0.8737182
-0.5291489	0.38397	0.2241048
0.9612146	0.3265471	1.0282882

1989

0.4302483	0.7864755	0.1063979
-1.0734431	0.7203413	1.2172224
-0.1009305	0.679235	0.8737182
-0.5291489	0.38397	0.2241048
0.9612146	0.3265471	1.0282882

1990

0.5149596	0.3714018	0.4912907
0.2834194	-0.0544639	0.4420213
-0.0468538	0.7184712	0.9660215
1.2425462	0.1658419	0.1055337
-0.0699174	1.2824912	0.7892265

Table 14: The matrices used to produce the results shown above in Table 12.

	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990
Absolute Maximum Error	0.331357241	0.298768385	0.530524	0.553257126	0.202459984	0.215174	1.079355482	0.426909269	0.382871	0.585613975	0.36657
Mean Absolute Error	0.02553114	0.022345887	0.028773	0.035123682	0.015817037	0.022603	0.026416795	0.019454292	0.0305	0.029682618	0.022984
Mean Error	-0.00279994	0.000160084	-0.00137	0.009874805	0.001755782	0.000715	-0.00168795	-0.00020902	0.006351	0.005498083	0.003258
Relative Absolute Error	0.055196402	0.042076961	0.066144	0.066523363	0.03774992	0.0595	0.050222136	0.04608796	0.068921	0.062250191	0.047359
Root Mean Squared Error	0.820477645	0.799976339	1.154756	1.580000352	0.540903214	0.716143	1.418531229	0.766620913	0.913229	1.176722403	0.825732
Score Result	9.344397066	9.344397066	9.344397	9.344397066	9.344397066	9.344397					

Table 20: Sample per year statistics from experiment run on the 11 years of data.