JYVÄSKYLÄ STUDIES IN COMPUTING
81

Ferrante Neri

Fitness Diversity Adaptation in Memetic Algorithms

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 81

Ferrante Neri

Fitness Diversity Adaptation in Memetic Algorithms

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston Villa Ranan Paulaharjun salissa
marraskuun 30. päivänä 2007 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in the building Villa Rana, Paulaharju Hall, on November 30, 2007 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2007

Fitness Diversity Adaptation in Memetic Algorithms

JYVÄSKYLÄ STUDIES IN COMPUTING 81

Ferrante Neri

Fitness Diversity Adaptation in Memetic Algorithms

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2007

Editors
Tommi Kärkkäinen
Department of Mathematical Information Technology, University of Jyväskylä
Irene Ylönen, Pekka Olsbo
Publishing Unit, University Library of Jyväskylä

URN:ISBN:978-951-39-8043-6
ISBN 978-951-39-8043-6 (PDF)
ISSN 1456-5390

ISBN 978-951-39-2979-4
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2007

ABSTRACT

Neri, Ferrante
Fitness Diversity Adaptation in Memetic Algorithms
Jyväskylä: University of Jyväskylä, 2007, 80 p.(+included articles)
(Jyväskylä Studies in Computing
ISSN 1456-5390; 81)
ISBN 978-951-39-2979-4
Finnish summary
Diss.

This work proposes novel tailored implementations of Memetic Algorithm for some specific classes of problems and, at the same time, proposes novel general ideas in Computational Intelligence algorithmic philosophy. Much emphasis is given to adaptation and coordination of the local searchers. Several adaptive schemes have been designed; all of them resorting to a measurement of the fitness diversity as an estimation of the diversity amongst individuals of the population. According to the philosophy common to the algorithms included in this thesis, the algorithm should behave like an intelligent structure, thus able to analyze online the optimization process and then apply countermeasures necessary for continuing and efficiently finalizing the search.

This thesis includes eight articles addressing applications having various natures such as biology, image processing, telecommunication, and electrical engineering. Each problem has been analyzed by considering the features of each fitness landscape being handled and each fitness function being optimized. The resulting algorithms seem to have promising performance in terms of final solution detected and convergence velocity. Extended numerical experiments have been carried out in each case in order to show the statistical significance of results.

Keywords: Memetic Algorithms, Multimeme Algorithms, Adaptive System, Fitness Diversity, Population Diversity, Computational Intelligence, Evolutionary Algorithms, Local Search, Noisy Fitness Landscapes, Human Immunodeficiency Virus, Image Processing, Peer to Peer, Electric Drives

Author

Dr. Ferrante Neri
Department of Mathematical
Information Technology,
University of Jyväskylä
Finland

Supervisors

Professor Raino A. E. Mäkinen
Department of Mathematical
Information Technology,
University of Jyväskylä
Finland

Dr. Jari Toivanen
Department of Mathematical
Information Technology,
University of Jyväskylä,
Finland

Reviewers

Prof. Pablo Moscato
School of Electrical Engineering and
Computer Science,
Faculty of Engineering and Built Environment
University of Newcastle,
Australia

Dr. William Rand
Northwestern Institute on Complex Systems,
Evanston, IL,
USA

Opponent

Dr. Jouni Lampinen
Department of Computer Science,
University of Vaasa,
Finland

ACKNOWLEDGEMENTS

I am deeply grateful to Prof. Raino Mäkinen for having believed in me and the constant and generous support he gave me during my doctoral studies at the University of Jyväskylä. I would like to give special thanks to Dr. Jari Toivanen for his invaluable suggestions and his scientific help.

I would like to give deepest thanks to Mr. Ville Tirronen for his precious scientific work and his indefatigable and constant cooperation and Prof. Tuomo Rossi for his generous scientific and human support.

I would like to express my sincere gratitude to Prof. Tommi Kärkkäinen and Dr. Kirsi Majava for their useful corrections and discussions.

I would also like to thank Dr. Yew-Soon Ong, Dr. Giuseppe Leonardo Cascella, Mr. Mikko Vapa, Mr. Niko Kotilainen, Dr. Nadia Salvatore and Prof. Silvio Stasi for having acted as co-author for some articles included in this thesis.

I would like to give special thanks to Ms. Anna Kononova for the frequent and useful discussions and the constant and inspiring presence.

I also thank my family for the patience they had and their daily encouragement.

Last but not least, I wish to give special thanks to my true friend Paolo Matelloni for his loyalty, faithfulness and constant presence during my stay in Finland.

"The reasonable man adapts himself to the world; the unreasonable one persists in trying to adapt the world to himself. Therefore all progress depends on the unreasonable man."

George Bernard Shaw

LIST OF FIGURES

LIST OF TABLES

CONTENTS

LIST OF INCLUDED ARTICLES

PI F. Neri, J. Toivanen, and R. Mäkinen, An Adaptive Evolutionary Algorithm with Intelligent Mutation Local Searchers for Designing Multidrug Therapies for HIV, *Applied Intelligence, Special Issue on Computational Intelligence in Medicine and Biology, Volume 27, Issue 3, pages 219-235, December 2007*

PII F. Neri, J. Toivanen, G. L. Cascella, and Y.-S. Ong, An Adaptive Multimeme Algorithm for Designing HIV Multidrug Therapies, *IEEE/ACM Transactions on Computational Biology and Bioinformatics, Special Issue on Computational Intelligence Approaches in Computational Biology and Bioinformatics, Volume 4, Issue 2, pages 264-278, April 2007*

PIII V. Tirronen, F. Neri, T. Kärkkäinen, K. Majava, and T. Rossi, A Memetic Differential Evolution in Filter Design for Defect Detection in Paper Production, *Applications of Evolutionary Computing, Lectures Notes in Computer Science, Volume 4448, pages 320-329, (EvoIASP Best Paper Nomination), April 2007*

PIV F. Neri, V. Tirronen, T. Kärkkäinen, and T. Rossi, Fitness Diversity Based Adaptation in Multimeme Algorithms: A Comparative Study, *Proceedings of the IEEE Congress on Evolutionary Computation, Special Session on Memetic Algorithms, Singapore, pages 2374-2381, September 2007*

PV V. Tirronen and F. Neri, A Fast Randomized Memetic Algorithm for Highly Multimodal Problems, *to appear on Evolutionary Methods in Design Optimization and Control, P. Neittaanmäki, J. Periaux, T. Tuovinen eds.*

PVI F. Neri, N. Kotilainen, and M. Vapa,, An Adaptive Global-Local Memetic Algorithm to Discover Resources in P2P Networks, *Applications of Evolutionary Computing, Lectures Notes in Computer Science, Volume 4448, pages 61-70, (EvoCOMNET Best Paper Nomination), April 2007*

PVII F. Neri, G. L. Cascella, N. Salvatore, and S. Stasi, An Adaptive Prudent-Daring Evolutionary Algorithm for Noise Handling in On-line PMSM Drive Design, *Proceedings of the IEEE Congress on Evolutionary Computation, Special Session on Evolutionary Computation in Dynamic and Uncertain Environments, Singapore, pages 584-591, September 2007*

PVIII F. Neri and R. Mäkinen , Hierarchical Evolutionary Algorithms and Noise Compensation Via Adaptation, *Evolutionary Computation in Dy-*

namic and Uncertain Environments, S. Yang, Y-S. Ong, Y Jin eds., Studies in Computational Intelligence, pages 345-369, (book chapter), April 2007

1 INTRODUCTION

Memetic Algorithms (MAs) are a class of computational intelligence algorithms which combine global search features of an evolutionary framework with Local Search (LS) to improve the solutions. MAs were introduced in [1] and [2] which established a new metaphor, where the concept of gene transmission, widely used in evolutionary computing, is replaced by the concept of meme which is defined as the "unit of imitation in cultural transmission" [3]. The main idea behind a MA is that the solutions belonging to a population within an Evolutionary Algorithm (EA) framework are supposed not only to generate offspring but also to communicate with each other in order to transmit ideas useful for improving their quality. This concept is algorithmically translated by means of an hybridization of population-based algorithms with exact methods [2] and later systematically defined as an evolutionary framework which employs within its generation cycle some local searcher components (deterministic or stochastic) [4]. Thus, the solutions represent the individuals of an evolving population and the local search is a lifetime learning for the individuals [5]. In the fashion of the metaphor, the individuals go to school, study and improve their fitness; thus allowing them to perform better and better. In addition, the individuals communicate amongst themselves thus transmitting their achieved learning. Although there exist several kinds of hybridization, e.g. intelligent crossover, intelligent initial sampling [4] as will be shown later, in general a MA is composed of an evolutionary framework and a LS employed to improve a subset solution during the evolutionary process. If not one, but a set of LSs is employed, the algorithm is named Multimeme Algorithm. Figure 1 shows the pseudo-code of a general MA implementation.

```
generate initial population;
evaluate each candidate solutions;
while termination conditions
   select parents;
   recombination of the parents;
   improve offspring via local search;
   mutation on the resulting offspring;
   evaluate offspring;
   improve offspring via local search;
   survivor selection for the next generation;
end-while
```

FIGURE 1 General pseudo-code of a MA

1.1 Tailored Algorithmic Design

During the 20^{th} century the development of numerical analysis and computers led to the implementation of optimization algorithms which were supposed to allow the machines to take over the work normally done by humans. On the other hand, the strict requirements in the hypotheses of such methods usually did not match the engineering problems. Thus, in the 60s and 70s some algorithms that do not require a deep a priori knowledge of the optimization problems were developed (e.g. Rosenbrock [6] or Hooke-Jeeves [7]). Nevertheless, due to their inner structure, these algorithms make an implicit use of information about the gradient (perform the so called "hill-climb") and thus present the drawback of being inefficient for solving global optimization problems and thus handling the multi-modalities.

During those years some algorithms inspired by nature i.e. Evolutionary Programming [8], Evolution Strategy [9], [10], and Genetic Algorithms [11] were designed. These methods, under the umbrella name of Evolutionary Algorithms (EAs), were making use of an analogy with evolution in order to perform the global search of an optimum. Since no implicit information about the gradient is used in EAs and no specific hypotheses are required EAs have been widely used in engineering problems during the 80s and 90s. The capability of EAs to handle multi-modalities and since they often outperform classical methods and hill-climbers allowed a massive diffusion of such algorithms amongst scientists and practitioners leading to the common sense notion that EAs are "universal optimizers" and thus better than the other algorithms.

The idea that EAs are better than other algorithms and that in general

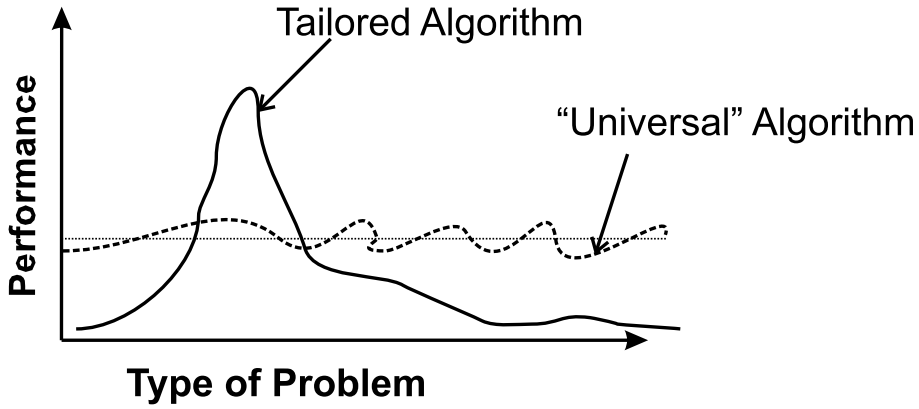

FIGURE 2 Graphical representation of the No Free Lunch Theorem

an algorithm A might be better than another algorithm B entered into crisis when a probabilistic approach was employed in order to compare the performance of two generic algorithms. The theoretical conclusion of such study is summarized in the No Free Lunch Theorem theory [12].

1^{st} **No Free Lunch Theorem:** For a given pair of algorithms A and B

$$\sum_{f} P\left(x_m | f, A\right) = \sum_{f} P\left(x_m | f, B\right), \tag{1}$$

where $P\left(x_m | f, A\right)$ is the probability that algorithm A detects the optimal solution x_m for a given objective function f (i.e. optimization problem). $P\left(x_m | f, B\right)$ is the analogue probability for algorithm B. This probability expresses the performance of the algorithm under analysis. In other words, the 1^{st} No Free Lunch Theorem states that the average performance, over all possible optimization problems, is the same for every arbitrary pair of algorithms. Thus, there does not exist either an optimal optimizer or a universal optimizer.

For the sake of completeness, the 2^{nd} No Free Lunch extends the result to time variant objective functions and states that if one algorithm outperforms another for certain kinds of cost function dynamics, then the reverse must be true on the set of all other cost function dynamics.

Figure 2 shows a graphical representation of the No Free Lunch Theorem. More specifically, the dashed line represents the behavior of an algorithm which has mediocre performance for a wide spectrum of problems while the solid line has very good performance in a limited amount of cases and quite poor performance otherwise. A graphical representation of the No Free Lunch Theorem shows that the areas below the two lines are the same.

Usually classical EAs (e.g. GA) have a performance of the kind shown by the dashed line, meaning that they are likely able to lead to some improvement

upon initial sampling but in real-world optimization problems (e.g. highly multi-modal, noisy etc., see [13]) they are often unable to detect a satisfactory solution.

Thus, the approach in this thesis considers each optimization problem as something whose features are unique and therefore must be analyzed beforehand; subsequently a tailored algorithm for the specific class of problems must be designed in order to have high performance for the specific application.

1.2 An Introduction to Local Search

A **Local Searcher** (LS) is an algorithm which applies an iterative process of examining the set of points in the neighborhood of a candidate solution, and replacing it with a better neighbor. The principal components that affect the workings of a LS are the following:

- The **explorative structure** of the local searcher is the logic, employed by the LS, of generating a new candidate solution to be analyzed. It can be **deterministic** or **stochastic**
- The **order** of the LS defines the required knowledge of the fitness function and it is restricted to continuous optimization problems. More specifically the LS is said to be **order zero** if it does not require the calculation of the derivatives, **order one** if it requires the calculation of the first order derivatives, **order two** if it requires the second order derivatives
- The **pivot rule** of a LS defines the criteria for accepting an improving point. A **steepest ascent** (descent) pivot rule terminates the exploratory loop only after the entire neighborhood has been searched, A **greedy** pivot rule terminates the exploratory loop as soon as an improvement is found.
- The **depth** of the local searcher defines its termination condition. It could stop at the first improvement (first iteration), only when no better solution can be found or in an intermediate situation (e.g. a few improvements)
- The **neighborhood generating function** defines the set of points that can be reached from the current best solution by application of the move operators.

Fig 3 shows the pseudocode of a general LS.

```
iteration=0;
while condition on the depth
   while condition on the pivot rule
      generate a neighbor;
      evaluate the neighbor and compare it with the current best;
   end-while
   update the current best;
   iteration=iteration+1;
end-while
```

FIGURE 3 LS pseudocode

1.3 Baldwinianism vs. Lamarckianism

When a local search is applied and possibly an improvement upon the starting point is achieved, the subsequent problem is how to use the new solution within the evolutionary system and in which way the novel genotypic information should perturb the system.

In scientific literature two different philosophies have been proposed regarding lifetime learning:

- **Lamarckian**: the improvement acquired by an individual during its lifetime causes a genotypic mutation and thus can be transmitted to its offspring

- **Baldwinian**: the improvement acquired by an individual does not affect its genotype but only its fitness value and thus cannot be transmitted to its offspring

It should be mentioned that although the Lamarckian philosophy seems not to have any validity in human genetics, its metaphorical employment in MAs can turn out to be very successful. In fact, the Baldwinian philosophy proposes that application of the local searcher must be an instrument suggesting promising direction search but the actual improvement of the genotypes must occur only by means of the variation operator. More specifically, when a local searcher is applied, its fitness is replaced with the fitness after the learning while the genotypic information (the chromosome) is not modified. Thus, during the selection process a chromosome will survive when its fitness after the learning process is better. If this chromosome can survive for a sufficient number of generations, then it will be possible to evolve by genetic operations, into

the actual chromosome corresponding to the fitness after the learning. It is shown in [14] that the Baldwinian learning can efficiently direct the genotypic changes. It is obvious that the Lamarckian approach is definitely more aggressive than the Baldwinian one. Better performance of one or the other basically depends, in accordance with the No Free Lunch Theorem, on the nature of the problem and the kind of hybridization implemented. For example, if only a small number of solutions undergo lifetime learning a Lamarckian approach can be preferable and if the fitness landscape contains very wide suboptimal basins of attraction a Baldwinian approach is probably more efficient.

1.4 Intelligent Operators

A simple way to design a MA is to generate hybrid components by hybridizing the standard operators of an EA with a LS.

A first example of this class of MAs is the intelligent initialization. The usual initialization of an EA is carried out by performing a pseudo-random sampling with uniform distribution function within the decision space. For some problems, e.g. when the global optimal basin of attraction is narrow and rather small, it may turn out beneficial in terms of convergence velocity to perform a sampling which is as spread out as possible. This is done in order to cover the maximum possible width of the decision space. In such cases it might be wise to execute the initial sampling by using a deterministic technique (e.g. Latin Hypercube [15], [16]). Moreover, in highly multivariate cases an initial sampling which explores the entire decision space is not applicable due to an excessive computational effort. Thus, an analysis on the influence of each design variable on the fitness values can be beneficial in detecting the subset of variables which dramatically influence the fitness behavior of mediocre solutions (initial sampling solutions) and therefore deserve application of a deterministic process at the beginning of the algorithm.

In other cases, it might be known that a fairly performing suboptimal solution exists (e.g. in control problems [17]) or it might be possible to detect it by preliminarily applying a heuristic. In these conditions, the goal of the initial sampling might be to massively sample the neighborhood of the suboptimal solution but still have a small portion of the population spread out elsewhere. For such a problem a randomized sampling by means of a non-uniform distribution could be applied.

A second example of local search integration within standard evolutionary operators is the implementation of intelligent variation operators i.e. intelligent crossover or intelligent mutation. In [18] a crossover hill-climber is

designed. This operator, according to a steepest descent pivot rule, explores all possible offspring before accepting those that perform best. In [19] a tailored intelligent crossover for protein structure prediction has been designed making use of knowledge of the physics of the problem. In paper [20] a distance-preserving crossover has been introduced with application to the Travelling Salesman Problem (TSP). This crossover imposes that the offspring edges are inherited from both parents (and not only one parent) and applies a nearest-neighbor heuristic to combine the chromosome sections in the most convenient way. In [21] several implementations of MAs applying greedy local search in the initialization and crossovers are extensively analyzed with reference to the application to the TSP.

1.5 Lifetime Learning, Memetic and Multimeme Algorithms

The most common way to perform hybridization is with so called lifetime learning which is the application of the local search to individuals of the population during the evolution. Plenty of different solutions have been proposed in the literature about the way in which the hybridization can be executed. In some cases it is proposed to apply the LS to the best individual, in other cases the LS is applied to the worst or to a subpopulation; in some cases the local search is applied many times with a limited depth while in other cases it is applied a few times with a high depth value.

Although a proper choice strictly depends on the employed algorithmic components and the problem under study (see [2], [18] and, [22]), a general guideline has been given in [23] where it is proven that a local search whose move operator is not the same as those in evolutionary recombination and mutation is beneficial at least in reducing worst-case run times. In MAs, one crucial concept is that algorithmic components (both evolutionary and local search), having different features and natures, should explore the decision space from different perspectives and interact with each other in a competitive and cooperative logic (see [2] and [1]). This concept leads to the coordinated employment of different LSs within the same evolutionary framework as proposed in [24] which proposes different LSs for the TSP and [25] where several types of Tabu Search are used by each agent. In [26] the distinction between MAs and Multimeme Algorithms (MmAs) is systematically given, meaning that the former uses only one (usually complex) local search while the latter employs a set of (usually simple) local searchers. In both cases, for MAs and MmAs, LSs have to interact with evolutionary components and thus need to be efficiently integrated within the system.

1.6 Measurement of the Diversity

As previously stated, when a MA or a MmA is designed, the problem regarding how the hybridization can efficiently be performed arises i.e. how the evolutionary framework can intelligently execute the coordination of local searchers. Since the first implementations of MAs (see [2] and [25]), the idea that coordination of the algorithmic components requires an adaptive scheme was proposed. In addition, in [2], it was mentioned that the cooperation and competition within a MA could be naturally coordinated by means of a diversity measurement. This concept has been revisited in [24] and the term "diversity crisis" in the context of adaptation has been introduced. The necessity of controlling the diversity amongst individuals of the population is due to the fact that MAs might be subject to diversity loss since the application of LSs could lead to focusing on a restricted number of good solutions. Moreover, if the LS is executed with a high depth (up to the local optimum), it is fundamental that new basins of attraction are constantly detected; if the LS is executed with a low depth, the MA search could become focused only on the basin of attraction of a few solutions. In recent years, high demand of sophisticated engineering simulators required the application of algorithms able to handle highly multivariate and highly multi-modal fitness landscapes, therefore the necessity of designing algorithms able to avoid stagnation and premature convergence, notwithstanding a limited population size (with respect to the dimensionality of the problem) became fundamental. Therefore, several adaptation schemes aiming at controlling and preserving the population diversity have been designed. In [4] and [27] it is shown how a simulated annealing-like adaptive scheme can help in maintaining the population diversity. In [28] a similar logic but based on the entropy variation has been proposed for adaptation. In [29] the diversity is implicitly controlled by the application of LSs coordinated by a probabilistic criterion. In [30] a memory based approach is used for tracking the operator moves and preserving the diversity. In [31] two diversity-based probabilistic criteria for determining the frequency of local search activation are proposed for parallel MAs.

1.7 Contents of the Thesis

This thesis, in accordance with the No Free Lunch Theorem, proposes tailored MAs for solving some specific classes of problems. The general algorithmic philosophy employs the idea of cooperation and competition amongst algo-

rithmic components and proposes novel adaptive schemes for executing coordination of LSs and parameter setting. The adaptive schemes are based on the design of several indexes that measure fitness diversity and their use within the evolutionary framework.

More specifically, the fitness diversity is used as an index to monitor the algorithmic process and then intelligently apply and coordinate the local search components and execute an adaptive parameter setting. As a general guideline over the papers included in this thesis, when the diversity is high, the evolutionary framework is supposed to exploit the available genotype by searching in the promising directions. On the contrary, when the diversity is low the framework can efficiently be assisted by local search components in order to offer alternative perspective to the search of the optimum. In addition, LSs with different features are used in dependance of the diversity level. If the diversity is moderately low, rather explorative LSs are employed in order to detect new promising search directions; if the diversity is extremely low, highly exploitative LSs are employed in order to quickly detect the optimum of the corresponding basin of attraction and thus end the game.

The diversity measurement is also employed in the case of noisy environment in order to perform the coordination of the averaging components.

Eight original articles are included in the thesis. In Article **PI**, a MmA has been applied for designing the optimal Human Immunodeficiency Virus (HIV) therapy. The algorithm executes the coordination of the LSs and parameter setting by means of an index, namely ξ, which measures the fitness diversity amongst individuals of the population by computing the difference between best and average fitness normalized to the best fitness.

In Article **PII**, for a different HIV model another kind of MmA has been implemented employing another kind of measurement of fitness diversity, namely ψ. The index ψ has been designed taking into account the specific features of the fitness landscape being optimized.

In Article **PIII**, a MmA employing a Differential Evolution framework and local searchers has been proposed for an image processing application and a third kind of measurement of fitness diversity, namely ν, has been employed.

In Article **PIV**, a comparative analysis of the three adaptive schemes from articles **PI**, **PII** and **PIII** is performed by integrating the adaptive rules within the Fast Adaptive Memetic Algorithm (FAMA) proposed in [17] and applying the three variants of FAMA to a large set of different test functions. In Article **PV**, the results shown in Article **PIV** are extended by adding the comparison with a FAMA employing a pseudo-random number instead of the diversity index.

In Article **PVI**, another MmA employing a fitness diversity adaptation

based on the measurement proposed in Article **PII** has been applied for the neural network training with reference to a telecommunication problem. The noise in the fitness landscape has been considered in the algorithmic design by an online averaging of the results.

The problem of the optimization in presence of uncertainties has been analyzed in depth in Article **PVII** and Article **PVIII** for the Gaussian noise with reference to a control engineering problem and for a non-Gaussian algorithmic noise with reference to structural optimization, respectively. The noise handling has been performed by adaptive implicit and explicit averaging of the fitness values and by the aid of two cooperative/competitive survivor selection schemes adaptively coordinated by means of the diversity index proposed in Article **PI**.

1.7.1 Contribution of the Author in Joint Publications

The author contributed by designing each algorithm, experimental setup and statistical test presented in the publications. The joint publications have been done in cooperation with experts of either the application problem or the mathematical modelling. In addition the author significantly contributed to the writing and organization of all the papers.

1.8 Further Developments

The study on fitness diversity adaptation in MAs has recently been extended by combining the concept of fitness diversity measurement with a probabilistic scheme for coordinating the LSs. In particular, Chapter 2 proposes an enhancement of the algorithm in Article **PIII** for filter design for defect detection in paper production while Chapter 3 proposes a novel MmA which employ an adaptation based on the measurement of the super-fit performance with respect to performance of the other individuals. Two applications are shown: the first is for design of a control system for a Direct Current Motor, the second is the same filter design as in Chapter 2 and in Article **PIII**. Both the algorithms proposed in the following sections employ a Differential Evolution framework and a set of local searchers. This combination seems to be very promising in many applications.

2 AN ENHANCED MEMETIC DIFFERENTIAL EVOLUTION IN FILTER DESIGN FOR DEFECT DETECTION IN PAPER PRODUCTION

In recent years, machine vision systems for quality inspection and fault detection have become standard in the paper industry. These systems monitor the paper web for structural defects such as holes and for defects in quality such as faint streaks, thin spots, and wrinkles [32]. Detection of weak defects is crucial in quality paper production since their presence in paper sheets raises difficulties in, for example printing, thus leading to significant monetary losses. The paper web inspection is a challenging task since it must be executed under strict real time constraints. In fact, paper machines can achieve a speed of over 25 m/s, while the defects are barely a few millimeters in size.

Defect detection is in the spectrum of low-level vision since it is related to both edge detection and textural analysis and can be seen as a pre-segmentation technique for identifying defects and insulate them from the background.

In order to solve this class of problems several solutions have been proposed over the years in the literature and industrial applications. Classical edge detection methods and simple segmentation schemes, such as plain threshold have been widely studied in the literature [33] and have become the toolbox of machine vision engineering. These methods, in spite of their popularity with the industry, are often fragile and need to be tuned by human experts for each new condition.

Thus, Computational Intelligence (CI) approaches have been successfully applied in low-level vision during recent years. For example, such approaches aim at executing machine learning by means of neural networks [34], [35] or employing evolutionary algorithms to derive high performance operators for specific applications [36], [37], [38], [39], [40]. The latter approach is very

promising due to the fact that these operators are potentially adaptable in different situations without human expert decision making [41].

This section aims to study the defect detection in paper production by means of image processing techniques. In this application, the high real time requirements pose a serious limitation to applicable techniques. In addition, the defects studied in this application are rather faint and masked by the natural structure of the paper which varies by time and process state. Moreover, compared to other fields of defect detection, we lack regular texture or estimable background, such as is encountered with, for example, textiles [42].

Defect detection in paper production has been studied over the years and several solutions have been proposed in industries and the literature. The most commonly used approaches in industrial applications are based on simple threshold techniques, [43]. Since these techniques are inadequate for weak defects, more sophisticated methods are required. A popular approach is to utilize texture-based techniques. In [44] the defects are characterized as deviations from the background texture, the problem is encoded as a two class classification problem by means of local binary patterns as source of features and a Self Organizing Map (SOM) as a clustering/classification tool. However, wrinkles and streaks are often quite faint perturbations of texture, and can be easily missed with texture analysis.

This work proposes an approach which employs Finite Impulse Response (FIR) filters, parameterized by a Gabor function, specifically optimized for the task of weak defect detection. The problem oriented design of the FIR filters seems very promising for similar image processing problems with other fields of application, for example in [45] for texture segmentation and in [46] for a vehicle tracking problem. Paper [46] also proposed a hybrid evolutionary algorithm in order to perform the filter design. FIR filters are flexible and have relatively efficient machine implementations which makes them applicable for the task. Gabor functions are useful for filter formulation in cutting down the number of parameters and are presumed applicable due to similarities with mammalian vision cortex responses. One important difficulty related to this approach is that the problem oriented design of a FIR filter requires the solution of an often challenging optimization problem characterized by a multivariate fitness function.

2.1 Features of the Filter and Problem Formulation

Paper inspection is realized using transillumination. The images acquired in this way contain noisy characteristics as the whole paper structure is imaged.

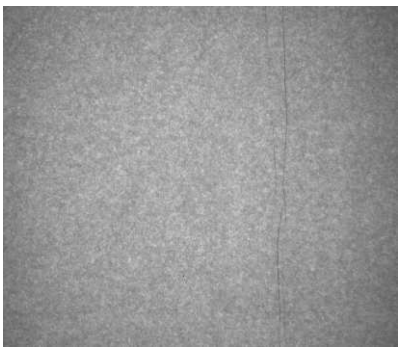

FIGURE 4 Source image

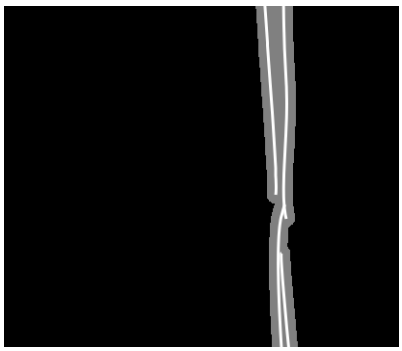

FIGURE 5 Label image

Images may vary by a different background noise field (paper formation), illumination and defect shape. Training of the filter is based on a set of source images supplied by Viconsys Oy [1] and a corresponding set of label images. These two sets constitute the training set. Images are taken in a static situation with a resolution of approximately 0.6 mm/pixel. Figure 4 shows an image belonging to the training set with the corresponding label image Figure 5.

The employment of Gabor Filters [47] has been proposed since they turned out to be very successful in similar applications of machine vision for defect detection [48] [49] and edge detection [50]. In particular, the following Gabor function [47] is proposed:

$$\begin{aligned} Gb[\theta,\psi,\sigma_x,\sigma_y,\lambda](x,y) = exp\left(\frac{(x\cos\theta+y\sin\theta)^2}{2\sigma_x^2}\right) \\ exp\left(\frac{(-x\sin\theta+y\cos\theta)^2}{2\sigma_y^2}\right)\cos\left(\frac{2\pi}{\lambda}+\psi\right), \end{aligned} \tag{2}$$

where θ is the angle perpendicular to parallel stripes of the filter, ψ is the phase offset (the filter is symmetrical when $\psi = 0$ and antisymmetrical when $\psi = \frac{\pi}{2}$). Furthermore, σ_x, σ_y specify both the size of the filter and it's ellipticity and λ is the wavelength of the filter. In other words we use a directed bandpass filter with bandwidth determined by λ and the ratio σ_x/σ_y.

Gabor filters are limited by their relatively high computational cost. The real time requirements of our application rule out the use of Fast Fourier Transform (FFT) and computation of filters via convolution theorem. To satisfy real time requirements in spatial domain, we sample the Gabor function into a 7×7 discrete kernel. This will limit the available scales and frequencies. However since the problem is empirical, we allow the filter to be truncated to kernel size, trading accuracy to overcome aforementioned limits [51].

[1] `http://www.viconsys.fi/`

TABLE 1 Design parameters

parameter	description	range of variability
$\alpha(1), \alpha(7)$	weight	$[-100, 100]$
$\alpha(2), \alpha(8)$	θ	$[0, 2\pi]$
$\alpha(3), \alpha(9)$	ψ	$[0, 2\pi]$
$\alpha(4), \alpha(10)$	σ_x	$[0, 20]$
$\alpha(5), \alpha(11)$	σ_y	$[0, 20]$
$\alpha(6), \alpha(12)$	λ	$[0, 20]$

Since the defects tend to contain large variations in angle and the filter is direction dependent, the outputs of two different filters are combined. The coordination of the filters is carried out by assigning a weight coefficient to each filter and then selecting that which produces maximal intensity for a given pixel. A post-filtering stage, by means of a gaussian filter, is included in order to mitigate the effect of spurious responses due to noise [52]. Thus, for a given image I, α indicating the vector of 12 elements representing the design parameters of the two filters, the filter formula is given by:

$$F(\alpha, I) = G_{15,15} \star max \begin{array}{l} [(\alpha(1)\, Gb(\alpha(2), ...\alpha(6)) \star I), \\ (\alpha(7)\, Gb(\alpha(8), ...\alpha(12)) \star I)] \end{array} \quad (3)$$

where $\alpha(1)$ and $\alpha(7)$ are the weights for both filters, $\star$ denotes the two dimensional discrete convolution and $G_{r,t}$ represents an ordinary gaussian kernel of size $r \times t$. The design parameters of the Gabor filters are shown in Table 1.

The problem of the filter design thus consists of finding a proper set of design parameters α. In order to estimate the fitness of a candidate solution α, the following procedure is carried out.

Let us indicate with S the source image and with L the corresponding label image. The filtered images F are divided into three regions based on the label images: the defect region D defined as the set of those pixels (x, y) of the image F such that $L(x, y) = 1$, the clean region C defined as the set of those pixels (x, y) such that $L(x, y) = 0$, and the non-interesting region characterized by other colors (grey in Figure 5). Then let us define the similarity function sim as follows:

$$sim(S, L) = a\sigma(D) + b\sigma(C) + c\sqrt{|\mu(D) - \mu(C)|}, \quad (4)$$

where μ and σ denote mean and standard deviation over the set of pixels and a, b, c are weight coefficients. The first term in (4) measures uniformity of the

defect region, the second term measures the noise in background (C region) and the third term measures separation of the two regions. The weight coefficients have been set as $a = 1, b = 2$ and $c = -3$ taking into consideration that it is highly desirable that the filter clearly separates defects from the background, it is also important that noise in the background does not lead to detection of false defects. Finally, the fitness function f is given by:

$$f(\alpha) = \frac{1}{n_I} \sum_{k=1}^{n_I} sim\left(F(\alpha, S_k), L_k\right), \tag{5}$$

where n_I is total number of images in the training set, S_k and L_k are respectively the k^{th} source and label image from the training set. The filter design is thus stated as the minimization of f over $H = [-100, 100]^2 \times [0, 2\pi]^4 \times [0, 20]^6$. For the sake of clarity, it can be highlighted that the optimization problem under examination is defined in a multi-dimensional continuous space characterized by 12 dimensions where some of them characterize some the real-valued parameters and the others characterize the angular parameters.

2.2 Enhanced Memetic Differential Evolution

In order to minimize the fitness f shown in (5) an EMDE is proposed here. The description of algorithmic components constituting the EMDE and their operation is given in the following.

2.2.1 Differential Evolution Framework

An initial sampling of $S_{pop} = 100$ individuals is executed pseudo-randomly with a uniform distribution function over the decision space H. At each generation, for S_{pop} times, four individuals α_1, α_2, α_3 and α_4 are extracted from the population pseudo-randomly. Recombination according to the logic of a DE occurs at first by generating α'_{off} according to the following formula [53], [54]:

$$\alpha'_{off} = \alpha_1 + K(\alpha_2 - \alpha_3), \tag{6}$$

where $K = 0.7$ is a constant value set according to the suggestions given in [53]. Then, in order to increase the exploration of this operator, a casuality is introduced by switching some design parameters of α'_{off} with the corresponding genes of α_4. Each switch occurs with a uniform mutation rate $p_m = 0.3$, as suggested in [55], and the offspring α_{off} is thus generated. The fitness value of α_{off} is calculated and, according to a steady-state strategy, if α_{off} outperforms α_4, it replaces α_4, if on the contrary $f(\alpha_{off}) > f(\alpha_4)$, no replacement occurs.

2.2.2 Local Searchers

The EMDE employs the following three local searchers which assist the evolutionary framework (DE) by offering alternative exploratory perspectives.

The Hooke Jeeves Algorithm

The HJA [7], [56] initializes the exploratory radius h_{HJA-0}, an initial candidate solution α and a 12×12 direction exploratory matrix $U = diag(w(1), w(2), ..., w(12))$, where $w(m)$ is the width of the range of variability of the m^{th} variable. Let us indicate with $U(m,:)$ the m^{th} row of the direction matrix $m = 1, 2, ..., 12$.

The HJA consists of an exploratory move and a pattern move. Indicating with α the current best candidate solution and with h_{HJA} the generic radius of the search, the HJA during the exploratory move samples solutions $\alpha(m) + h_{HJA}U(m,:)$ (″+″ move) with $m = 1, 2, ..., 12$ and the solutions $\alpha(m) - h_{HJA}U(m,:)$ (″-″ move) with $m = 1, 2, ..., 12$ only along those directions which turned out unsuccessful during the ″+″ move. If a new current best is found α is then updated and the pattern move is executed. If a new current best is not found, h_{HJA} is halved and the exploration is repeated.

The HJA pattern move is an aggressive attempt of the algorithm to exploit promising search directions. Rather than centering the following exploration at the most promising explored candidate solution (α), the HJA tries to move further [57]. The algorithm centers the subsequent exploratory move at $\alpha \pm h_{HJA}U(m,:)$ (″+″ or ″-″ on the basis of the best direction). If this second exploratory move does not outperform $f(\alpha)$ (the exploratory move fails), then an exploratory move with α as the center is performed. The HJA stops when the budget condition of 500 fitness evaluations is reached. The initial exploratory radius h_{HJA-0} is adaptively set as later explained. For the sake of clarity, the pseudo-code of the HJA for a given current best solution α is shown in Figure 6.

The Stochastic Local Searcher

The SLS [58] picks up a solution α and initializes a parameter σ_{SLS-0}. Then, 24 ($m = 1, 2, ..., 24$) perturbation vectors $h_{SLS}(m)$ are generated; these vectors having the same length of α and each gene being a random number of a normal distribution having mean value in $\alpha(m)$ and standard deviation $\sigma_{SLS}w(m)$. Number of perturbations is chosen so that SLS and HJA have similar computational costs. For each of these perturbation vectors, $\alpha + h_{SLS}$ is calculated and the related fitness value is saved. If the most successful perturbation has a

while budget condition
EXPLORATORY MOVE
  for $m = 1 : 12$
    compute and save $f(\alpha + h_{HJA}U(m,:))$;
    if $f(\alpha + h_{HJA}U(m,:)) > f(\alpha)$
      compute and save $f(\alpha - h_{HJA}U(m,:))$;
    end-if
  end-for
PATTERN MOVE
  detect the best performing candidate solution among those explored;
  if the best performing is α
    halve h_{HJA};
  else
    $\alpha =$ best performing candidate solution over those explored;
    calculate $f(\alpha \pm h_{HJA}U(m,:))$;
    if $f(\alpha \pm h_{HJA}U(m,:)) > f(\alpha)$
      $\alpha = \alpha \pm h_{HJA}U(m,:)$;
    end-if
  end-if
end-while

FIGURE 6 HJA pseudo-code

while budget condition
  for $m = 1 : 24$
    generate perturbation vector $h_{SLS}(m) \sim N(\alpha, \sigma_{SLS}w(m))$;
    calculate and save $\alpha + h_{SLS}(m)$ and $f(\alpha + h_{SLS}(m))$;
  end-for
  pick up the best performing perturbed solution $\alpha + h_{SLS}$;
  if $f(\alpha + h_{SLS}) \leq f(\alpha)$;
    $\alpha = \alpha + h_{SLS}$;
  else
    $\sigma_{SLS} = \frac{\sigma_{SLS}}{2}$;
  end-if
end-while

FIGURE 7 SLS pseudo-code

```
while budget conditions
  Perturb the current best solution thus generating α_per;
  Calculate the fitness value of the perturbed solution;
  if f(α_per) < f(α)
    Accept the perturbed solution as a new current best solution (α = α_per);
  else
    Calculate the probability p = e^((f(α)−f(α_per))/Temp);
    Generate a pseudo-random value u ∈ [0,1];
    if u < p
      Accept the perturbed solution as a new current best solution;
    else
      Keep α as the current best solution;
    end-if
  end-if
  Reduce Temp;
end-while
```

FIGURE 8 Simulated Annealing pseudo-code

better performance than the starting solution, the perturbed solution replaces the starting one, otherwise σ_{SLS} is halved and the process is repeated. The algorithm stops when the budget on the number of fitness evaluations (500) is exceeded. The setting of σ_{SLS-0} is adaptively performed as it will be later described in adaptation subsection. Figure 7 shows the pseudo-code of the SLS.

Simulated Annealing

The SA metaheuristic [59], [60] offers a third exploratory perspective in the decision space which can choose a search direction leading to a basin of attraction different from that where starting candidate solution α_0 is. The exploration is performed by using the same perturbation logic as was described in the SLS. This local searcher, as well as the others, is run each time for 500 fitness evaluations (budget condition). Initial temperature $Temp_0$ is adaptively set following the necessity of the evolutionary process (see below in adaptation subsection). The temperature $Temp$ is reduced according to a hyperbolic law following the suggestions in [61]. For sake of clarity, the pseudo-code of the SA for a given current best solution α is shown in Figure 8.

2.2.3 Functioning of the Local Searchers

As highlighted in [26] the crucial problems when a Memetic (or Multimeme) Algorithm is designed are the choice of *which* local searchers are employed and *how* the hybridization with an evolutionary framework is efficiently performed. This section aims to address these two topics explaining reasons behind the choice of these specific local searchers and thus propose the integration within the Differential Evolution Framework.

In [54] it is shown that that the DE, despite its simplicity, can be very efficient in many applications and can outperform classical metaheuristics (e.g. Genetic Algorithms and Evolution Strategies). For the present application, [62] and Article **PIII** prove the effectiveness of DE. On the other hand, it is known from literature [55] that the DE is subject to stagnation problems due to its highly explorative features.

In order to enhance the DE algorithmic performance for the class of problems under study, we aim to propose an adaptive Multimeme Algorithm [63], [64] which hybridizes the potentials of the DE with a list of local searchers which are supposed to assist the evolutionary framework. This list of local searchers is composed of algorithmic components having various features in terms of exploration logic and pivot rule and offer alternative perspectives [26] for exploring the decision space and handle the fitness landscape. In addition, the local searchers, integrated within the DE framework, are supposed to compete and cooperate [2] in the spirit of Meta-Lamarckian learning [65].

The HJA is a fully deterministic local searcher characterized by a steepest descent pivot rule [4] with high exploitative features. Thus, it is clear that the HJA can be employed for executing the hill-descent of a promising basin of attraction and is less likely to "jump out" from it as remarked in [17].

The SLS also has very exploitative features since it attempts to improve a given solution in the neighborhood of a given solution and is also characterized by a steepest descent pivot rule since it chooses, at each step, the best solution only after having calculated the fitness value of 24 potential candidates. On the other hand, the SLS differs from the HJA in the neighborhood generating function [4] since the SLS, unlike the HJA, has a stochastic structure for generating new candidate solutions. In this sense the SLS, notwithstanding its mainly local features, contains a mild explorative logic.

The well-known structure of the SA allows, with a certain probability which decreases over time, to accept less satisfactory solutions in order to eventually improve the starting candidate solution [59], [60]. This feature makes the SA rather explorative and allows the algorithm to significantly improve bad performing solutions. On the other hand, its application to a solution which is already performing good can lead to the loss of a promising

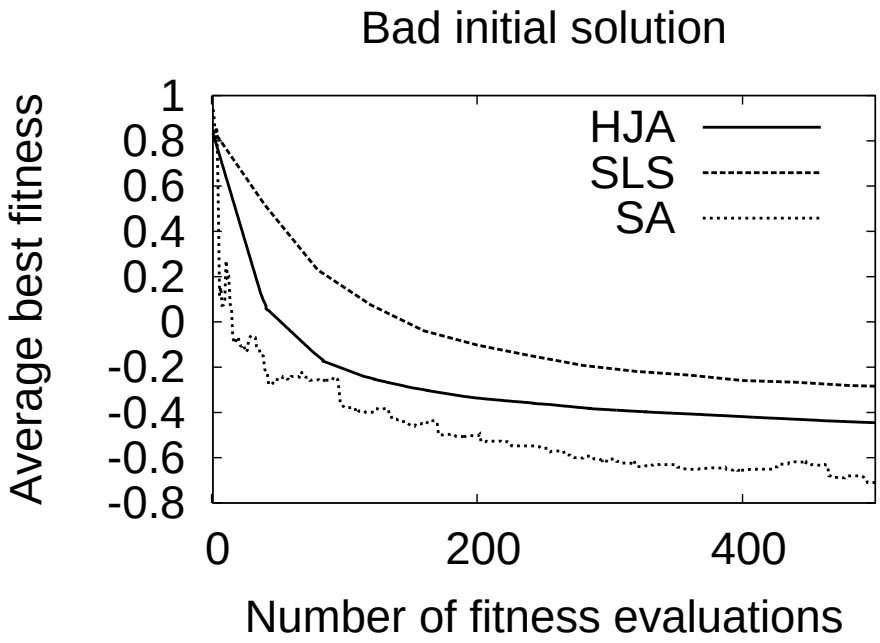

FIGURE 9 Local searcher performance for bad performing initial candidate solutions

genotype and in the worst case of a promising basin of attraction.

In order to better understand the functioning of the three local searchers for the problem under study and then have some hints about an efficient hybridization the following test has been designed. On the basis of the study carried out in Article **PIII**, 10 candidate solutions having low performance, 10 having mediocre performance and 10 having good performance have been sampled. For each set of 10 solutions, we attempted to improve the genotypes by applying the HJA, SLS and SA for 500 fitness evaluation. For each local searcher, the performance has been averaged (Average Best Fitness) over the 10 experiments carried out.

Figure 9 shows the performance of the local searcher for bad performing initial solutions. It can be noted that for a bad performing initial solution, the SA outperforms both the SLS and the HJA and leads to a significant and quick enhancement in the fitness value. Moreover, the comparison between SLS and HJA shows that the HJA outperforms the SLS. According to our interpretation, this result is due to the fact that the HJA performs an efficient local optimization in a not so promising basin of attraction while the SLS, due to the combination of stochastic structure and local features, improves the solution

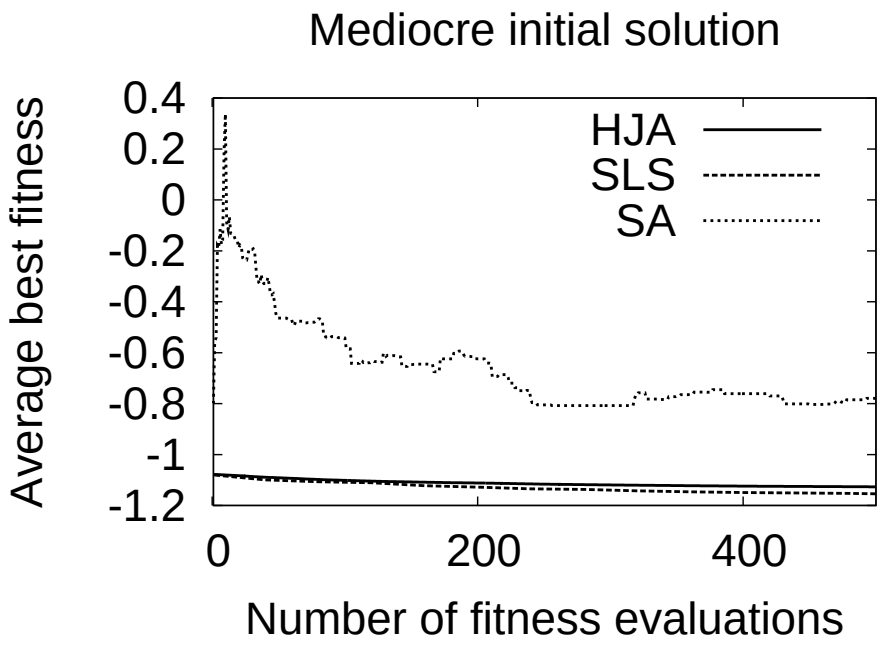

FIGURE 10 Local searcher performance for mediocre performing initial candidate solutions

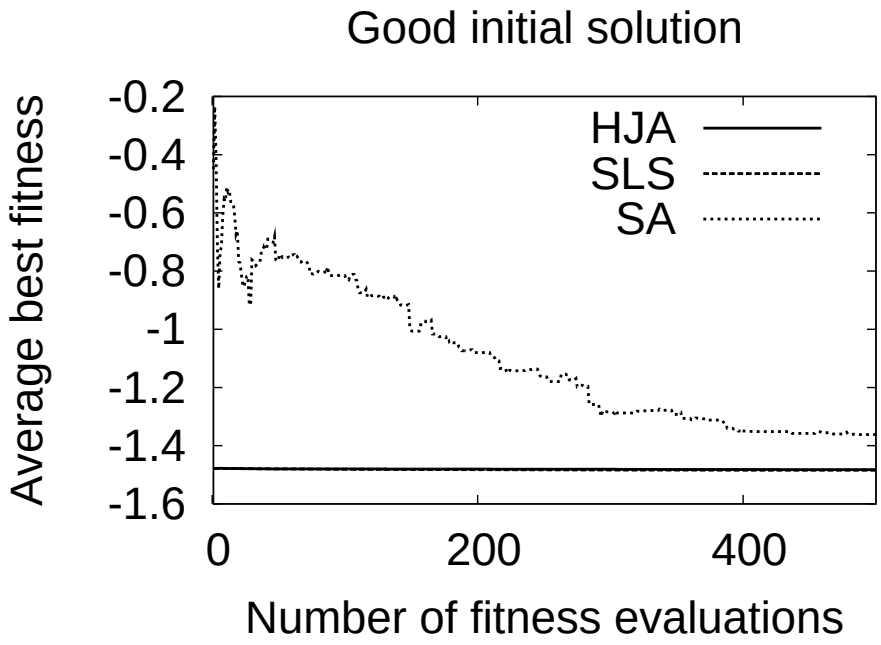

FIGURE 11 Local searcher performance for good performing initial candidate solutions

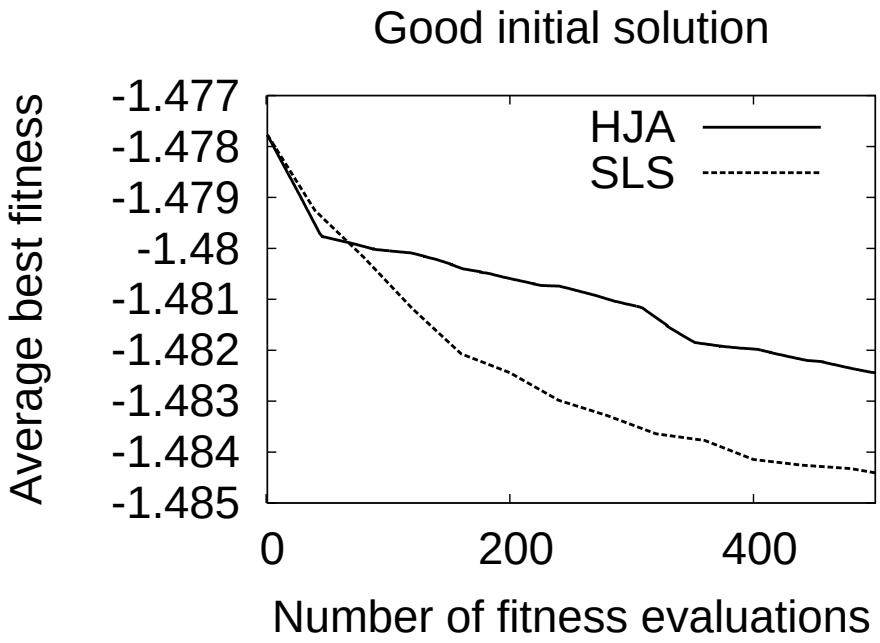

FIGURE 12 HJA vs SLS for good performing initial candidate solutions (zoom detail)

within the basin of attraction without succeeding in the hill-descent.

Figure 10 shows the performance of the local searcher for mediocre performing initial solutions. For a mediocre initial solution, it is clear that SLS and HJA clearly outperform the SA and the SLS detects better solutions than the HJA. Results in Figure 10 thus show that application of SA can be very beneficial for poorly performing initial solutions but not for solutions which have a rather good performance.

Figure 11 shows the performance of the local searcher for good performing initial solutions. In this case, it is clear that the SA tends to lose a good genotype and after does not manage to find a new solution having better performance than the initial one. On the contrary, as shown in the zoom detail in Figure 12, the SLS and HJA perform well in further enhancing a solution which already has a very low fitness value, see Article **PIII**. SLS and HJA have rather similar performance and converge to a similar solution but once again, SLS seems to be more promising than the HJA. It is interesting to consider that HJA, for all the three sets of experiments carried out, has intermediate performance with respect to the other two local searchers. For bad initial solutions it performs better than the SLS but worse than the SA whilst for good initial solutions worse than the SLS but better than the SA.

2.2.4 Adaptive Coordination of the Local Searchers

In order to perform coordination of the local searchers, an adaptive functioning is proposed. Every 1000 DE fitness evaluations the following index is calculated (see Article **PIII**):

$$\nu = \min\left\{1, \frac{\sigma_f}{|f_{avg}|}\right\}, \tag{7}$$

where $|f_{avg}|$ and σ_f are respectively the average value and standard deviation over the fitness values of individuals of the population. The parameter ν can vary between 0 and 1 and can be seen as a measurement of the fitness diversity and distribution of the fitness values within the population [66], [17]. More specifically, if $\nu \approx 0$, the fitness values are similar amongst each other, on the contrary if $\nu \approx 1$, the fitness values are different amongst each other and some individuals thus perform much better than the others (see also Articles **PI**, **PII** and **PVIII**). Moreover, Article **PIII** shows that in early generations the value of ν is high and that over the generations it decreases. In other words, for a high value of ν, the population likely contains bad initial solutions for the application of local searchers and for a low value of ν, the population likely contains good initial solutions for the application of local searchers (see Figure 13 for the case of a plain DE).

The reason behind the choice of ν as a measurement of the population diversity instead of ξ proposed in [17] and Article **PI** or ψ proposed in Article **PII** can be explained in the following way. Articles [17], **PI** and **PII** propose for various applications and fitness landscapes adaptive multimeme algorithms which employ a *plus* strategy in the spirit of the Evolution Strategy framework [10]. On the contrary, the DE framework in the EMDE employs a *steady-state* logic.

Indexes ξ and ψ are both proportionate to the absolute value of the difference between the best and average fitness values and are thus very sensitive indexes to variations in population diversity. This feature of ξ and ψ was designed on purpose since the survivor selection in a plus strategy logic leads to a loss of genotypical information, therefore it was fundamental to detect abrupt variations in population diversity in order to prevent stagnation and premature convergence. The steady-state logic of the DE is not strongly subject to premature convergence; on the other hand, DE is mainly subject to stagnation [55]. Although stagnation is in general an undesirable situation, the generation of a super-fit individual is not necessarily to be avoided over the entire evolutionary process. Therefore, our aim is to apply an index which is less sensitive to fitness diversity variations (ν) and, moreover, depends on the

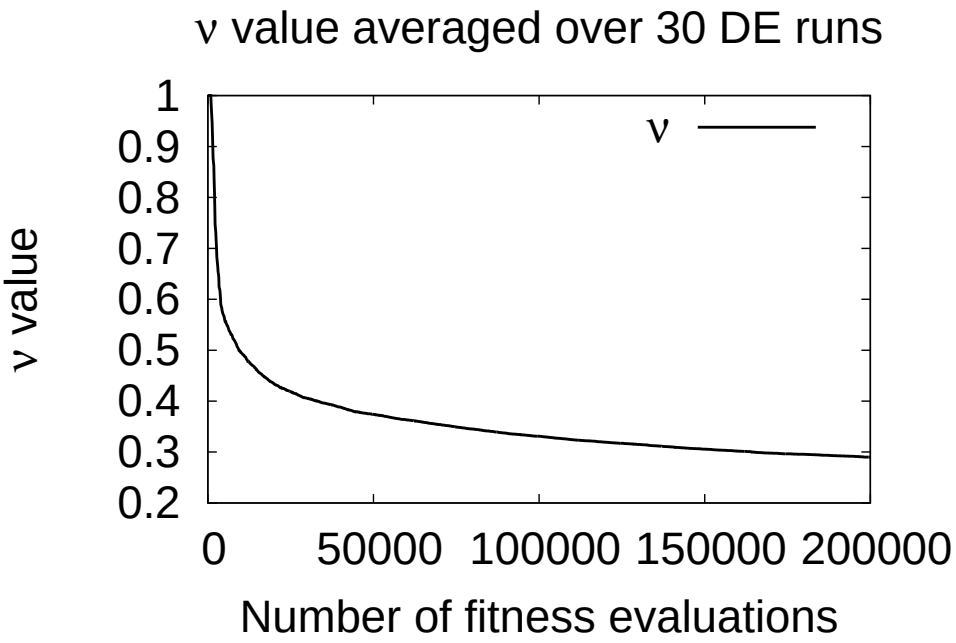

FIGURE 13 Average trend of ν for a plain DE framework

standard deviation and thus on the fitness distribution over all individuals of the population.

More specifically, our algorithmic philosophy aims, at each evolutionary stage, to detect the most suitable local searcher by analyzing the effect of the various algorithmic components on the optimization process. In addition, the assistance of the local searchers is supposed to prevent stagnation by offering alternative exploratory perspectives. For example, if one individual has, by chance, been improved by the DE and has much better performance than the others, the Meta-Lamarckian employment of the SLS to a mediocre individual would lead to its enhancement offering a fresh genotype for the subsequent DE 1000 fitness evaluations.

Considering the preliminary study shown above, the index ν is used to activate local searchers. A value ϵ is sampled by means of a uniform distribution function within the interval $[0, 1]$. Then, the following exponential distribution function is considered [67]:

$$P\left(\mu_p, \sigma_p, \nu\right) = exp\left(\frac{-\left(\nu - \mu_p\right)}{2\sigma_p^2}\right), \tag{8}$$

where $\sigma_p = 0.1$ and $\mu_p \in \{0.1, 0.2, 0.3, 0.4\}$. Taking into account the preliminary results shown above, the following novel probabilistic scheme is proposed.

(a) if $\epsilon < P\left(0.4, \sigma_p, \nu\right)$, one individual is pseudo-randomly extracted from the population and the SA is applied for 500 fitness evaluations

(b) if $\epsilon < P\left(0.3, \sigma_p, \nu\right)$, the HJA is applied for 500 fitness evaluations to the individual having the best performance

(c) if $\epsilon < P\left(0.2, \sigma_p, \nu\right)$, one individual is pseudo-randomly extracted from the population and the SLS is applied for 500 fitness evaluations

(d) if $\epsilon < P\left(0.1, \sigma_p, \nu\right)$, the SLS is applied for 500 fitness evaluations to the individual having the best performance

Conditions (a) and (c) mean that alternative exploratory perspectives are offered to the DE framework and it is taken into account that for a bad initial solution (early stage of the evolution) the SA performs better than the SLS and conversely for a good initial solution (late stage of the evolution) the SLS performs better than the SA. Conditions (b) and (d) are related to the best individual of the population. Condition (b) states that a promising basin of attraction must be immediately descended by means of a highly exploitative local searcher (HJA). On the contrary, condition (d) is related to the late stages of the evolution and states that finalization of the optimization process can be more efficiently performed by a local searcher rather than by an evolutionary framework [68].

In addition, two heuristic rules have been employed. According to the first one, if the application of SA does not lead to any improvement, the initial solution is not replaced. According to the second one, the HJA is never applied twice to the same individual if a previous HJA application did not already lead to any improvement.

The index ν is also used to execute the parameter setting of the local searchers. Concerning HJA and SLS, $\sigma_{SLS-0} = \nu/2$ and $h_{HJA-0} = \nu/2$. This choice means that the initial radius of exploration of the local searchers should be large when the fitness diversity is high and small when the fitness diversity is low. More specifically, since the smaller ν is, the nearer the end of the optimization process is (see articles **PVIII**, **PI** and **PII**), when ν is small the local searchers attempt to detect a better performing solution within the neighborhood of the starting solution, when ν is large the local searchers have a more explorative behavior. Regarding the SA, the initial temperature $Temp_0$ is adaptively set to be $Temp_0 = \nu$. This means that the probability to accept a worse solution depends on the fitness diversity. In other words, the algorithm does not accept worse solutions when the fitness diversity is low and thus the SA does not attempt, with a high probability, to accept solutions which have performance worse than the initial one.

generate initial population pseudo-randomly;
while budget condition
  initialize fitness counter to 0;
  while fitness counter$<$ 1000
    execute DE recombination and offspring generation;
  end-while
  compute $\nu = \min\left\{1, \frac{\sigma_f}{|f_{avg}|}\right\}$;
  sample $\epsilon \in [0,1]$
  if $\epsilon < e^{\frac{-(\nu-0.4)}{2\sigma_p^2}}$
    execute SA on an individual pseudo-randomly selected;
  end-if
  if $\epsilon < e^{\frac{-(\nu-0.3)}{2\sigma_p^2}}$
    execute HJA on the individual having the best performance;
  end-if
  if $\epsilon < e^{\frac{-(\nu-0.2)}{2\sigma_p^2}}$
    execute SLS on an individual pseudo-randomly selected;
  end-if
  if $\epsilon < e^{\frac{-(\nu-0.1)}{2\sigma_p^2}}$
    execute SLS on the individual having the best performance;
  end-if
end-while

FIGURE 14 EMDE pseudo-code

The algorithm stops when 200000 fitness evaluations have been executed. Figure 14 shows the EMDE pseudo-code.

2.3 Numerical Results

2.3.1 Experimental Setup

For the EMDE 30 simulation experiments have been executed. Each experiment has been stopped after 200000 fitness evaluations. Every 100 fitness evaluations, the best fitness value has been saved. The average over the 30 experiments defines the Average Best Fitness (ABF). Analogously, 30 experiments have been carried out with a Genetic Algorithm (GA), an Evolution Strategy

TABLE 2 Optimization results

	GA		ES		SA		DE		MDE		EMDE	
	Fil 1	Fil 2	Fil 1	Fil 2	Fil 1	Fil 2	Fil 1	Fil 2	Fil 1	Fil 2	Fil 1	Fil 2
wgh	-9.873	8.164	-5.893	-41.058	0.492	41.483	2.026	91.448	68.139	-18.267	7.108	79.172
θ	0.709	1.937	0.918	2.320	2.882	1.581	1.014	1.512	1.521	1.763	2.925	1.510
ψ	1.518	2.710	1.008	4.622	0.187	0.594	2.838	4.092	1.724	4.135	5.300	4.185
σ_x	1.213	1.629	1.952	5.875	3.681	1.585	1.000	1.519	7.802	4.713	1.573	17.329
σ_y	1.593	1.816	4.552	13.751	2.001	3.160	1.000	15.737	3.064	3.116	6.333	9.484
λ	4.650	4.058	2.330	0.944	0.465	3.713	0.684	3.522	3.549	1.751	2.332	3.469
f^b	-1.056		-1.273		-1.361		-1.491		-1.526		-1.514	
<f>	-1.031		-1.139		-1.181		-1.472		-1.491		-1.491	
f^w	-0.987		-1.050		-0.978		-1.433		-1.462		-1.462	
σ_{exp}	0.035		0.064		0.116		0.016		0.013		0.012	

(ES), a SA metaheuristic, a plain DE and the Memetic Differential Evolution (MDE) proposed in Article **PIII** in order to perform a comparison of the performance between the EMDE and other popular meta-heuristics.

A standard generational Genetic Algorithm (GA) [69] with $S_{pop} = 100$ has been implemented. The GA employs a pseudo-random initial sampling, linear ranking parent selection with stochastic universal sampling, and arithmetic crossover, Gaussian mutation [5].

An Evolution Strategy (ES) with $S_{pop} = 100$ for our problem has been implemented. As a standard ES, this ES does not contain any parent selection and, thus, it considers all populations as a population of parents [10]. A standard intermediate recombination has been chosen and the Gaussian mutation has been implemented resorting to the $1/5$ success rule [9]. Finally, a $(\mu + \lambda)$ strategy has been chosen.

The same SA explained in Section 2.2 with initial temperature $Temp_0 = 1$ and hyperbolical reduction of the temperature has been run 30 times for 200000 fitness evaluations. The same DE explained in Section 2.2 and employed in the DE framework has been applied as well.

2.3.2 Optimization Results

Table 6 shows results of the optimization process for the six algorithms under study. Table 6 shows the design parameters obtained at the end of the most successful experiments for both the filters Fil 1 and Fil 2, the corresponding fitness value f^b, the worst (f^w) and the average (<f>) fitness values over the 30 experiments carried out and the corresponding standard deviation σ_{exp}.

Results in Table 6 show that the the EMDE outperforms, in terms of final value, GA, ES, SA and DE. Moreover, the EMDE reaches final values which

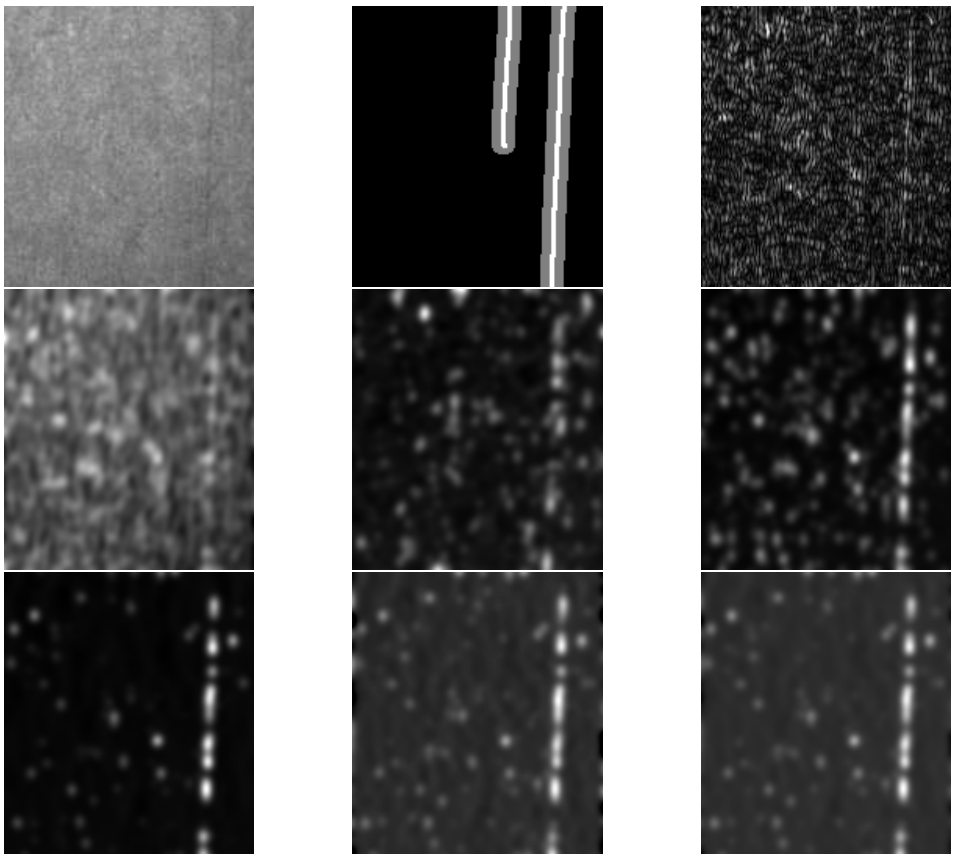

FIGURE 15 First image belonging to the training set. From the upper left corner source image, label image, filtered image by Sobel mask, GA, ES, SA, DE, MDE and EMDE

perform approximately as good as for the MDE. Moreover, the algorithms based on a DE framework outperform the standard metaheuristics (GA, ES and SA) for the problem under analysis. This result confirms and extends the study in [70] which proves the superiority of the DE with respect to GAs for a similar application.

Figure 15, Figure 16, Figure 17 and Figure 18 show the source image, label image, image filtered by an horizontal Sobel mask [33] and the filtered image obtained by the best performing solutions obtained from each algorithm (the filter parameters are shown in Table 6). Figure 15 and 16 refer to two images belonging to the training test whilst Figure 17 and 18 refer to two images not belonging to the training test.

Figures 15, 16, 17, and 18 show that the performance of the Sobel mask is clearly inferior to all solutions derived by optimization methods.

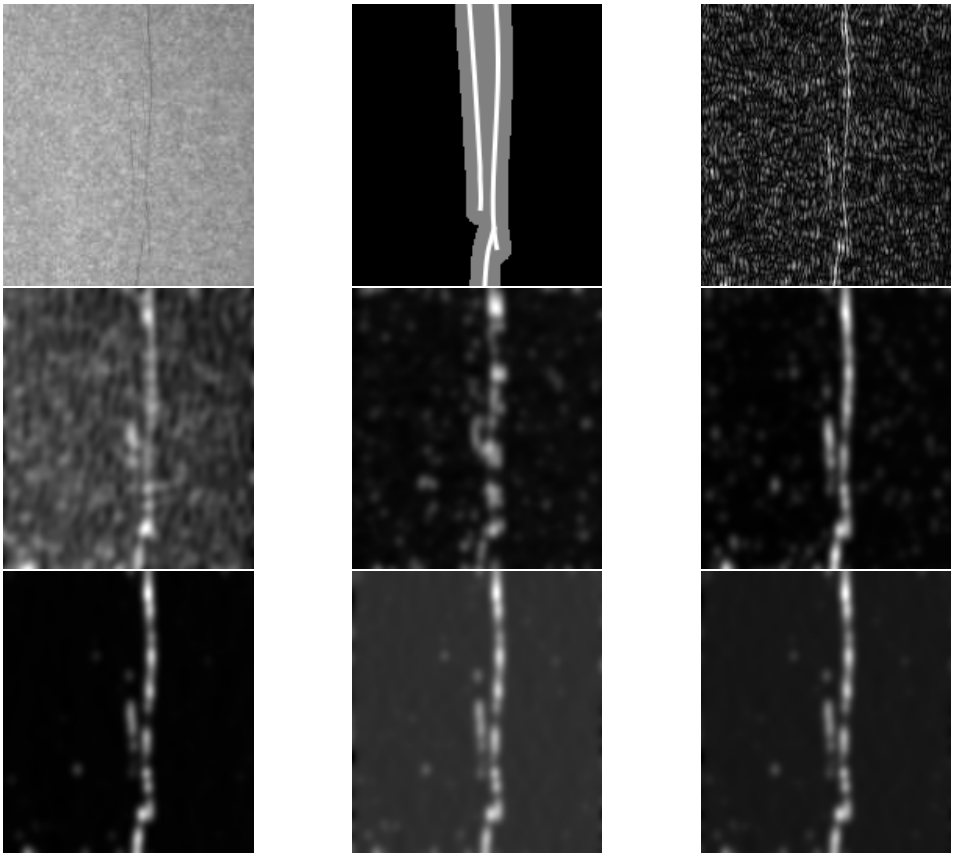

FIGURE 16 Second image belonging to the training set.

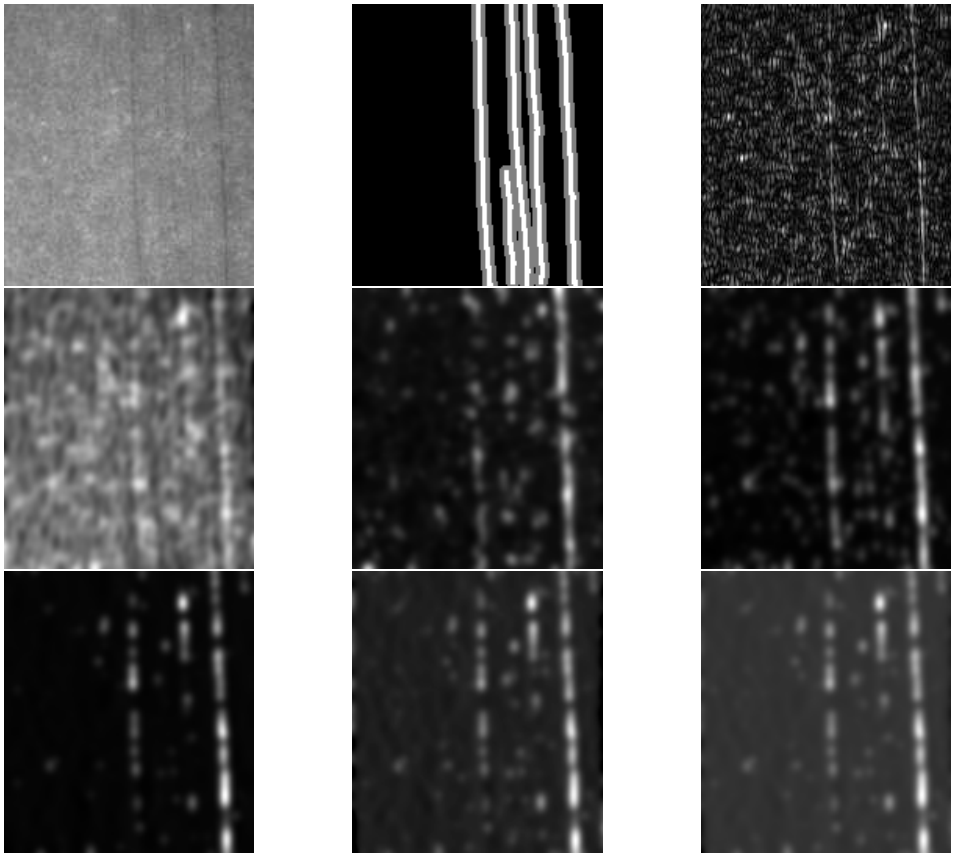

FIGURE 17 First image not belonging to the training set.

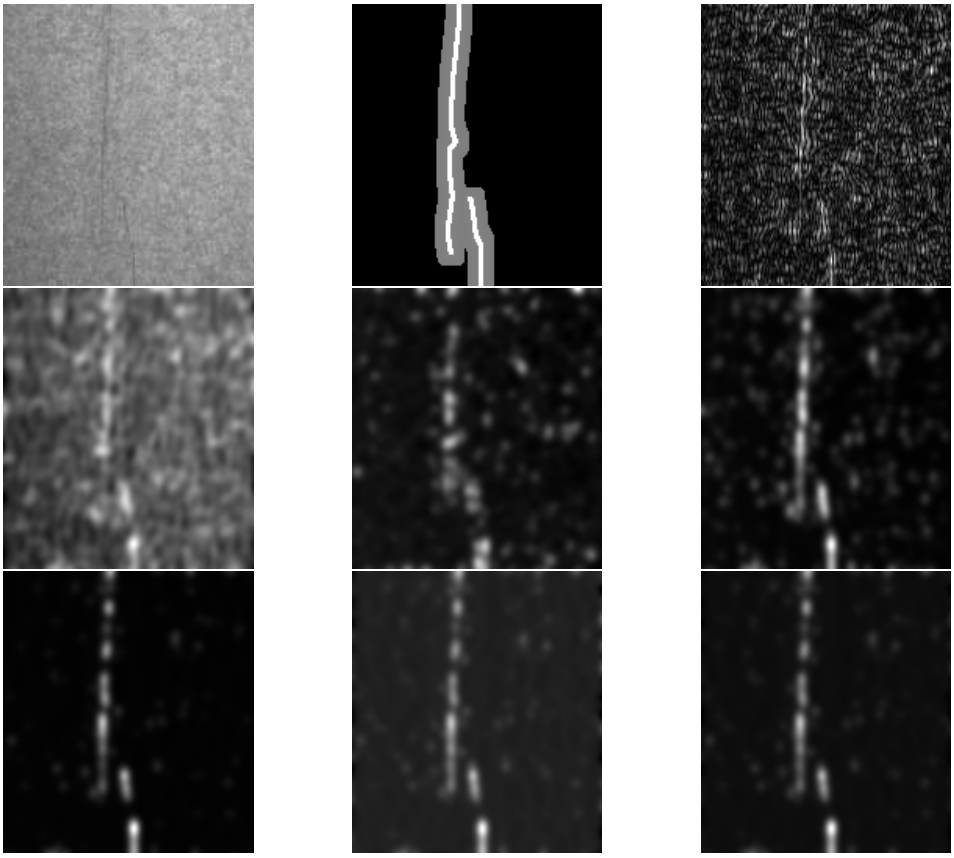

FIGURE 18 Second image not belonging to the training set.

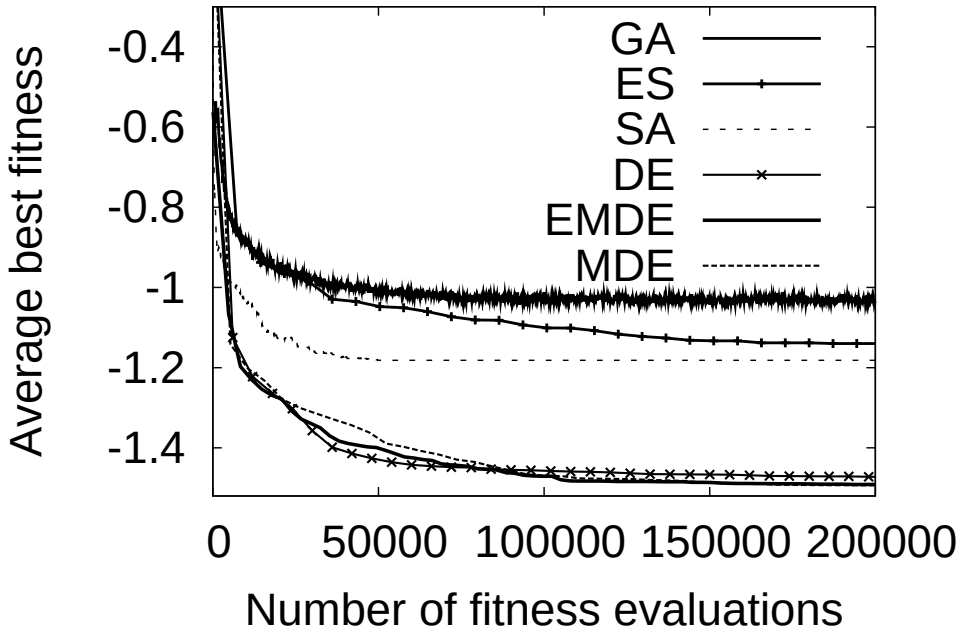

FIGURE 19 Algorithmic performance

2.3.3 Analysis of the Performance

Figure 19 shows the comparison of the algorithmic performance for the six algorithms under study. Results in Table 6 and Figure 19 confirm that the algorithms employing the DE logic clearly outperform the other three metaheuristic. The comparison between the MDE and the DE shows that the MDE is slightly slower in reaching the optimum but eventually outperforms the DE. According to our interpretation, the presence of local searchers softens the explorative feature of the DE framework thus temporarily slowing down the optimization process. On the other hand, local searchers exploit the available genotypes and give a better chance for the DE framework to detect promising search directions. Regarding the proposed algorithm, the EMDE outperforms the MDE in terms of convergence velocity since it is almost as quick as the plain DE and converge to final values as good as the MDE. In the first half of the evolution, the SA efficiently assists the DE framework and balances the slowdown due to exploitative local searchers (HJA and SLS). This effect leads to a slight improvement, with respect to MDE, in convergence velocity performance. In the second half of the evolution, as well as the MDE, the EMDE exploits the available genotypes by means of HJA and SLS and finalizes the optimization process outperforming the plain DE.

Figure 20 shows the trend of ν averaged over 30 experiments vs the num-

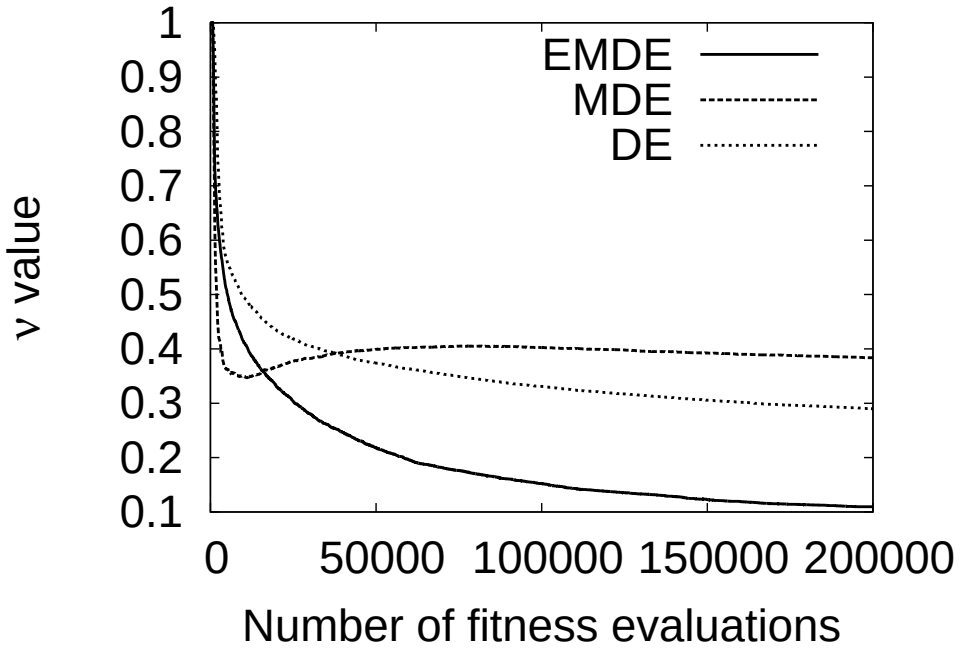

FIGURE 20 Trend of ν

ber of fitness evaluations for DE, MDE and EMDE. Since, as highlighted above, ν is a measurement of fitness diversity and distribution over the individuals of the population, it follows that if the trend of ν takes on a constant non-null value for a large portion of the evolution, the algorithm is probably stagnating. Moreover, the higher the value of ν the less possibility the algorithm has of eventually converging. On the basis of this analysis, Figure 20 shows that the EMDE reaches a lower value of ν than the DE and MDE and is thus better performing in terms of stagnation prevention. The non-monotonous behavior of ν for the MDE is due to the fact that the significant improvement of some solutions caused by the exploitative local searchers is not properly transmitted to the other solutions during the functioning of the DE framework. On the contrary, the EMDE seems to be more promising in this sense because ν tends systematically to decrease and reaches a much lower value. We believe that a proper control in the behavior of ν and population diversity is a crucial point in order to execute properly a successful hybridization between DE and local searchers.

According to our interpretation, the improvement in terms of stagnation prevention with respect to the MDE is due to the proposed probabilistic scheme and the shorter runs of local searchers. In fact in **PIII** the local searchers were run for 1000 fitness evaluations and their coordination was executed by means of a threshold based adaptation. The shorter local searcher runs allow

the DE framework, for this application, to better transmit the improvement gained by a single solution to the other individuals of the population. The probabilistic adaptive scheme gives a chance, although with a low probability, to apply all three of the local searchers during the whole of the evolution process. This fact introduces a casuality in the system which seems to lead some improvements in the algorithmic performance for the class of problems under study.

2.4 Conclusion

Numerical results show that the DE based algorithms outperform other popular metaheuristics for the class of problems under study. The comparison with a plain DE shows that the EMDE is able to reach solutions having slightly better performance. The comparison with the MDE previously carried out by the same authors shows that the EMDE outperforms the previous Memetic Differential Evolution in terms of convergence velocity still reaching high quality final values.

The enhancements with respect to the MDE are due to integration, besides the HJA and the SLS, of the SA within the DE framework since it speeds up the early stages of the evolutionary process. Moreover, the proposed adaptive scheme and the new parameter setting regarding the employment of local searchers seems to be very promising in terms of stagnation prevention.

3 SUPER-FIT CONTROL ADAPTATION IN MEMETIC DIFFERENTIAL EVOLUTION FRAMEWORKS

Differential Evolution (DE) [53] is a reliable and versatile function optimizer. DE, like most popular Evolutionary Algorithms (EAs), is a population based tool. DE, unlike other EAs, generates offspring by perturbing the solutions with a scaled difference of two randomly selected population vectors, instead of recombining the solutions by means of a probability function. In addition, DE employs a steady state logic which allows replacement of an individual only if the offspring outperforms its corresponding parent.

Due to its algorithmic structure, over the optimization process DE generates a super-fit individual which leads the search until a better performing individual is generated. As highlighted in [55], a DE population can be subject to stagnation since in the case that no offspring individuals outperform the corresponding parents for a large number of generations. On the other hand, the persistence of a super-fit individual over a certain number of generations does not necessarily imply poor functioning of the algorithm, this may be a "natural" stage of the evolution if the other individuals are somehow updated during the run. It is therefore fundamental to find a proper balance between the enhancements of the search leader (super-fit individual) and the remaining individuals of the population.

In order to prevent stagnation, several studies have been carried out in recent years. In [55], on the basis of an experimental study, some suggestions on how to perform the parameter setting are given. In [71] a dynamic parameter setting changing in accordance with time is proposed. A further study on the proper parameter setting in a DE framework is performed in [72] and a fuzzy logic based criterion in order to automatically solve this problem is pro-

posed in [73]. This fuzzy logic approach has been analyzed in more depth in [74] and [75]. [76] proposes a fitness based adaptive setting of the scale factor. [77] extends the work in [76] and proposes the concept of competition during on-line parameter setting. In [78] randomizing the scale factor of the perturbation vector is proposed. This operation leads to an increase in the pool of potential trial vectors, thus reducing the risk of stagnation without increasing the population size. Recently, a further parameter study with reference to real world problems has been carried out in [79].

In order to enhance performance several approaches which hybridize the DE with other optimization techniques have also been considered. In [80] the DE framework is embedded with two additional operators which have the role of increasing convergence velocity of the DE without jeopardizing the algorithmic features in terms of population diversity. In [81] an enhancement of the algorithm in [80] for solving constrained optimization problems is proposed. [82] proposes a hybridization of the DE with the Powell method for accelerating the DE performance with reference to an electrical engineering problem. Other hybridizations with single local search algorithm are given in [83] and [84]. In [85] the combination of the DE with Particle Swarm Optimization Algorithm is shown. [86] proposes a hybrid DE for mixed integer-real coded problems. [87] and [88] propose a co-evolutionary and an enhanced implementation of the algorithm described in [86] respectively. In [89] the DE has been hybridized with a multiplier updating method for constrained problems. [90] proposes the hybridization of the DE with an Ant Colony Optimization algorithm and [91] with the Salomon's Evolutionary Gradient Search method. In [92] a hybrid DE with a mutation local searcher based on the attract-repulsion logic is proposed. **PIII** proposes a hybridization of a Differential Evolution (DE) framework with two local searchers adaptively coordinated by means of a rule based on fitness diversity. Moreover, **PIII** applies the resulting algorithm to the design of a Finite Impulse Response (FIR) filter for defect detections in paper production.

This section proposes a novel implementation of an adaptive Memetic Algorithm (MA) [64] employing the DE as an evolutionary framework. This algorithm, namely Super-Fit Memetic Differential Evolution (SFMDE) makes use of a Particle Swarm Optimization (PSO) Algorithm, the Nelder Mead Algorithm (NMA) and the Rosenbrock Algorithm (RA) within the DE framework. Coordination of the local searchers is adaptively executed by means of a parameter which measures the quality of the super-fit individual with respect to other individuals of the population.

3.1 Super-Fit Memetic Differential Evolution

For a given minimization problem of an objective function $f(x)$ where x is a vector of n design variables $x(1), x(2), \ldots, x(i), \ldots, x(n)$ in a decision space H, the SFMDE consists of the following.

3.1.1 Generation of the Super-Fit Individual by Particle Swarm Optimization

To begin with, the algorithm generates the initial population by pseudo-randomly sampling (uniform distribution) S_{pop} individuals within the decision space. Fitness values of these individuals are calculated and the one having the best performance is detected.

The best performing solution and $S_{pop}^{PSO} \leq S_{pop}$ individuals, pseudo-randomly selected from the initial population, undergo PSO [93], [94]. The best performing solution initializes the particle best x_{pb} (the best overall solution detected) while the best performing solution amongst the S_{pop}^{PSO} individuals is called the global best x_{gb}. At each generation the individuals are perturbed by means of a velocity vector. For a given solution x_i the update rule is given by:

$$v_i = v_i + c_1\gamma\left(x_{pb} - x_i\right) + c_2\gamma\left(x_{gb} - x_i\right), \tag{9}$$

$$x_i = x_i + v_i, \tag{10}$$

where γ is a pseudo-random value sampled in $[0, 1]$ and c_1, c_2 are constant parameters called learning factors. The fitness values of the new solutions are calculated and, if an improvement upon the best individuals occurs, x_{pb} and x_{gb} are also updated. The procedure is repeated until a budget condition is not exceeded. The best overall solution detected by the PSO is reinserted in the population made up of S_{pop} individuals by replacing the individual having the worst performance.

The main idea is that the PSO should quickly improve a solution having poor performance and include it in the DE population. This solution should therefore be a super-fit individual, with the role of leading the DE search. According to our algorithmic philosophy, the DE should then exploit the genotypic information of the solution returned by the PSO and at the same time attempt to improve upon it by a massive exploration of the decision space.

In order to justify this algorithmic choice the following preliminary test has been designed. $S_{pop} = 50$ individuals have been sampled in $[-5.12, 5.12]^{10}$, the best individual has been saved and $S_{pop}^{PSO} = 20$ individuals have been pseudo-randomly selected. Next, the PSO has been run for 300 fitness evaluations with $c_1 = c_2 = 2$ in order to minimize the Rastrigin function

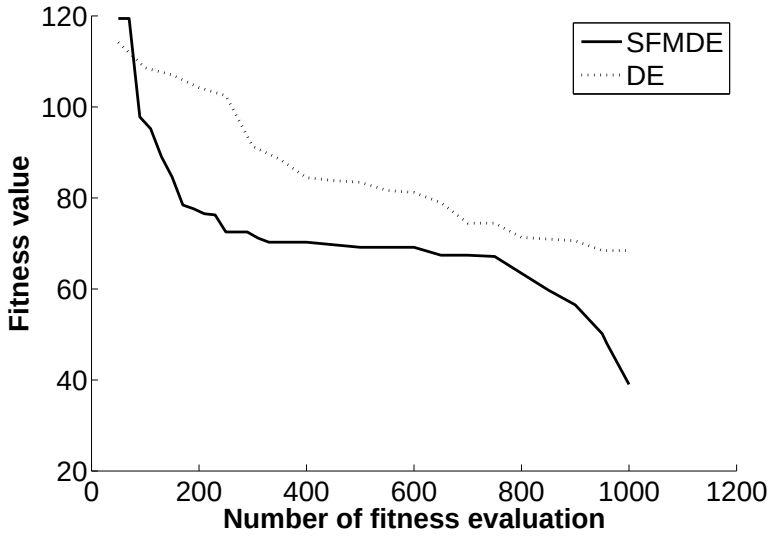

FIGURE 21 Comparison of the SFMDE and DE during early generations for the Rastrigin function

$$R(x) = 10n + \sum_{i=1}^{n} \Big(x\,(i)^2 - 10\cos\left(2\pi x\,(i)\right) \Big). \tag{11}$$

The best resulting individual has been reinserted into the population by replacing the worst individual and the DE has been run for an additional 650 fitness evaluations. A plain differential evolution with $S_{pop} = 50$ has been run for the same function for 1000 fitness evaluations. Each algorithm has been run 30 times. Every 50 fitness evaluations the best fitness value has been saved. Figure 21 shows the average algorithmic performance for the Rastrigin function. With the same experimental setup, performance during the early generations has been analyzed also for the Schwefel function

$$S\,(x) = \sum_{i=1}^{n} -x\,(i) \cdot \sin\left(\sqrt{|x\,(i)|}\right) \tag{12}$$

in $[-500, 500]$ and the Griegwangk's function

$$G\,(x) = \sum_{i=1}^{n} \frac{x\,(i)^2}{4000} - \prod_{i=1}^{n} cos\left(\frac{x(i)}{\sqrt{(i)}}\right) + 1 \tag{13}$$

in $[-600, 600]$. Comparison of the performance for these two test functions is shown in Figure 22 and Figure 23 respectively.

Numerical results in Figure 21, Figure 22, and Figure 23 show that the hybridization of the DE with the PSO in early generations leads to clear benefits in terms of performance. It can be clearly noted, in Figures 21 and 23, that

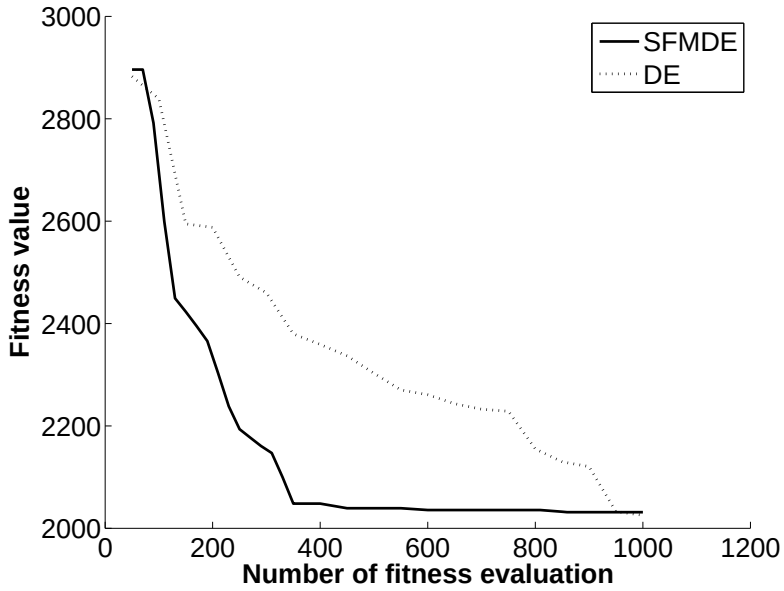

FIGURE 22 Comparison of the SFMDE and DE during early generations for the Schwefel function

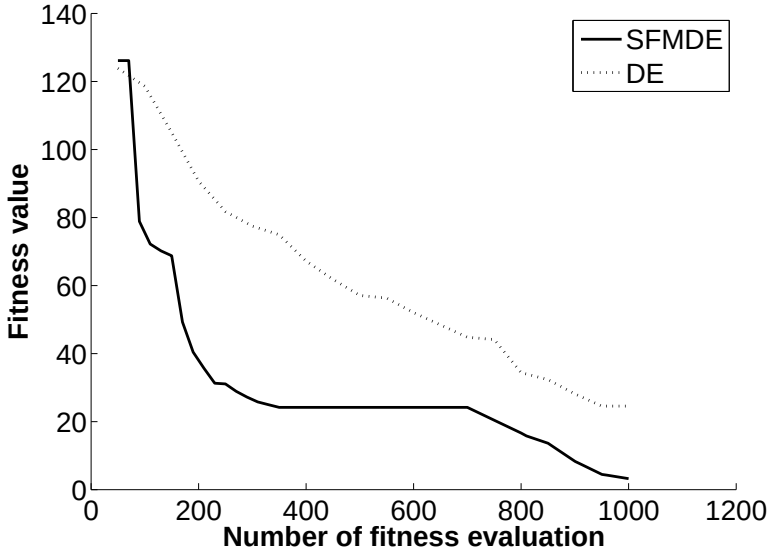

FIGURE 23 Comparison of the SFMDE and DE during early generations for the Griegwangk's function

after application of the PSO (at 350 fitness evaluations) there is a significant improvement in the fitness values which is followed by a mild improvement for a certain portion of the optimization process and subsequently by a new significant improvement. According to our interpretation, the super-fit at the beginning of the process leads the search, enhancing the average fitness value but does not improve upon the super-fit individual. After a certain number of generations, the other individuals of the population manage to improve upon the solution detected by the PSO and an overall advantage with respect to the DE occurs.

3.1.2 Differential Evolution Framework

At each subsequent generation, for S_{pop} times, four individuals x_1, x_2, x_3 and x_4 are extracted from the population pseudo-randomly. Recombination and mutation according to the logic of a DE explained in Section 2.2.1 occurs.

3.1.3 Local Searchers

The SFMDE also makes use of two local searchers which are supposed to compete and cooperate [65] in order to assist the DE framework at detecting promising search directions in a meta-Lamarckian logic.

The Nelder Mead Algortihm

The Nelder Mead Algorithm (NMA) [95] works on a set of $n+1$ solutions in order to perform a local search since it employs an exploratory logic based on a dynamic construction of a polyhedron (simplex). More specifically, for a given set of $n+1$ solutions $x_0, x_1, \ldots, x_n$ sorted in descending order according to their fitness values (i.e. x_0 is the best), the NMA attempts to improve x_n. In order to pursue this aim the NMA calculates the centroid x_m of the polyhedral identified by the remaining n points:

$$x_m = \frac{1}{n} \sum_{j=0}^{n-1} x_j \tag{14}$$

Subsequently, the NMA generates a trial solution x_r by reflecting x_n through the opposite face of the polyhedron. This operation, namely reflection is characterized by:

$$x_r = x_m + F_1 \left(x_m - x_n \right) \tag{15}$$

where F_1 is a weighting factor.

If a reflection is successful i.e. $f(x_r) < f(x_0)$, NMA further exploits a promising search direction by applying the expansion operation:

$$x_e = x_r + F_2 (x_m - x_n) \tag{16}$$

where F_2 is a weighting factor. If the expansion is also successful x_e replaces x_0 and the new set of $n+1$ points is used for the subsequent iteration. Conversely, if the expansion fails then x_r replaces x_0.

If x_r did not improve upon x_0, NMA compares the performance of x_r and x_{n-1}. If $f(x_r) < f(x_{n-1})$, then x_r replaces x_n. If this trial is also unsuccessful, x_r and x_n are compared. If $f(x_r) < f(x_n)$, however x_r replaces x_n and the outside contraction operation is executed:

$$x_c = x_m + F_3 (x_m - x_n) \tag{17}$$

where F_3 is a weighting factor. If x_r does not outperform x_n then the inside contraction is executed:

$$x_c = x_m - F_3 (x_m - x_n) \tag{18}$$

The effect of the contraction is then analyzed: if the contraction was successful i.e. $f(x_c) < f(x_n)$ then x_c replaces x_n. If on the contrary contraction fails the shrinking is executed. A set of n solutions $x_j = 0.5 \left(x_0 - x_j\right)$ with $j = 1, 2, \ldots, n$, is generated around x_0.

The algorithm is repeated with these new n solutions and x_0. F_1, F_2 and F_3 have been set equal to 1, 1, 0.5 as suggested in [96].

The Rosenbrock Algorithm

The Rosenbrock Algorithm (RA) works on a solution and attempts to improve upon it by means of a local search logic [6]. From a starting point x_0, a trial is made in all the n orthogonal directions of the n-dimensional decision space. When a solution that is better or equal (i.e., a success) in the objective function to previous solutions is found, the changed variable vector is retained and the step length is multiplied by a positive factor $\alpha > 1$. For a failure, the vector of variables is left unchanged and the step length is multiplied by a negative factor $-1 < \beta < 0$. Following the Rosenbrock's suggestion $\alpha = 3$ and $\beta = -0.5$ has been set. This process is repeated until at least one success followed by a failure is registered in each direction. When such a condition is satisfied, the orthogonalization procedure of Gram and Schmidt (see [97]) is executed and the search, with the new set of directions, begins again. The algorithm is stopped when a budget condition is exceeded.

3.1.4 Comparative Analysis of the Local Searchers

The choice of these two local searchers has been made taking into account their features and features of the DE framework. At first glance, it can be seen that both the NMA and RA are very exploitative compared to the DE. In fact, the DE framework is very explorative and the two local searchers are supposed to offer different perspectives in exploration of the decision space [26]. Secondly the two local searchers are very different with respect to the pivot rule and the neighborhood generating function [4]. It is clear that the RA, since at each step it explores a number of points equal to the dimensionality of the problem and only afterwards performs a movement, has a steepest descent pivot rule. The NMA, on the other hand, tends to accept the first search direction which leads to an improvement with respect to the previous best solution; thus the NMA has a greedy descent pivot rule. It follows that the RA is very efficient in performing the hill-climbing (hill-descending) and locally exploring a basin of attraction. Regarding the neighborhood generating function, while the RA explores the neighborhood of a given solution, the NMA attempts to explore directions given by other solutions. The NMA neighborhood generating function thus allows detection of new points not belonging to the same basin of attraction as the starting point. In this sense, it might be stated that the NMA has more explorative features compared to the RA. Therefore, according to our algorithmic philosophy, the NMA can be used to further improve some fairly promising solutions while the RA can be used to finalize the hill-descent of the basin of attraction at the end of the optimization process.

3.1.5 Adaptation

At the end of each DE generation the following parameter is calculated:

$$\chi = \frac{\left|f_{best} - f_{avg}\right|}{max\left|f_{best} - f_{avg}\right|_k} \tag{19}$$

where f_{best} are f_{avg} are the fitness values of, respectively, the best and average individuals of the population. $max\left|f_{best} - f_{avg}\right|_k$ is the maximum difference observed (e.g. at the k^{th} generation), overall, beginning from the start of the optimization process. It is clear that χ varies between 0 and 1; it scores 1 when the difference between the best and average fitness is the biggest observed, overall, and scores 0 when $f_{best} = f_{avg}$ i.e. all the population is characterized by a unique fitness value.

Besides considering it as a measurement of the fitness diversity (see [17] and Article **PII**), χ is an estimation of the best individual performance with respect to the other individuals. In other words, χ measures how much the

super-fit outperforms the remaining part of the population. More specifically, the condition $\chi \approx 1$ means that one individual has a performance far above the average and thus one super-fit individual is leading the search. Conversely, the condition $\chi \approx 0$ means that the performance of the individuals are comparable and there is not a super-fit. As a general guideline, a DE population which contains a super-fit individual needs to exploit the direction offered by the super-fit in order to eventually generate a new individual that outperforms the super-fit. Conversely, a DE population made up of individuals with comparable fitness values requires that one individual that clearly outperforms the others is generated in order to have a good search lead.

3.1.6 Coordination of the Local Searchers

The parameter χ is employed to perform coordination of the local searchers. According to our algorithmic design, the SFMDE adaptation (and coordination of the local searchers) is based on an attempt to intelligently balance the DEs' necessity to generate a super-fit individual and prevent stagnation due to an excessive difference between the best performing individual and the others. More specifically, for each local searcher a generalized beta distribution function is generated:

$$p(\chi) = \frac{1}{B(\alpha, \beta)} \cdot \frac{(\chi - a)^{(\alpha-1)} (b - \chi)^{(\beta-1)}}{(b - a)^{(\alpha+\beta-1)}}, \tag{20}$$

where a and b are respectively the inferior and superior limits of the distribution; $B(\alpha, \beta)$ is the beta function; α and β are the shape parameters. For the RA α and β have been set 2 and 5; for the NMA 2 and 2. Both the distribution functions have been normalized with respect to the maximum of the distribution in order to have co-domain in $[0, 1]$. At each generation of the SFMDE, a value of χ is used for determining the probability of activating each of the local searchers. Each of the probability values are compared with a pseudo-random number generated between 0 and 1. If this number is lower than the probability value, the corresponding local search is performed. In addition, it has been set that the RA is applied to the best performing individual of the population while the NMA is applied to pseudo-randomly selected individual. For both local searchers, the improved solution replaces the worst performing individual of the DE population.

Figure 24 gives a graphical representation of the probability functions related to the local searcher activations.

Choice of performing the coordination of the local searchers by means of a probabilistic criterion has been done taking into account that the introduction

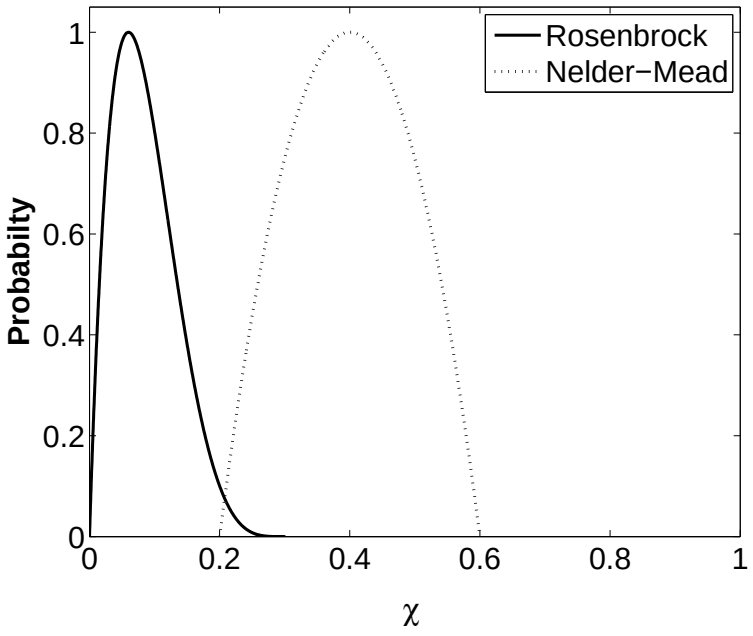

FIGURE 24 Graphical Representation of the Probabilistic Scheme for Activating Local Searchers

of random components within a DE framework seems to be beneficial in terms of stagnation prevention, analogous with the studies in [78].

Choice of the generalized beta distribution is derived from the fact that such distribution is two-side bounded and therefore is suitable for our aim. Moreover, the shape parameters allow a detailed design of the probability distribution on the basis of the the corresponding local searcher features. In particular, as with the NMA, $\alpha = \beta = 2$ leads to a symmetric distribution function around an arranged value. This choice has been made due to consideration that the NMA can lead to improvement of fairly good solutions by exploiting a promising search direction and moderately exploring a region of the decision space. An application in presence of a highly super-fit individual can lead to a waste of computational efforts since the DE framework has not yet explored the available promising search direction. An application in a very low fitness diversity population can likely turn out to be ineffective since the population would probably fall within the final basin of attraction and the NMA is not very efficient in finalizing the optimization process (see [17]). Thus, the NMA is supposed to work on a pseudo-randomly selected individual in presence of a best performing individual having a fitness value fairly above the average. The NMA, therefore, has the role of assisting the DE framework either by of-

fering alternative search directions or by outperforming the current super-fit. The symmetric probability function can be justified as implementation with the intention of having a maximum employment of the NMA around the condition described above and a gradual decrease as long as the algorithm moves from it in both the directions.

Regarding the RA, the idea is to utilize exploitative features of the algorithm and its steepest descent pivot rule in order to improve quality of the best performing individual, attempting to either restore a population led by a super-fit or hill-descent in the final basin of attraction, thus ending the optimization process. The asymmetrical shape of the distribution has been chosen in order to achieve a high probability of RA activation in proximity of the zero and a quick decrease as long as the χ value increases. Noteworthy is the right handed tail of the distribution (see values around 0.2 in Figure 24). This shape allows the algorithm to assign a low probability of executing the RA even in low diversity conditions that are not so extreme. Thus, if the best performing individual falls within a wide basin of attraction, the algorithm gives a chance for the RA to quickly finalize the search. Finally, one heuristic rule has been implemented in order to avoid a waste of computational overhead: the RA is not run again on a solution if previously it failed in improving upon it. Figure 25 shows the SFMDE pseudo-code.

3.2 Application 1: Design of a DC Motor Speed Controller

Nowadays most motion actuators are set up by electric motors since they offer high performance in terms power density, efficiency, compactness and lightness. On the other hand, in order to have satisfactory functioning of the motor, an effective control is needed. Basically, an efficient motor control can be achieved either by applying a complex and expensive control system (see [98], [99], [100], [101]) or by using a simple and cheap control system, e.g. Proportional Integral (PI) based [102], which requires a design often very difficult to implement. In the latter case, the control design of an electric motor consists of detecting those system parameters that ensure a good system response in terms of speed and current behavior. This leads to a multi-objective optimization problem too complex for analytical solution [103]. Moreover, the application of classical design strategy [104], [105], [106] likely leads to unsatisfactory results. Thus, during recent years, interest in computational intelligence techniques has increased (see [107], [108], and [17]).

This paper attempts to apply the SFMDE to the control design of the DC Motor whose electrical and mechanical features are shown in Table 3.

generate initial population pseudo-randomly;
apply PSO to the best performing individual;
replace the PSO result with the worst performing individual of the population;
while budget condition
  initialize fitness counter to 0;
  execute DE generation;
  compute $\chi = \frac{|f_{best} - f_{avg}|}{max|f_{best} - f_{avg}|_k}$;
  sample $\epsilon \in [0, 1]$
  if $\epsilon < \frac{1}{B(2,2)} \cdot \frac{(\chi - a)^{(2-1)}(b - \chi)^{(5-1)}}{(b-a)^{(2+2-1)}}$
    execute NMA on an individual pseudo-randomly selected;
    replace the NMA result with the worst performing individual of the population;
  end-if
  if $\epsilon < \frac{1}{B(2,5)} \cdot \frac{(\chi - a)^{(2-1)}(b - \chi)^{(5-1)}}{(b-a)^{(2+5-1)}}$
    execute RA on the individual having the best performance;
    replace the NMA result with the worst performing individual of the population;
  end-if
end-while

FIGURE 25 SFMDE pseudo-code

TABLE 3 DC Motor Nameplate

Parameter	Value
Armature resistance	2.13 Ω
Armature induction	0.0094 H
Moment of inertia	$2.4e^{-6}$ $Kg \cdot m^2$
Rated armature voltage	12 V
Rated armature current	1.2 A
Rated load torque	0.0213 Nm
Rated speed	400 rad/s

Figure 26 shows the block diagram of the control scheme.
The control scheme is based on dynamic equations of the motor:

$$v_a = R_a \cdot i_a + L_a \cdot \frac{di_a}{dt} + e, \tag{21}$$

$$v_f = R_f \cdot i_f + L_f \cdot \frac{di_f}{dt}, \tag{22}$$

$$e = K\Phi \cdot \omega, \tag{23}$$

$$T = K\Phi \cdot i_a, \tag{24}$$

$$J \cdot \frac{d\omega}{dt} = T - T_r, \tag{25}$$

where v_a is the voltage applied to the armature circuit, v_f is the voltage applied to the excitation circuit, R_a, R_f, L_a, L_f, i_a, and i_f are the resistance, inductance and current for the armature and the excitement circuits respectively, T and T_r are the electromagnetic and load torque respectively, $K\Phi$ is the torque constant, ω is the rotor speed, J is the moment of inertia and e is the voltage generated by the rotor of the electric machine while rotating.

The DC motor control system is composed of two PI controllers. The first is used to control current and the second speed. The PIs transfer functions of the current and the speed controls are respectively $K_{pi} + \frac{K_{ii}}{s}$ and $K_{p\omega} + \frac{K_{i\omega}}{s}$. The speed reference is pre-filtered through a smoothing filter to reduce overshoot and the current required by the control in response to a speed step. The transfer function of the smoothing filter is $\frac{1}{(1+\tau_s m)}$.

With reference to Figure 26, the control design consists of determining the parameters K_{pi}, K_{ii}, $K_{p\omega}$, $K_{i\omega}$, and τ_{sm} which guarantee very small values in rise and settling time, steady state error and overshoots. The decision space $H \subset \Re^5$ is a five dimensional hyper-rectangle given by the Cartesian product constructed around solution x_0 obtained by applying the classical symmetrical optimum (SO) criterion to design the speed regulator and the absolute value optimum (AVO) criterion to design the current regulator [109]. The lower and upper bounds of each interval have been set according to the following equations:

$$x_{lb}(i) = 10^{-6} \cdot x_0(i), \tag{26}$$

$$x_{ub}(i) = 3 \cdot x_0(i). \tag{27}$$

In order to evaluate the performance of each candidate solution, the four speed and load torque step training test shown in Figure 27 is simulated by means of Matlab/Simulink as a discrete time control drive in order to realistically emulate an industrial digital drive. The performance of a candidate

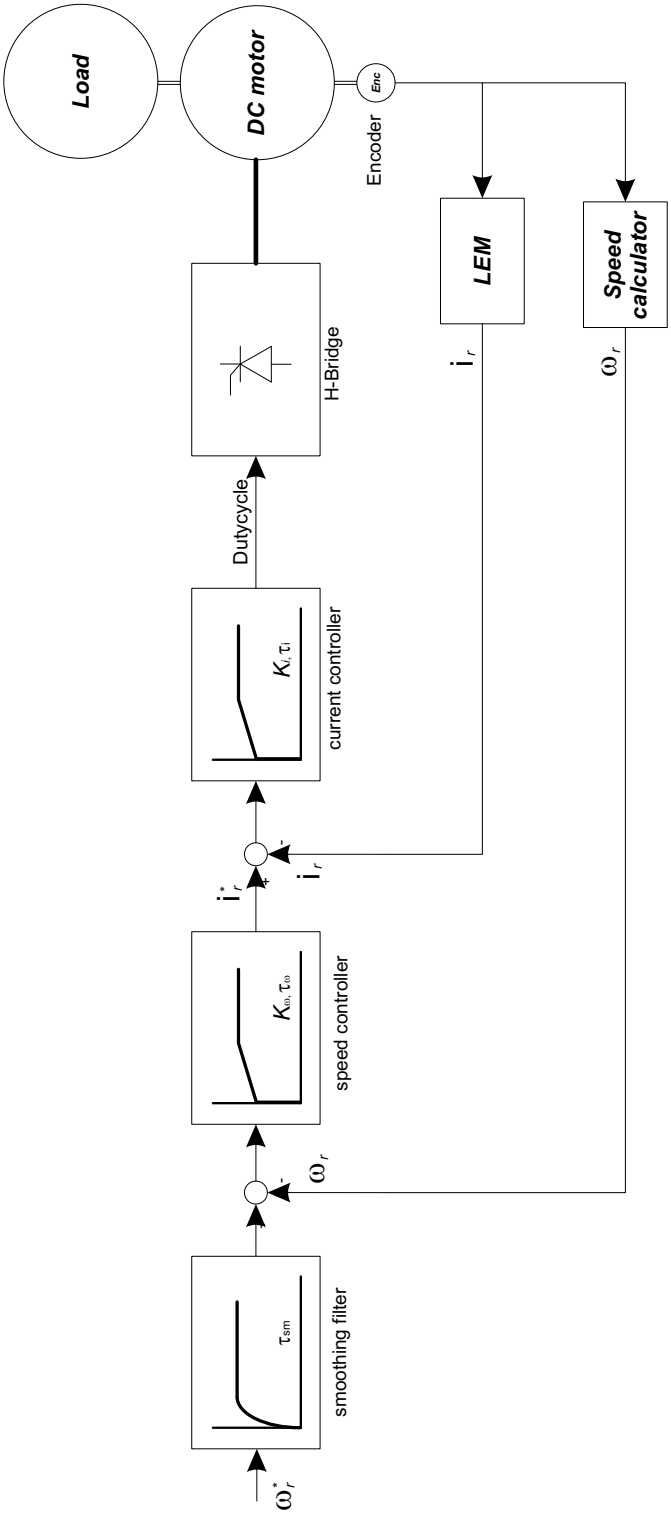

FIGURE 26 Block diagram of a DC motor control

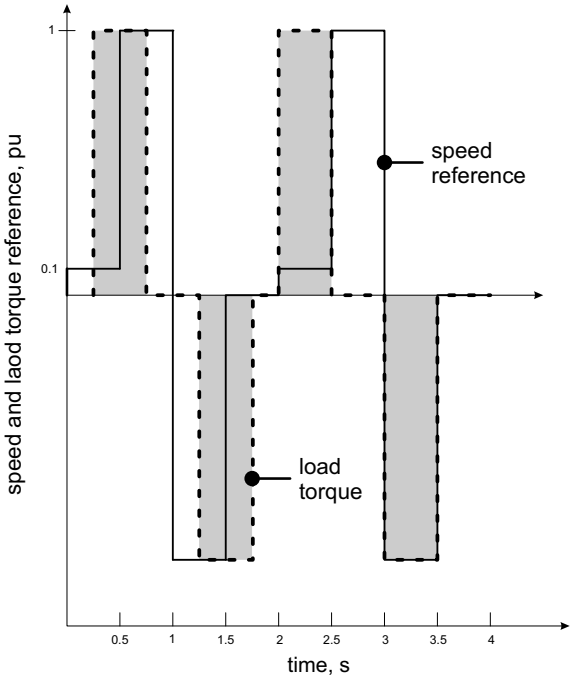

FIGURE 27 Training test is a combination of speed commands and load torque

solution is given by:

$$f = \sum_{j=1}^{4} \left(a_1 \cdot oS_j + a_2 \cdot tr_j + a_3 ts_j + a_4 err_j \right) \tag{28}$$

where oS_j is the overshoot, tr_j the rise time, ts_j the settling time and err_j the sum of the absolute values of the speed error in settling condition during the j_{th} trial step.

Figure 28 illustrates oS_j, tr_j, ts_j, and err_j for the generic j_{th} step of the training test. The weights are set to $a_1 = \frac{1}{50}$, $a_2 = 1$, $a_3 = 1$, and $a_4 = \frac{1}{2000}$ so that any objective function has a value comparable to the others. Finally, it must be remarked that, since each fitness evaluation requires a computationally expensive simulation test (8 s each evaluation, see [110]), the problem is very demanding in terms of computational overhead. The SFMDE has been run in order to minimize, in the decision space H, the fitness function f shown in (28). The performance of the SFMDE has been compared with a plain Differential Evolution (DE), a Genetic Algorithm (GA), the Particle Swarm Optimization (PSO) and the memetic PSO-DE namely Swarm Differential Evolution Algorithm SDEA proposed in [85].

In regard to the SFMDE, the PSO has been executed for 120 fitness evaluations on a population of $S_{pop}^{PSO} = 20$ individuals. Maximum velocity has been

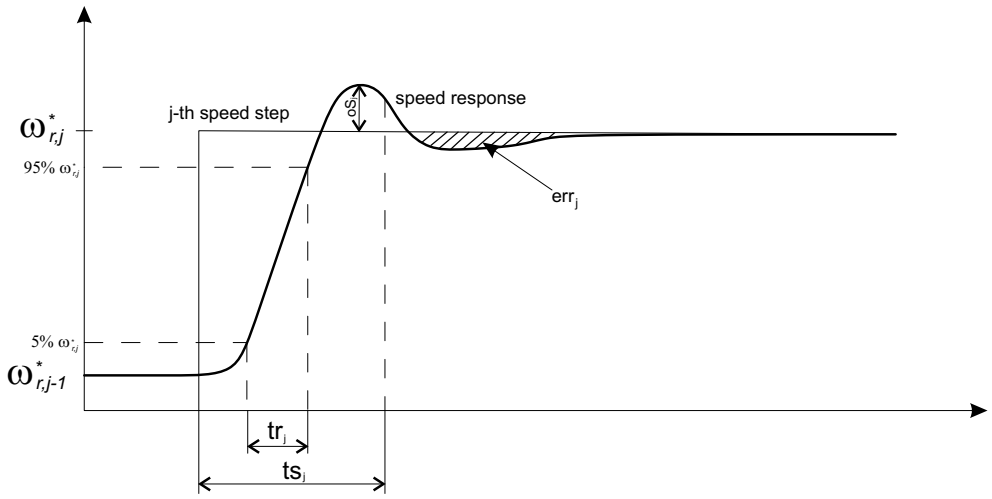

FIGURE 28 j_{th} speed step of the training and values for objective function evaluation

TABLE 4 Best Solutions

Algorithm	$K_{p\omega}$	K_{pi}	$K_{i\omega}$	K_{ii}	τ_{sm}
DE	$1.6652 \cdot 10^{-6}$	3.2192	$2.8063 \cdot 10^{-9}$	721.9234	$4.5837 \cdot 10^{-10}$
GA	$3.2524 \cdot 10^{-4}$	1.7102	$2.8063 \cdot 10^{-9}$	360.4331	$4.2713 \cdot 10^{-4}$
PSO	$2.8632 \cdot 10^{-4}$	1.9576	$2.8063 \cdot 10^{-9}$	425.7751	$4.5837 \cdot 10^{-10}$
SDEA	$2.5748 \cdot 10^{-4}$	2.1296	$2.8063 \cdot 10^{-9}$	469.8608	$4.5837 \cdot 10^{-10}$
SFMDE	$4.3190 \cdot 10^{-4}$	1.3973	$3.1938 \cdot 10^{-5}$	286.5198	$2.6780 \cdot 10^{-4}$

set $0.2(x_{ub} - x_{lb})$. Both, the RA and NMA are run for 100 fitness evaluations each time they are activated. The DE framework works on a population size $S_{pop} = 50$ with $K = 0.7$ and $p_m = 0.3$ as stated above. a and b in (20) have been set equal to 0 and 0.3 for RA and 0.2 and 0.6 for NMA.

The DE has been run with the same parameter setting of the DE framework in the SFMDE.

The GA has been run with a population size equal to 50, a crossover rate 0.8, gaussian mutation with 0.1 mutation rate and elitism equal to 2.

The PSO works on a population of 50 individuals with a maximum velocity equal to $0.2(x_{ub} - x_{lb})$.

The SDEA has also been run on a population of 50 individuals and with the same parameter setting of the PSO and the DE. Parameter T, defined as the ratio of the DE runs divided by the PSO runs, has been set equal to 0.2.

Each algorithm has been run 50 times for 1500 fitness evaluations. The best solutions detected by each algorithm are listed in Table 4.

Figure 29 shows the speed response of the best solution detected by the

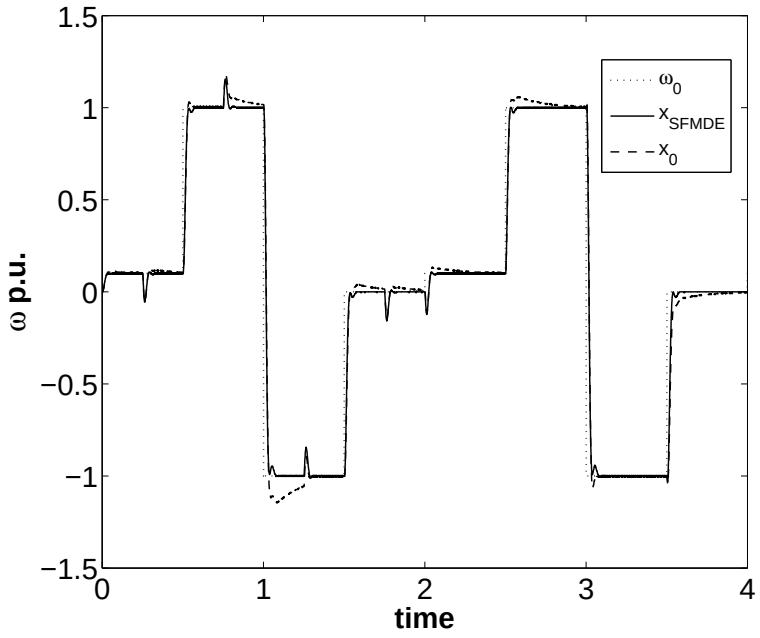

FIGURE 29 Speed response of the best performing solutions

TABLE 5 Numerical Results

Algorithm	f_{min}	f_{max}	f_{mean}	σ
DE	1.1747	1.2486	1.2169	0.0279
GA	1.2601	1.9894	1.4852	0.2972
PSO	1.1697	1.3903	1.2263	0.0922
SDEA	1.1758	1.3877	1.2649	0.1059
SFMDE	1.1078	1.1901	1.1461	0.0360

SFMDE, x_{SFMDE} and the initial solution x_0. The SFDMA solution clearly presents better performance in terms of overshoot, rise and settling time and steady state error.

Figure 30 and 31 show two zoom details of the speed response given in Figure 29. Figure 30 shows the response to a speed step while Figure 31 shows the response to the load torque.

The minimum f_{min}, the maximum f_{max} and the mean f_{mean} final fitness values over the 50 runs are listed in Table 5 for the five algorithms under study. In addition, the related standard deviation values σ are also shown. It can be seen that the SFMDE outperformed the other algorithms in terms of minimum, maximum and mean fitness values.

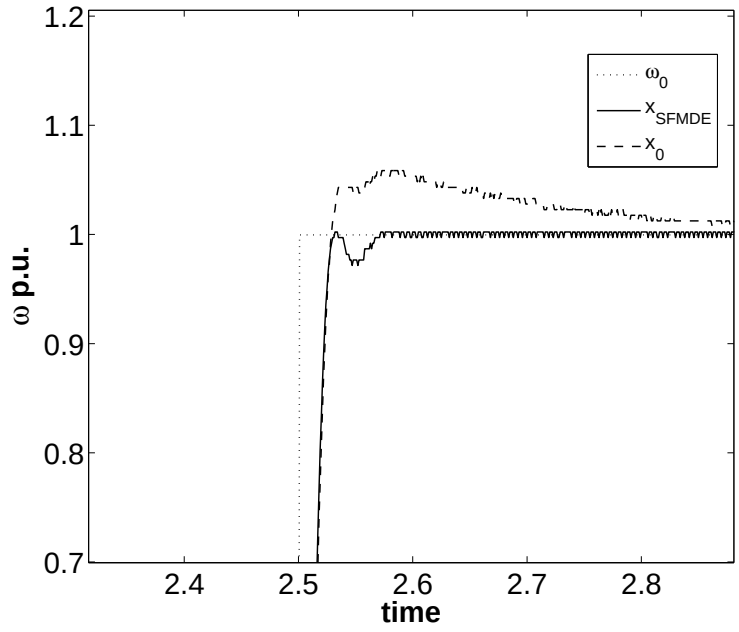

FIGURE 30 Speed step response

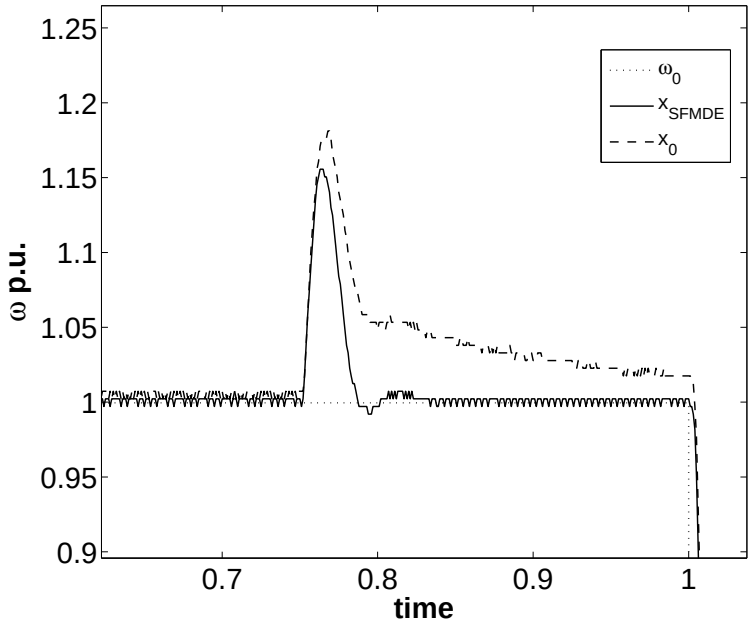

FIGURE 31 Load torque response

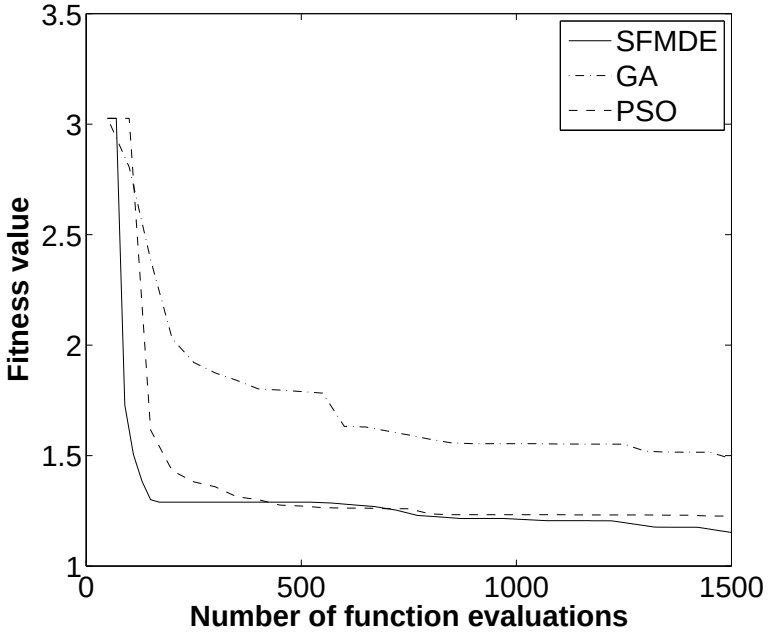

FIGURE 32 Performance comparison of SFMDE, GA, and PSO

Figure 32 and 33 show performance of the SFMDE with respect to the other algorithms under study. The performance is expressed in terms of average fitness value (over the 50 runs) dependant on the number of fitness evaluations. As shown in Figure 32 the GA, for this problem, performs significantly worse than the other algorithms. On the other hand DE, PSO, SDEA and SFMDE offer a good performance. The SFMDE compared to the other metaheuristics seems to be very promising since it converges to better performing solutions and it also has the best performance in terms of convergence velocity.

3.3 Application 2: Digital Filter Design for Defect Detection in Paper Production

For the same image processing application described in Section 2 the SFMDE has been applied. Performance of the SFMDE has been compared with a plain Differential Evolution (DE), an Evolution Strategy (ES), the Simulated Annealing (SA) and the Memetic Differential Evolution (MDE) proposed in Article **PIII** for solving the same problem.

The SFMDE has been run on a population of 100 individuals. The DE framework setting has been executed as shown in Section 2. The PSO has

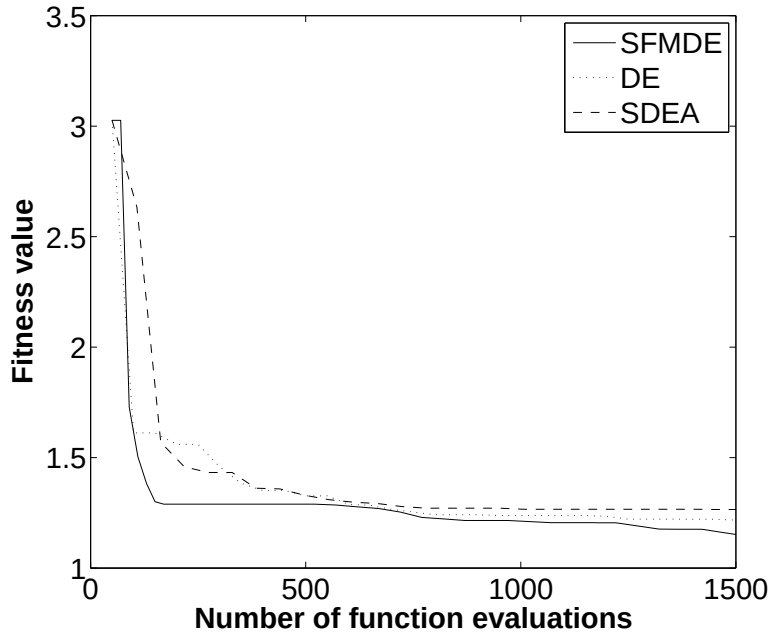

FIGURE 33 Performance comparison of SFMDE, DE and SDEA

been run for 800 fitness evaluations on a population of 20 individuals with a maximum velocity equal to 15.5. The local searchers have been run for 500 fitness evaluations; with reference to (20), $a = 0$ and $b = 0.6$ for the NMA and $a = 0$ and $b = 0.2$ for the RA respectively.

The DE has been executed with the same parameter setting used for the DE framework.

The ES works on a population size made up of $= 100$ individuals. As a standard ES, this ES does not contain any parent selection and, thus, it considers all populations as a population of parents [10]. A standard intermediate recombination has been chosen and the Gaussian mutation has been implemented resorting to the $1/5$ success rule [9]. Finally, a $(\mu + \lambda)$ strategy has been chosen.

SA has been run with an initial temperature equal to 1 and hyperbolical reduction of the temperature as suggested in [61].

MDE is another kind of memetic differential evolution framework made up of differential evolution and two highly exploitative local searchers adaptively coordinated by a fitness diversity control parameter. The same parameter setting shown in Article **PIII** has been performed.

Each algorithm has been run 30 times for 100000 fitness evaluations. The best solutions, overall, detected (parameters of the two filters Fil 1 and Fil 2

TABLE 6 Optimization results

	ES		SA		DE		MDE		SFMDE	
	Fil 1	Fil 2	Fil 1	Fil 2	Fil 1	Fil 2	Fil 1	Fil 2	Fil 1	Fil 2
weight	-5.893	-41.058	11.751	1.553	-1.507	94.995	21.479	91.568	82.701	-2.297
θ	4.330	0.408	0.643	1.613	3.530	1.519	2.135	4.658	3.104	2.919
ψ	4.552	5.099	0.819	3.318	4.912	3.122	6.042	3.153	3.104	4.725
σ_x	7.201	14.886	3.347	1.947	2.498	20	2.713	1.498	1.507	4.152
σ_y	1.008	10.905	6.023	7.128	1.512	20	3.609	20	20	20
λ	2.330	0.943	0.846	4.816	1.667	3.522	1.494	3.455	3.441	2.324
f_{min}	-1.273		-1.361		-1.487		-1.509		-1.521	
f_{mean}	-1.239		-0.978		-1.481		-1.493		-1.500	
f_{max}	-1.019		-0.732		-1.473		-1.339		-1.466	
σ	0.0723		0.116		0.00382		0.0335		0.0081	

respectively) by each algorithm are listed in Table 6. Moreover, the values of minimum, mean and maximum fitness and the related standard deviation values are also shown.

Figure 34 shows the functioning of the filters obtained by the optimization algorithms under analysis for an image belonging to the training set while Figure 35 shows the functioning for an image not belonging to the training set. From the upper left corner the source image, the label image, and the images filtered by means the application of the ES, SA, DE, MDE, SFMDE, are shown in order.

Figure 36 shows the average performance, over the 30 experiments for the algorithms under scrutiny.

Numerical results show that the algorithms based on a DE framework (DE, MDE, SFMDE) outperform the other algorithms considerably. The SFMDE slightly outperforms DE and MDE in terms of final solution detected. In addition, Figure 36 shows that the SFMDE has much better performance than all the other algorithms in terms of convergence velocity. It can be seen that, unlike the DE and MDE, the SFMDE reached, during all 30 runs, reasonably good results after only 30000 fitness evaluations. The high quality performance in terms of convergence velocity is a remarkable property of the SFMDE for this application since it allows a better match with the industrial demand of acquiring a quite efficiently tailored filter bank in a reasonably short training time.

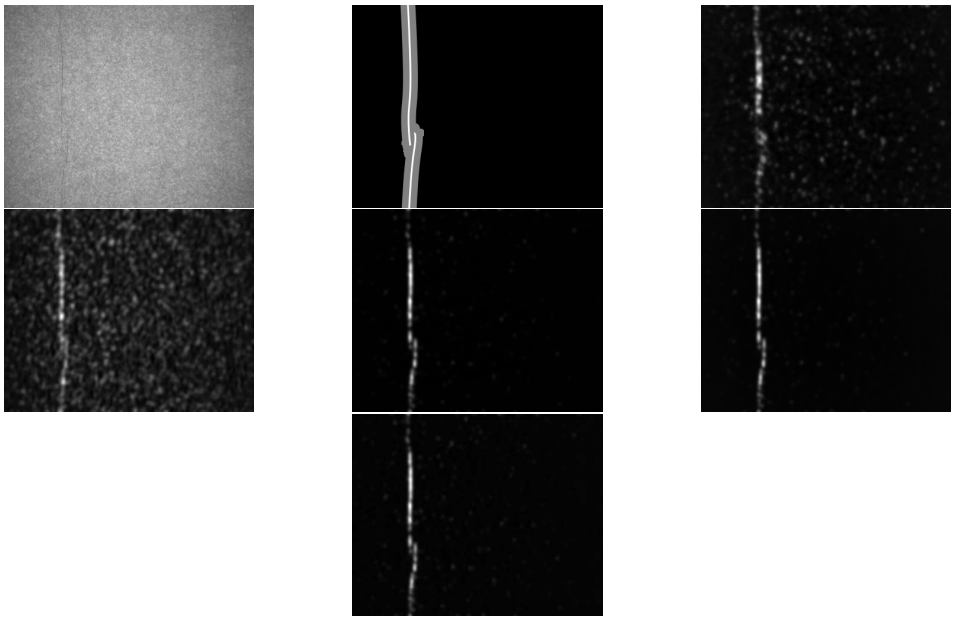

FIGURE 34 Image belonging to the training set.

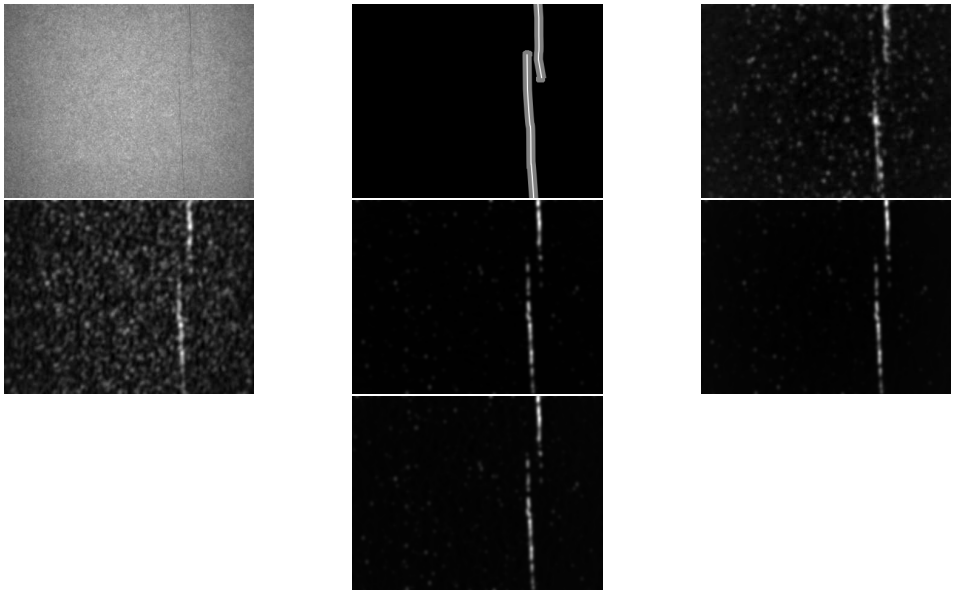

FIGURE 35 Image not belonging to the training set.

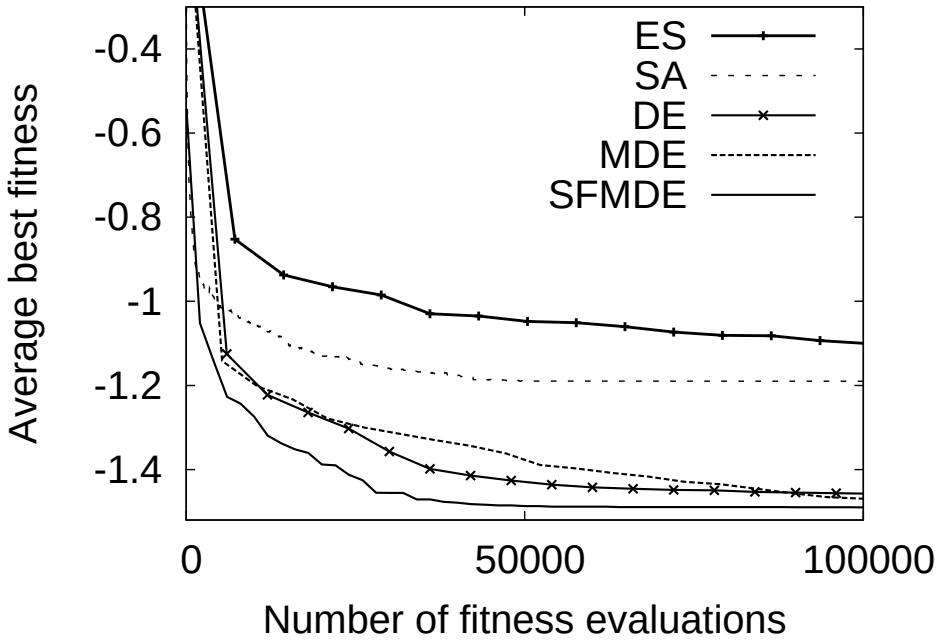

FIGURE 36 Algorithmic performance

3.4 Conclusion

The Super-Fit Memetic Differential Evolution (SFMDE), proposed in this paper, is composed of a differential evolution framework and a list of three local searchers having diverse features and employment within the algorithm. The general algorithmic philosophy of the SFMDE is to adaptively balance the improvement of the super-fit individual and the rest of the population. In order to pursue this aim, the local searchers assist the evolutionary framework in various stages of the evolution and on both super-fit and other individuals.

Numerical results on both applications show that the early hybridization with the PSO is beneficial for detecting a promising search direction within the DE framework. The adaptive employment of the other two local searchers also seems to be very beneficial in maintaining high convergence velocity performance and detecting a final value better than the other meta-heuristics used for a comparison.

YHTEENVETO (FINNISH SUMMARY)

Memeettiset algoritmit ovat evoluutiostrategioita ja paikallisia hakumenetelmiä yhdistäviä globaalin optimoinnin menetelmiä. Väitöskirjassa esitetään uusia, kehittyneitä memeettisiä algoritmeja sekä niiden tietokonetoteutuksia ja sovelluksia. Kehitetyissä menetelmissä paikallishakua koordinoidaan älykkäällä logiikalla, joka ottaa huomioon laskettujen likiratkaisujen joukon (populaation) monimuotoisuuden. Väitöskirjassa on esitetty useita uusia tapoja monimuotoisuuden arviointiin ja menetelmien adaptiivisuuteen sovellettaessa memeettisiä algoritmeja reaalimaailman ilmiöiden optimointiin. Esitetyille algoritmeille on yhteistä, että ne pyrkivät toimimaan älykkäästi, so. analysoimaan ajon aikana optimoinnin edistymistä ja tarvittaessa muuttamaan hakustrategioita, jotta optimointitehtävän ratkaisu saavutetaan tavoitellulla tarkkuudella.

Väitöskirja koostuu kahdeksasta julkaistua ja niitä käsittelevästä yhteenvedosta. Kaksi julkaisuista käsittelee memeettisten algoritmien soveltamista HIV-lääkityksen teoreettisen mallin optimisäätöön, yksi teollisuuden kuvankäsittelyongelmaan, yksi resurssien hakuun tietoliikenneverkoista. Kaksi artikkeleista koostuu esitettyjen adaptiivisten tekniikoiden teoreettisesta tarkastelusta. Kaksi viimeistä artikkelia käsittelee kohdefunktion kohinan vaikutuksen eliminoimista adaptiivisin tekniikoin. Kohdefunktion normaalijakautunutta kohinaa analysoidaan sähkömoottorin ohjausjärjestelmän optimoinnin tapauksessa. Viimeisessä artikkelissa analysoidaan kohinaa, joka on peräisin kahden evoluutioalgoritmin hierarkkisesta kytkennästä min-max ongelman ratkaisemiseksi. Kohina on luonteeltaan epätavanomaista, eikä sitä ole aikaisemmin analysoitu kirjallisuudessa.

Väitöskirjan yhteenveto-osassa esitellään lisäksi kaksi uutta algoritmia, jotka antavat viitteitä tutkimustyön jatkon suuntaviivoista. Molemmat algoritmit käyttävät evoluutiostrategiana differentiaalievoluutiota, mutta hyödyntävät eri adaptiivisia paikallishakutekniikoita. Uutuutena kahdeksaan väitöskirjaan sisällytettyihin artikkeleihin verrattuna on algoritmien erilainen tekniikka paikallishaun aktivoimiseksi. Uusissa algoritmeissa aktivointi tapahtuu todennäköisyyksiin perustuvalla kriteerillä kiinteän kynnysarvon sijaan. Alustavien testien perusteella paikallishaun koordinointi tällä logiikalla, kynnysarvoon perustuvan logiikan sijaan, antaa parempia tuloksia.

REFERENCES

[1] M. G. Norman and P. Moscato, "A competitive and cooperative approach to complex combinatorial search," Tech. Rep. 790, Caltech Concurrent Computation Program, 1989. expanded version published at the Proceedings of the 20th Informatics and Operations Research Meeting, Buenos Aires, Aug. 1991.

[2] P. Moscato, "On evolution,search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," Tech. Rep. 826, Caltech Concurrent Computation Program, 1989.

[3] R. Dawkins, *The selfish gene*. Oxford Press, 1976.

[4] W. E. Hart, N. Krasnogor, and J. E. Smith, "Memetic evolutionary algorithms," in *Recent Advances in Memetic Algorithms* (W. E. Hart, N. Krasnogor, and J. E. Smith, eds.), pp. 3–27, Springer, 2004.

[5] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computation*. Springer-Verlag, 2003.

[6] H. H. Rosenbrock, "An automatic method for findong the greatest or least value of a function," *The Computer Journal*, vol. 3, no. 3, pp. 175–184, 1960.

[7] R. Hooke and T. A. Jeeves, "Direct search solution of numerical and statistical problems," *Journal of the ACM*, vol. 8, pp. 212–229, Mar. 1961.

[8] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*. John Wiley, 1966.

[9] I. Rechemberg, *Evolutionstrategie: Optimierung Technisher Systeme nach prinzipien des Biologishen Evolution*. Fromman-Hozlboog Verlag, 1973.

[10] H. Schwefel, *Numerical Optimization of Computer Models*. Chichester, England, UK: Wiley, 1981.

[11] J. H. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[12] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[13] F. Neri, *Advanced Optimization Algorithms in Electrical Engineering*. PhD thesis, Technical University of Bari, Italy, 2007.

[14] G. E. Hinton and S. J. Nowlan, "How learning can guide evolution," *Complex Systems*, vol. 1, pp. 495–502, 1987.

[15] M. D. McKay, W. J. Conover, and R. J. Beckman, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, pp. 239–245, 1979.

[16] R. L. Iman, J. C. Helton, and J. E. Campbell, "An approach to sensitivity analysis of computer models, part 1. introduction, input variable selection and preliminary variable assessment," *Journal of Quality Technology*, vol. 13, no. 3, pp. 174–183, 1981.

[17] A. Caponio, G. L. Cascella, F. Neri, N. Salvatore, and M. Sumner, "A fast adaptive memetic algorithm for on-line and off-line control design of pmsm drives," *IEEE Transactions on System Man and Cybernetics-part B, special issue on Memetic Algorithms*, vol. 37, no. 1, pp. 28–41, 2007.

[18] W. E. Hart, *Adaptive Global Optimization with Local Search*. PhD thesis, University of California, San Diego, CA, USA, 1994.

[19] R. Unger and J. Moult, "A genetic algorithm for 3d ptotein folding simulations," in *Proceedings of the 5th International conference on Genetic Algorithms* (S. Forrest, ed.), pp. 581–588, Morgan Kauffman, 1993.

[20] B. Friesleben and P. Merz, "A genetic local search algorithm for solving the symmetric and asymmetric travelling salesman problem," in *Proceeding of the IEEE Conference on Evolutionary Computation*, pp. 616–621, 1996.

[21] P. Merz and B. Freisleben, "Memetic algorithms for the travelling salesman problem," *Complex System*, vol. 13, pp. 297–345, 2001.

[22] M. Lozano, F. Herrera, N. Krasnogor, and D. Molina, "Real-coded memetic algorithms with crossover hill-climbing," *Evolutionary Computation Journal*, vol. 12, pp. 273–302, 2004.

[23] N. Krasnogor, *Studies in the Theory and Design Space of Memetic Algorithms*. PhD thesis, University of West England, 2002.

[24] P. Moscato and F. Tinetti, "Blending heuristics with a population-based approach," tech. rep., Universidad Nacional de La Plata, Argentina, 1994.

[25] P. Moscato, "An introduction to population approaches for optimization and hierarchical objective functions: A discussion on the role of tabu search," *Annals of Operations Research*, vol. 41, no. 1-4, pp. 85–121, 1993.

[26] N. Krasnogor, "Toward robust memetic algorithms," in *Recent Advances in Memetic Algorithms* (W. E. Hart, N. Krasnogor, and J. E. Smith, eds.), pp. 185–207, Springer, 2004.

[27] N. Krasnogor, A. Aragón, and J. Pacheco, *Memetic Algorithms*, vol. 36 of *Operations Research/Computer Science Interfaces Series*, pp. 225–248. 2006.

[28] J. Tang, M. H. Lim, and Y. S. Ong, "Adaptation for parallel memetic algorithm based on population entropy," in *Proceedings of GECCO 2006*, pp. 575–582, Springer, 2006.

[29] P. Zou, Z. Zhou, G. Chen, and X. Yao, "A novel memetic algorithm with random multi-local-search: a case study of tsp," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 2, pp. 2335–2340, 2004.

[30] M. Wiering, "Memory-based memetic algorithms," in *Proceedings of the Thirteenth Belgian-Dutch Conference on Machine Learning* (A. Nowe, T. Lenaerts, and K. Steenhout, eds.), pp. 191–198, 2004.

[31] J. Tang, M. H. Lim, and Y. S. Ong, "Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems," *Soft Computing Journal*, vol. 11, pp. 873–888, July 2007.

[32] R. D. Smith, ed., *Roll and Web Defect Terminology*. TAPPI Press, 1995.

[33] A. K. Jain, *Fundamentals of Digital Image Processing*. Prentice Hall, 1989.

[34] C. Chang, "Contextual-based Hopfield neural network for medical image edge detection," *Optical Engineering (Bellingham, Washington)*, vol. 45, no. 3, pp. 37006–37006, 2006.

[35] G. Valli, R. Poli, S. Cagnoni, and G. Coppini, "Neural networks and prior knowledge help the segmentation of medical images," *Journal of Computing and Information Technology*, vol. 6, no. 2, pp. 117–133, 1998.

[36] R. Poli, "Genetic programming for feature detection and image segmentation," in *Evolutionary Computing*, no. 1143 in Lecture Notes in Computer Science, pp. 110–125, Springer-Verlag, Apr. 1996.

[37] S. Cagnoni, A. Dobrzeniecki, R. Poli, and J. Yanch, "Genetic-algorithm-based interactive segmentation of 3d medical images," *Image and Vision Computing Journal*, vol. 17, no. 12, pp. 881–896, 1999.

[38] B. Hernandez, G. Olague, R. Hammoud, L.Trujillo, and E. Romero, "Visual learning of texture descriptors for facial expression recognition in thermal imagery," *Computer Vision and Image Understanding*, vol. 106, pp. 258–269, May 2007.

[39] G. Olague, E. Romero, L. Trujillo, and B. Bhanu, "Multiclass object recognition based on texture linear genetic programming," in *Applications of Evolutionary Computing, Lecture Notes in Computer Science*, pp. 291–300, Springer, 2007.

[40] G. Olague, F. Fernandez, C. Pérez, and E. Lutton, "The infection algorithm: An artificial epidemic approach for dense stereo correspondence," *Artificial Life*, vol. 12, pp. 593–615, Oct. 2006. to appear.

[41] L. Trujillo and G. Olague, "Synthesis of interest point detectors through genetic programming," in *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation* (Keijzer, M. et al., ed.), vol. 1, pp. 887–894, ACM Press, Jul. 2006.

[42] C. Chan and G. Pang, "Fabric Defect Detection by Fourier Analysis," *IEEE Transactions on Industry Applications*, vol. 36, no. 5, p. 1267, 2000.

[43] S. Parker and J. Chan, "Dirt counting in pulp: An approach using image analysis methods," in *Signal and Image Processing (SIP 2002)*, 2002.

[44] J. Iivarinen, J. Pakkanen, and J. Rauhamaa, "A som-based system for web surface inspection," in *Machine Vision Applications in Industrial Inspection XII*, vol. 5303, pp. 178–187, SPIE, 2004.

[45] D. Dunn and W. Higgins, "Optimal Gabor filters for texture segmentation," *IEEE Transactions on Image Processing*, vol. 4, pp. 947–964, July 1995.

[46] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection using evolutionary gabor filter optimization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, pp. 125–137, June 2005.

[47] J. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters," vol. 2, no. 7, pp. 1160–1169, 1985.

[48] D. Tsa and S. Wu, "Automated Surface Inspection Using Gabor Filters," *The International Journal of Advanced Manufacturing Technology*, vol. 16, no. 7, pp. 474–482, 2000.

[49] T. Weldon, W. Higgins, and D. Dunn, "Efficient Gabor filter design for texture segmentation," *Pattern Recognition*, vol. 29, no. 12, pp. 2005–2015, 1996.

[50] Y. Ji, K. H. Chang, and C.-C. Hung, "Efficient edge detection and object segmentation using Gabor filters," in *ACM-SE 42: Proceedings of the 42nd annual Southeast regional conference*, pp. 454–459, ACM Press, 2004.

[51] J. Ilonen, J. Kämäräinen, and H. Kälviäinen, "Efficient computation of Gabor features," Research Report 100, Lappeenranta University of Technology, Department of Information Technology, 2005.

[52] A. Kumar and G. Pang, "Defect detection in textured materials using gabor filters," *IEEE Transactions on Industry Applications*, vol. 38, no. 2, pp. 425–440, 2002.

[53] R. Storn and K. Price, "Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces," Tech. Rep. TR-95-012, ICSI, 1995.

[54] K. V. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Springer, 2005.

[55] J. Lampinen and I. Zelinka, "On stagnation of the differential evolution algorithm," in *Proceedings of 6th International Mendel Conference on Soft Computing* (P. Oŝmera, ed.), pp. 76–83, 2000.

[56] Kaupe, F., Jr., "Algorithm 178: direct search," *Communications of the ACM*, vol. 6, pp. 313–314, June 1963.

[57] C. T. Kelley, *Iterative Methods of Optimization*, pp. 212–229. Philadelphia, USA: SIAM, 1999.

[58] H. H. Hoos and T. Stützle, *Stochastic Local Search Foundations and Applications*. Morgan Kaufmann / Elsevier, 2004.

[59] S. Kirkpatrick, C. D. J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, no. 220, pp. 671–680, 1983.

[60] V. Cerny, "A thermodynamical aprroach to the traveling salesman problem," *Journal of Optimization, theory and Application*, vol. 45, no. 1, pp. 41–51, 1985.

[61] H. Szu and R. Hartley, "Fast simulated annealing," *Physiscs Letters A*, vol. 122, pp. 157–162, 1987.

[62] R. Storn, "Designing nonstandard filters with differential evolution," *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 103–106, 2005.

[63] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: model, taxonomy, and design issues," *IEEE Transactions on Evolutionary Computation*, vol. 9, pp. 474–488, 2005.

[64] Y. S. Ong, M. H. Lim, N. Zhu, and K. W. Wong, "Classification of adaptive memetic algorithms: A comparative study," *IEEE Transactions On Systems, Man and Cybernetics - Part B*, vol. 36, no. 1, pp. 141–152, 2006.

[65] Y. S. Ong and A. J. Keane, "Meta-lamarkian learning in memetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 2, pp. 99–110, 2004.

[66] F. Neri, G. L. Cascella, N. Salvatore, A. V. Kononova, and G. Acciani, "Prudent-daring vs tolerant survivor selection schemes in control design of electric drives," in *Applications of Evolutionary Computing* (F. R. et al., ed.), vol. LNCS 3907, pp. 805–809, Springer, 2006.

[67] N. Balakrishnan and A. P. Basu, *The Exponential Distribution: Theory, Methods, and Applications*. Gordon and Breach, 1996.

[68] A. E. Eiben and J. E. Smith, "Hybrid evolutionary algorithms," in *Introduction to Evolutionary Computing, Hybridisation with other Techniques: Memetic Algorithms*, Slides of the Lecture Notes, Chapter 10, 2003.

[69] D. Whitley, "The genitor algorithm and selection pressure. why rank-based allocation of reproductive trials is best," in *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 116–121, Morgan Kaufmann, 1989.

[70] N. Karaboga and B. Cetinkaya, "Performance comparison of genetic and differential evolution algorithms for digital FIR filter design," in *Advances in Information Systems LNCS*, vol. 3261, pp. 482–488, Springer, 2004.

[71] I. L. Lopez Cruz, L. Van Willigenburg, and G. Van Straten, "Parameter control strategy in differential evolution algorithm for optimal control," in *Proceedings of the IASTED International Conference Artificial Intelligence and Soft Computing (ASC 2001)* (M. Hamza, ed.), pp. 211–216, ACTA Press, May 2001.

[72] J. Liu and J. Lampinen, "On setting the control parameter of the differential evolution algorithm," in *Proceedings of the 8th international Mendel conference on soft computing*, pp. 11–18, 2002.

[73] J. Liu and J. Lampinen, "Adaptive parameter control of differential evolution," in *Proceedings of the 8th international Mendel conference on soft computing*, pp. 19–26, 2002.

[74] J. L. J. Liu, "A fuzzy adaptive differential evolution algorithm," in *Proceedings of the 17th IEEE region 10 international conference on computer, communications, control and power engineering*, vol. I, pp. 606–611, 2002.

[75] J. L. J. Liu, "A fuzzy adaptive differential evolution algorithm," *Soft Computing - A Fusion of Foundations, Methodologies and Applications, Springer*, vol. 9, pp. 448–462, June 2005.

[76] M. M. Ali and A. Törn, "Population set based global optimization algorithms: Some modifications and numerical studies," *Computers and Operations Research, Elsevier*, no. 31, pp. 1703–1725, 2004.

[77] J. Tvrdík, "Differential evolution: Competitive setting of control parameters," in *Proceedings of the International Multiconference on Computer Science and Information Technology*, pp. 207–213, 2006.

[78] D. Zaharie, "Critical values for control parameters of differential evolution algorithm," in *Proceedings of 8th International Mendel Conference on Soft Computing* (R. Matuŝek and P. Oŝmera, eds.), pp. 62–67, 2002.

[79] K. Zielinskiand, P. Weitkemper, R. Laur, and K.-D. Kammeyer, "Parameter study for differential evolution using a power allocation problem including interference cancellation," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1857–1864, 2006.

[80] J.-P. Chiou and F.-S. Wang, "A hybrid method of differential evolution with application to optimal control problems of a bioprocess system," in *The 1998 IEEE International Conference on Evolutionary Computation Proceedings*, pp. 627–632, 1998.

[81] J.-P. Chiou and F.-S. Wang, "Hybrid method of evolutionary algorithms for static and dynamic optimization problems with application to a fed-batch fermentation process," *Computers and Chemical Engineering, Elsevier*, vol. 23, pp. 1277–1291, November 1999.

[82] R. Mydur, "Application of evolutionary algorithms and neural networks to electromagnetic inverse problems.," Master's thesis, Texas A and M University, Texas, USA, 2000.

[83] T. Rogalsky and R. W. Derksen, "Hybridization of differential evolution for aerodynamic design," in *Proceedings of the 8th Annual Conference of the Computational Fluid Dynamics Society of Canada*, pp. 729–736, June 2000.

[84] F.-S. Wang and H.-J. Jang, "Parameter estimation of a bioreaction model by hybrid differential evolution," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 1, pp. 410–417, 2000.

[85] T. Hendtlass, "A combined swarm differential evolution algorithm for optimization problems," in *Lecture Notes in Computer Science, Springer-Verlag*, vol. 2070, pp. 11–18, 2001.

[86] Y.-C. Lin, F.-S. Wang, and K.-S. Hwang, "A hybrid method of evolutionary algorithms for mixed-integer nonlinear optimization problems," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 3, pp. 2159–2166, July 1999.

[87] Y.-C. Lin, K.-S. Hwang, and F.-S. Wang, "Co-evolutionary hybrid differential evolution for mixed-integer optimization problems, taylor and francis," *Engineering Optimization*, vol. 33, no. 6, pp. 663–682, 2001.

[88] C.-T. Su and C.-S. Lee, "Network reconfiguration of distribution systems using improved mixed-integer hybrid differential evolution," *IEEE Transactions on Power Delivery*, vol. 18, pp. 1022–1027, July 2003.

[89] Y.-C. Lin, K.-S. Hwang, and F.-S. Wang, "Hybrid differential evolution with multiplier updating method for nonlinear constrained optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 1, pp. 872–877, May 2002.

[90] J.-P. Chiou, C.-F. Chang, and C.-T. Su, "Ant direction hybrid differential evolution for solving large capacitor placement problems," *IEEE Transactions on Power Systems*, vol. 19, pp. 1794–1800, November 2004.

[91] D. Neumann and H. X. de Araujo, "Hybrid differential evolution method for the mixed h2/h robust control problem under pole assignment," in *Proceedings of the 44th IEEE Conference on Decision and Control, and 2005 European Control Conference*, pp. 1319–1324, Dec.

[92] P. Kaelo and M. M. Ali, "Differential evolution algorithms using hybrid mutation," *Computational Optimization and Applications, Springer*, vol. 37, pp. 231–246, June 2007.

[93] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, pp. 39–43, 1995.

[94] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*, pp. 1942–1948, 1995.

[95] A. Nelder and R. Mead, "A simplex method for function optimization," *Computation Journal*, vol. 7, pp. 308–313, 1965.

[96] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence properties of the nelder-mead simplex method in low dimensions," *SIAM Journal on Optimization*, vol. 9, pp. 112–147, 1998.

[97] G. Birkho and S. M. Lane, *A Survey of Modern Algebra*. Macmillan, 1953.

[98] K. J. Åström and B. Wittenmark, *Adaptive Control, 2nd ed.* Prentice Hall, 1994.

[99] J.-J. Slotine and W. Li, *Applied Nonlinear Control*. Prentice Hall, 1990.

[100] H. M. F. Khorrami, P. Krishnamurthy, *Modeling and Adaptive Nonlinear Control of Electric Motors*. Springer, 2003.

[101] L. C. Jain and C. W. de Silva, *Intelligent Adap.tive Control: Industrial Applications*. CRC, 1998.

[102] K. J. Åström and T. Hägglund, "The future of PID control," *Control Engineering Practice*, vol. 9, no. 13, pp. 1163–1175, November 2001.

[103] H. Panagopoulos, K. J. Åström, and T. Hägglund, "Design of PID controllers based on constrained optimisation," *IEE Proceedings - Control Theory & Applications*, vol. 149, pp. 32–40, 2002.

[104] W. Dury, *Control Techniques Drives & Controls Handbook*. Institution Electrical Engineers, 2001.

[105] W. Leonhard, *Control of Electrical Drives, 2nd ed.* Springer, 2001.

[106] R. Krishnan, *Electronic Motor Drives: Modeling, Analysis and Control*. Prentice-Hall, 2001.

[107] F. C. M. M. P. Zanchetta, M. Sumner and E. Mininno, "On-line and off-line control design in power electronics and drives using genetic algorithms," in *Proceedings of 11th IEEE Conference on Artificial Intelligence Applications*, pp. 277–283, 1995.

[108] M. S. G. L. Cascella, N. Salvatore and L. Salvatore, "On-line simplex-genetic algorithm for self-commissioning of electric drives," in *Proceedings 11th European Power Electronics Conference*, pp. 277–283, 2005.

[109] K. J. L. Szklarski, A. Horodecki, *Electric Drive Systems Dynamics*. Elsevier Science Ltd, 1990.

[110] X. del Toro Garcia, F. Neri, G. L. Cascella, and N. Salvatore, "A surrogate assisted hooke-jeeves algorithm to optimize the control system of a pmsm drive," in *Proceedings of IEEE International Symposium on Industrial Electronics*, vol. 1, pp. 347–352, 2006.

ORIGINAL PAPERS

PI

AN ADAPTIVE EVOLUTIONARY ALGORITHM WITH INTELLIGENT MUTATION LOCAL SEARCHERS FOR DESIGNING MULTIDRUG THERAPIES FOR HIV

by

F. Neri, J. Toivanen and R. Mäkinen

Applied Intelligence, Special Issue on Computational Intelligence in Medicine and Biology, Volume 27, Issue 3, pages 219-235, December 2007

https://doi.org/10.1007/s10489-007-0069-8

PII

AN ADAPTIVE MULTIMEME ALGORITHM FOR DESIGNING HIV MULTIDRUG THERAPIES

by

F. Neri, J. Toivanen, G. L. Cascella and Y.-S. Ong

IEEE/ACM Transactions on Computational Biology and Bioinformatics, Special Issue on Computational Intelligence Approaches in Computational Biology and Bioinformatics, Volume 4, Issue 2, pages 264-278, April 2007

https://www.researchgate.net/deref/http%3A%2F%2Fdx.doi.org%2F10.1109%2FTCBB.2007.070202

PIII

A MEMETIC DIFFERENTIAL EVOLUTION IN FILTER DESIGN FOR DEFECT DETECTION IN PAPER PRODUCTION

by

V. Tirronen, F. Neri, T. Kärkkäinen, K. Majava and T. Rossi

Applications of Evolutionary Computing, Lectures Notes in Computer Science, Volume 4448, pages 320-329, (EvoIASP Best Paper Nomination), April 2007

https://doi.org/10.1007/978-3-540-71805-5_35

PIV

FITNESS DIVERSITY BASED ADAPTATION IN MULTIMEME ALGORITHMS: A COMPARATIVE STUDY

by

F. Neri, V. Tirronen, T. Kärkkäinen and T. Rossi

Proceedings of the IEEE Congress on Evolutionary Computation, Special Session on Memetic Algorithms, Singapore, pages 2374-2381, September 2007

https://www.researchgate.net/deref/http%3A%2F%2Fdx.doi.org%2F10.1109%2FCEC.2007.4424768

PV

A FAST RANDOMIZED MEMETIC ALGORITHM FOR HIGHLY MULTIMODAL PROBLEMS

by

V. Tirronen and F. Neri

to appear on Evolutionary Methods in Design Optimization and Control, P. Neittaanmäki, J. Periaux, T. Tuovinen eds.

PVI

AN ADAPTIVE GLOBAL-LOCAL MEMETIC ALGORITHM TO DISCOVER RESOURCES IN P2P NETWORKS

by

F. Neri, N. Kotilainen and M. Vapa,

Applications of Evolutionary Computing, Lectures Notes in Computer Science, Volume 4448, pages 61-70, (EvoCOMNET Best Paper Nomination), April 2007

https://doi.org/10.1007/978-3-540-71805-5_7

PVII

AN ADAPTIVE PRUDENT-DARING EVOLUTIONARY ALGORITHM FOR NOISE HANDLING IN ON-LINE PMSM DRIVE DESIGN

by

F. Neri, G. L. Cascella, N. Salvatore and S. Stasi

Proceedings of the IEEE Congress on Evolutionary Computation, Special Session on Evolutionary Computation in Dynamic and Uncertain Environments, Singapore, pages 584-591, September 2007

https://doi.org/10.1109/CEC.2007.4424523

PVIII

HIERARCHICAL EVOLUTIONARY ALGORITHMS AND NOISE COMPENSATION VIA ADAPTATION

by

F. Neri and R. Mäkinen

Evolutionary Computation in Dynamic and Uncertain Environments, S. Yang, Y-S. Ong, Y Jin eds., Studies in Computational Intelligence, pages 345-369, (book chapter), April 2007

https://doi.org/10.1007/978-3-540-49774-5_15