Soft Comput (2011) 15:1427–1448
DOI 10.1007/s00500-010-0681-0

ORIGINAL PAPER

# Optimization in dynamic environments: a survey on problems, methods and measures

**Carlos Cruz · Juan R. González · David A. Pelta**

Published online: 24 December 2010


**Abstract** This paper provides a survey of the research done on optimization in dynamic environments over the past decade. We show an analysis of the most commonly used problems, methods and measures together with the newer approaches and trends, as well as their interrelations and common ideas. The survey is supported by a public web repository, located at http://www.dynamic-optimization.org where the collected bibliography is manually organized and tagged according to different categories.

## 1 Introduction

Telecommunications, infrastructures, services and, in general, all the aspects concerning the information society, give raise to a set of complex problems where several objectives may need to be satisfied, that may vary with time, or where uncertainty of diverse types may exist in the values of the variables, coefficients, or even in the objectives. Also, alternatives that were bad in the past can be good nowadays or vice versa; criteria that were important before become irrelevant now, etc., and moving in this dynamic scenarios is a challenge. The development of solving strategies in the area of intelligent systems that may work in such dynamic and imprecise scenarios raises new challenges that are being addressed from different points of view by the international scientific community.

Many problems in this context can be modeled as dynamic optimization problems (DOPs) in which some elements of the underlying model change over the course of the optimization. Here, we will consider DOP defined as follows:

$$\text{DOP} = \left\{ \begin{array}{ll} \text{optimize} & f(x,t) \\ \text{s.t.} & x \in F(t) \subseteq S, t \in T \end{array} \right\}$$

where:

- $S \in R^n$, $S$ is the search space.
- $t$ is the time.
- $f : S \times T \rightarrow R$, is the objective function that assigns a numerical value ($f(x,t) \in R$) to each possible solution ($x \in S$) at time $t$.
- $F(t)$, is the set of feasible solutions $x \in F(t) \subseteq S$ at time $t$.

In other words, a DOP is a problem where the objective function or the restrictions change with time. The simplest method for solving these problems is ignoring dynamics, considering each change as the arrival of a new optimization problem and re-optimizing, but it is often impractical, as illustrated in Branke (2001). Therefore, the goal of the methods for dealing with DOPs is no longer to locate a stationary optimal solution, but to track its movement through the solution and time spaces as closely as possible. Classic algorithms have been adapted to cope with such dynamic scenarios enhancing their ability for tracking moving optima.

Since the first known reference to evolutionary algorithms (EAs) applied to dynamic optimization by Fogel et al. (1966), and the paper by Goldberg and Smith (1987)

C. Cruz · J. R. González · D. A. Pelta (✉)
Department of Computer Science and AI, Models of Decision and Optimization Research Group, Information and Communication Technologies Research Centre (CITIC-UGR), University of Granada, 18071 Granada, Spain
e-mail: dpelta@decsai.ugr.es

C. Cruz
e-mail: carloscruz@decsai.ugr.es

J. R. González
e-mail: jrgonzalez@decsai.ugr.es

almost 20 years later, EAs have been the most common approach used to solve this kind of problems. Recently, other optimization techniques have been adapted to dynamic environments: particle swarm optimization, cooperative strategies (Pelta et al. 2009b), immune-based algorithms (Trojanowski and Wierzchon 2009), stochastic diffusion search (Meyer et al. 2006), ant colony optimization (Guntsch et al. 2001), and so on.

The topic of dynamic optimization was partly reviewed in the past. For example, it was covered in the books by Branke (2001), Weicker (2003), Morrison (2004), Yang et al. (2007a, b) and in the survey by Jin and Branke (2005). These works were mainly related to Evolutionary Optimization. Some special issues were devoted to the topic, considering other kind of methods, like Branke (2005), Branke and Jin (2006a, b), and Yang et al. (2006).

A reference website[1] on the application of evolutionary algorithms to DOP was maintained by Branke; other methods were later added, but the site seems to be inactive since 2007.

In our opinion, the research field is getting mature and it becomes necessary to stop and carefully look at what (and how) has been done in the past decade in order to avoid "re-inventing the wheel", establish good research practices regarding for example proper evaluations of methods, and detect opportunities to go beyond "artificial" problems to real-world applications.

We claim that this work can be considered as a step towards these goals and we focus on two specific aims: (1) to provide an overview of the related works on dynamic optimization problems on the last decade; (2) to present a new repository about this topic that goes beyond a simple collection of bibliographic entries, providing an organized and "curated" view of the field.

The contribution is organized as follows: Sect. 2 describes the web repository and the process to classify the references of the papers collected. Section 3 describes the most used dynamic optimization problems. Section 4 describes the methods used to tackle DOPs. Then, Sect. 5 explains the measures and tests that have been used to assess the performance of a single method or to analyze and compare the results of several different methods. Finally, in Sect. 6 a brief summary and some suggestions for future research are provided.

## 2 Bibliography search and repository details

The first step in constructing the review was to define what to search, where the search should be conducted and how the search should be defined (search terms).

[1] http://www.aifb.uni-karlsruhe.de/~jbr/EvoDOP.

The search was conducted using the engines described in Table 1, using terms like "dynamic optimization", "optimization AND uncertain environments", "dynamic scenarios", "metaheuristics AND dynamic optimization", "metaheuristics AND time varying", etc. First, just journal articles were considered, but as the number of results was small, we decided to extend the search to consider book chapters (from books published by prestigious publishing companies) and conference papers (from respected conferences). All the process was done during the last semester of 2009. It is important to notice that some of the search engines provide results considering also "In press" or "Online First" articles, so the publication year appearing in the website may not match the "physical" publication.

Then, a carefully hand-made categorization along several dimensions was performed for each selected reference. The categories were type of publication, dynamism, methods, models, performance measures, and applications. Their meaning and potential values for each category are described below:

- *type of publication*: book, book chapter, journal article, conference paper, thesis, etc.
- *type of dynamism*: where the dynamism is present in the problem formulation (objective function, and/or restrictions).
- *methods*: evolutionary algorithms, ant colony optimization, cooperative strategies, etc.
- *performance measures*: which measures were used to evaluate the algorithms.
- *applications*: which problem is solved by the proposed method.
- publication year.

The repository has filters that allow searching information using the above advanced classification with an appropriate search form. Data such as title, authors, abstract, keywords, etc., and links to the paper are also shown. Besides, the site have a section dedicated to additional Resources and Links where relevant information is being included, like other researchers, software, problems, congresses, and competitions. Figure 1 shows the number of publications collected and the number of visits in the past 15 months. Most of the visits came from Europe and

**Table 1** Search engines where the bibliography was collected

| Engine | URL | Publication type |
|---|---|---|
| Scopus | http://www.scopus.com | Journal, book |
| Science Direct | http://www.sciencedirect.com | Journal, book |
| IEEExplore | http://ieeexplore.ieee.org | Journal, conferences |
| CiteSeerX | http://citeseerx.ist.psu.edu | Conferences |
| Google Scholar | http://scholar.google.com | Journal, conferences |

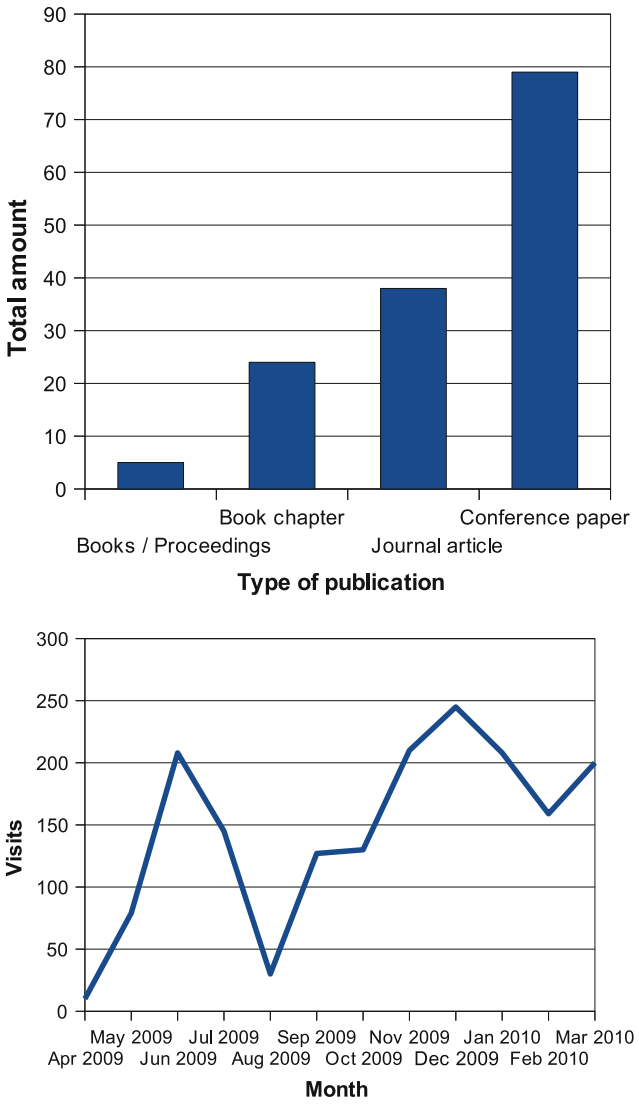


**Fig. 1** Number of contributions available in the repository (*top*) and number of visits from April, 2009 to March, 2010 (*bottom*)

that is also the reason for the noticeable decrease in August 2009, which is summer holidays time in most European universities.

We should also remark that the repository has a wider scope, including also information about "uncertainty". Although one may argue that dynamism represents a type of uncertainty, the meaning here is more related with imprecision or noise that can be managed with probabilities, fuzzy sets, and systems techniques, etc. This analysis is out of the scope of this review.

## 3 Dynamic optimization problems

The review performed allows to confirm that most of the research done up to this date on DOPs is based on synthetic or test problems, where the degree of dynamism and the complexity of the objective function is controllable. Table 2 shows a summary of the main problems in dynamic optimization and their references.

Many of these synthetic problems can be understood as DOPs generators that, departing from a predefined fitness landscape, allow to control the severity, the correlation or the type of the changes as well as to use different function types in some cases.

From the search results, it can be observed that one of the most widely used synthetic problems nowadays is still the *Moving Peaks Benchmark* (MPB) (Branke 1999). The idea is to have an artificial multi-dimensional landscape consisting of several peaks, where the height, width, and position of each peak is altered slightly every time a change in the environment occurs.

As many of the artificial problem generators rely on pretty similar ideas, we include here a complete description.

The cost function for $n$ dimensions and $m$ peaks has the following form:

$$F(\vec{x}, t) = \max_{i=1,\ldots,m} \frac{H_i(t)}{1 + W_i(t) \sum_{j=1}^{n} (x_j - X_{ij}(t))^2}$$

where $\vec{x} \in \Re^n$ is a particular solution, $\vec{X} \in \Re^{m \times n}$ is the set of $m$ peaks locations (we need $n$ values for locating a peak) and $\vec{H}, \vec{W} \in \Re^m$ are vectors storing the height and width of every peak. Figure 2 shows a scenario with $n = 1$ and $m = 5$.

The coordinates, the height $H_i$ and the width $W_i$ of each peak are randomly initialized. Then, at certain time steps, the height and width of every peak are changed by adding a random Gaussian variable multiplied by a specific "severity" factor ($h_{\text{sev}}, w_{\text{sev}}$, respectively). The location of every peak is moved by a vector $\mathbf{v}$ of fixed length $s$ in a random direction for $\alpha = 0$ or a direction depending on the previous direction for $\alpha > 0$. Thus $\alpha$ is a correlation coefficient allowing to control whether the changes exhibit a trend or not.

More formally, given $\sigma \in N(0, 1)$ a change can be described as

$$H_i(t) = H_i(t-1) + h_{\text{sev}} \times \sigma$$
$$W_i(t) = W_i(t-1) + w_{\text{sev}} \times \sigma$$
$$\mathbf{X}_i(t) = \mathbf{X}_i(t) + \mathbf{v}$$

The complexity of the problem can be augmented by increasing the number of dimensions or the number of peaks and by overlaying the whole landscape with a high-frequency noise.

Departing from this idea, a natural way for generating DOPs has been to introduce environment changes on a known stationary problem. In this sense, interesting benchmarks are obtained when the "peak" function from

**Table 2** Dynamic optimization problems (DOPs) and corresponding references

| Problem | References |
|---|---|
| Synthetic dynamic problems | DF1 generator (cones) (Eriksson and Olsson 2002; Esquivel and Coello Coello 2004, 2006; Morrison 2004; Peng and Reynolds 2004; Saleem and Reynolds 2000) |
| | Dynamic Ackley function (Schönemann 2004, 2007) |
| | Dynamic bit-matching (Jin and Branke 2005; Mori and Kita 2000a, b; Stanhope and Daida 1999) |
| | Dynamic deceptive functions (Tinós and Yang 2007b; Wang et al. 2009a, b; Yang 2003, 2006b, 2007; Yang and Tinós 2007a, b; Yang and Yao 2005, 2008) |
| | Dynamic knapsack problem (Branke et al. 2006a, b; Jin and Branke 2005; Karaman et al. 2005; Rohlfshagen and Yao 2009a, b; Simões and Costa 2003; Wang et al. 2009b; Yang 2008; Yang and Tinós 2007a, b; Yang and Yao 2005) |
| | Dynamic Onemax function (Droste 2003; Fernandes et al. 2008; Wang et al. 2009a; Yang 2003, 2005, 2007, 2008; Yang and Tinós 2007a, b; Yang and Yao 2008) |
| | Dynamic plateau functions (Wang et al. 2009a; Yang 2006b, 2008) |
| | Dynamic problem generator (Jin and Sendhoff 2004; Li and Yang 2008a; Morrison and De Jong 1999; Morrison 2004; Rand and Riolo 2005, 2006; Rohlfshagen and Yao 2009a, b; Tinos and Yang 2007c; Trojanowski and Wierzchon 2009; Ursem et al. 2002; Yang and Yao 2005) |
| | Dynamic Rastrigin function (Du and Li 2008; Olivetti de França et al. 2005; Schönemann 2004, 2007; Shi and Eberhart 2001; Tinos and Yang 1823; Venayagamoorthy 2004) |
| | Dynamic Rosenbrock function (Hu and Eberhart 2002; Olivetti de França et al. 2005; Shi and Eberhart 2001; Venayagamoorthy 2004) |
| | Dynamic royal road function (Fernandes et al. 2007, 2008; Tinós and Yang 2007b; Wang et al. 2009a, b; Yang 2003, 2007; Yang and Tinós 2007a, b; Yang and Yao 2005) |
| | Dynamic scheduling (Aydin and Öztemel 2000; Hart and Ross 1999; Mattfeld and Bierwirth 2004) |
| | Dynamic sphere (Arnold and Beyer 2002, 2006; Boumaza 2005; Olivetti de França et al. 2005; Schönemann 2004, 2007) |
| | Dynamic vehicle routing problem (DVRP) (Hanshar and Ombuki-Berman 2007; Montemanni et al. 2003; Pankratz 2005) |
| | Dynamic Griewank function (Olivetti de França et al. 2005; Shi and Eberhart 2001; Venayagamoorthy 2004; Wineberg and Oppacher 2000) |
| | Moving parabola (Abbass et al. 2004; Eberhart and Shi 2001; Hu and Eberhart 2002) |
| | Moving peaks benchmark (MPB) (Ayvaz et al. 2006; Blackwell and Branke 2004, 2006; Blackwell 2003, 2005, 2007; Branke and Schmeck 2003; Branke et al. 2000; Bui et al. 2005a, b; Du and Li 2008; Eriksson and Olsson 2004; Janson and Middendorf 2004; Jin and Branke 2005; Kramer and Gallagher 2003; Li and Yang 2008b; Li et al. 2006; Lung and Dumitrescu 2009; Mendes and Mohais 2005; Meyer et al. 2006; Moser and Hendtlass 2007; Novoa et al. 2009; Parrott and Li 2004; Pelta et al. 2009a, b; Ronnewinkel and Martinetz 2001; Trojanowski and Wierzchon 2009; Ursem 2000; Zou et al. 2004) |
| | Shaky ladder hyperplane-defined functions (SL-HDF) (Rand and Riolo 2005, 2006) |
| | Time-linkage numerical problems (Bosman 2007, 2005) |
| | Trap function based synthetic problems (Abbass et al. 2004; Laredo et al. 2008) |
| | XOR-based synthetic dynamic problems (Rohlfshagen et al. 2009a, b; Yang 2003, 2005, 2006a, b) |
| | Other synthetic dynamic problems (Branke 1999; Carlisle and Dozier 2000; Dam et al. 2007; Fernandes et al. 2007; Ghosh and Mühlenbein 2004; Guntsch et al. 2001; Mori and Kita 2000a, b; Mori et al. 2000a, b; Weicker and Weicker 1999; Woldesenbet and Yen 2009; Yan et al. 2004; Yen et al. 2001) |
| Real world applications | Aerospace design (Mack et al. 2007) |
| | Car distribution system (Michalewicz et al. 2007) |
| | Evolutionary robotics (Tinós and Yang 2007a) |
| | Financial optimization problems (Tezuka et al. 2007) |
| | Mobile ad hoc networks (Yang et al. 2010) |
| | Modeling of ship trajectory (Smierzchalski and Michalewicz 2000) |
| | Path planning (Elshamli et al. 2004; Michalewicz et al. 2007) |
| | Pollution control (Michalewicz et al. 2007) |
| | Pose problem (Rossi et al. 2008) |
| | Rober problem (Tumer and Agogino 2007) |
| | Robust design (Fan et al. 2007; Goh and Tan 2007; Handa et al. 2007; Hu et al. 2007; Ling et al. 2007; Rocco and Salazar 2007) |
| | Salting route optimization (Handa et al. 2007) |
| | Structural optimization (Neri and Mäkinen 2007) |
| | Varied-line-spacing holographic grating (VLSHG) (Ling et al. 2007) |
| | Wireless sensor networks (Quintão et al. 2007) |

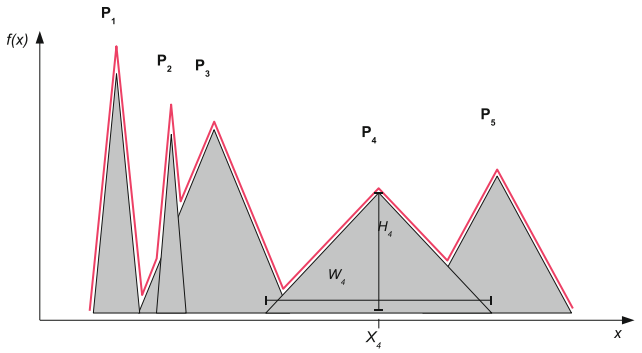


**Fig. 2** An example of the error landscape created with the moving peaks benchmark

MPB is replaced by "cones" [as in the DF1 generator from Morrison and De Jong (1999) and Morrison (2004)] or more complex functions like Sphere, Griewank, Rastrigin, etc. that are typical test problems in continuous optimization. Dynamic versions of combinatorial problems also exist, like dynamic knapsack problem (Branke et al. 2006a, b; Dasgupta and Mcgregor 1992; Goldberg and Smith 1987; Rohlfshagen and Yao 2009a, b), moving parabola (Angeline 1997), dynamic bit-matching (Droste 2003; Stanhope and Daida 1999), or dynamic scheduling problems (Mattfeld and Bierwirth 2004).

The way the dynamism is included into a stationary problem is well exemplified in Yang (2003, 2005). The idea is to depart from a binary-encoded stationary function $f(\mathbf{x})$ $(\mathbf{x} \in \{0,1\}^l)$ and use a bitwise exclusive-or (XOR) operator in the fitness calculation. This technique was tested on One-Max problem and plateau, royal road, and deceptive functions.

Assuming that the environment changes every $\tau$ generations, for each environmental period $k$ the XOR-ing mask $\mathbf{M}(k)$ is incrementally generated as follows:

$$\mathbf{M}(k) = \mathbf{M}(k-1) \oplus \mathbf{T}(k)$$

where $\oplus$ is the bitwise exclusive-or (XOR) operator (i.e. $1 \oplus 1 = 0$, $1 \oplus 0 = 1$, $0 \oplus 0 = 0$) and $\mathbf{T}(k)$ is an intermediary binary template randomly created with $\rho \times l$ ones for environmental period $k$ (cyclic or cyclic with noise changes were also considered). For the first period $k = 1$, $\mathbf{M}(1)$ is set to a zero vector. Then, the population at generation $t$ is evaluated with the formula

$$f(\mathbf{x}, t) = f(\mathbf{x} \oplus \mathbf{M}(k))$$

where $k = \lceil t/\rho \rceil$ is the environmental period index. With the XOR generator defined in this way, the parameter $\tau$ controls the speed of change while $\rho \in (0.0, 1.0)$ controls the severity of environmental changes. Bigger values of $\rho$ mean more severe environmental changes. The above procedure allows to change the fitness landscape while keeping certain properties of the original landscape, like the total number of optima and their values despite their locations are shifted.

Tinos and Yang (2007c) extend the XOR technique to real-valued optimization problems and applied it to a problem where the fitness function uses Sphere model and Generalized Griewank function. Also, Rohlfshagen et al. (2009a, b) using the XOR technique analyzed the effects of the magnitude and frequency of changes in a dynamic evolutionary optimization on a set of artificially designed pseudo-boolean functions.

Rohlfshagen and Yao (2009a, b) recently proposed a dynamic combinatorial optimization benchmark based on 0-1 knapsack problem instances that are generated using a small set of real-valued parameters. These parameters are varied over time by some set of differential equations so it is possible to model different types of transitions by controlling the shape and degree of interactions between the trajectories of the parameters.

Another interesting idea was proposed by Yin and Sendhoff (2004), where they suggest to construct dynamic optimization test problems using multi-objective optimization (MOO) concepts. A typical MOO problem consists on minimizing a set of objectives:

$$\min_{x \in S}(f_1(x), f_2(x), \ldots, f_m(x))$$

where $x$ is a candidate solution, $m$ is the number of objectives and $S$ is the search space. One "simple" way to deal with multiobjective problems, is to aggregate the individual objectives as

$$\min F(x) = \sum_{i=1}^{m} w_i \times f_i(x)$$

where $0 \leq w_i \leq 1,\ i = 1, \ldots, m, \sum w_i = 1$.

Using this idea, one can change the weights $w_i$ dynamically to construct dynamic single objective and multi-objective test problems systematically.

Finally, Li and Yang (2008a) proposed the *Generalized Dynamic Benchmark Generator*. It was designed with the idea of constructing dynamic environments across binary, real, and combinatorial solution spaces. To do this the authors defined a DOP as

$$F = f(x, \theta, t)$$

where $F$ is the optimization problem, $f$ is the cost function, $x$ is a feasible solution in the solution set $X$, $t$ is the real-world time, and $\theta$ are the system control parameters, which determine the solution distribution in the fitness landscape. The search space is constructed from a composition of base functions and the dynamism is obtained by tuning the system control parameters. There are six change types of this system: small step change, large step change, random change, chaotic change, recurrent change, and recurrent change with noise. Thus, GDBG can present different dynamic properties

using these change types. The authors also propose, for real coded landscapes, to rotate the landscape instead of shifting the positions of base functions. It should also be remarked that GDBG was used as a benchmark in CEC'2009 competition on dynamic optimization.

### 3.1 Real-world dynamic problems

Some interesting works about real-world problems showing dynamic environment behavior have started to appear in the very last years. The lower part of Table 2 shows a summary of the main "real world" problems in dynamic optimization and their references.

For example Mack et al. (2007) solve an aerospace design problem applying genetic algorithms (NSGA-II) and surrogate modeling. They tried the response surface-based multi-objective optimization of a radial turbine for an expander cycle-type liquid rocket engine. Three problems, a pollution control system, path planning of ships, and a car distribution system for "off-lease" cars, were solved by Michalewicz et al. (2007) using Adaptive Business Intelligence. Risk in financial optimization problems was studied by Tezuka et al. (2007) using a genetic algorithm.

There are also some works on dynamic vehicle routing problem made by different researchers (Hanshar and Ombuki-Berman 2002; Pankratz 2005) that could be applied to real-world problems modeled as instances of this problem. Rossi et al. (2008) worked on the pose problem where there is a need to know the pose of a moving object using evolutionary algorithms and Kalman filters. Dam et al. (2007) solved the task of performing evolutionary online data mining in a dynamic environment applying XCS, a genetics-based learning classifier system (GBLCS). Tumer and Agogino (2007) evolved evaluation functions for a multi rover system in a dynamic and noisy environment using evolutionary algorithms. Handa et al. (2007) optimized salting truck routes based on real-world data from the South Gloucestershire council in England, with dynamic optimization using a memetic algorithm that was able to achieve a 10% improvement over the available solutions in terms of the distances traveled by the trucks.

Neri and Mäkinen (2007) solved two structural optimization problems using evolutionary algorithms: the optimal design of a grounding grid and the optimal design of an elastic structure. Ling et al. (2007) studied the design of recording optics of varied-line-spacing holographic grating (VLSHG) in the National Synchrotron Radiation Laboratory by applying an standard crowding genetic algorithm. Quintão et al. (2007) controlled energy consumption and quality of service aspects on Wireless Sensor Networks applying Dijkstra Shortest Path and Prim Minimum Spanning Tree algorithms together with evolutionary algorithms, and mixed linear integer programming. More recently, Yang et al. (2010) tackled a dynamic shortest path routing problem on mobile ad hoc networks (MANETs) using genetic algorithms.

## 4 Methods for solving DOPs

As it was stated before, the goal of the methods for dealing with DOPs is no longer to locate a stationary optimal solution, but to track their progression through the space and time as closely as possible.

It is somehow obvious to think that this "tracking" would be better performed using a group of solutions/searchers than a single individual, and this idea is also reflected in the kind of methods that are being applied to deal with DOPs. The so-called population-based methods, like evolutionary algorithms or particle swarm optimization, showed themselves as clear alternatives for this domain.

The methods reviewed and categorized on the web repository are presented in Table 3. Observing the results, it is clear that the class of *Evolutionary Techniques* and their variants have been the most widely used methods to solve these DOPs. Nevertheless, other approaches are gaining more and more attention in the past years.

If we look at the methods used as a whole, we can see that despite using quite different motivations there are several important frequently used components or mechanisms that can be considered as *building blocks* for the majority of dynamic optimization methods. These components are needed to properly adapt the classic methods to dynamic scenarios, tackling the diversity loss and outdated memory problems. According to classic categories by Blackwell and Branke (2006) these building blocks can be grouped as

*Maintenance of diversity* It is essential to keep certain level of diversity as it is expected that a diverse population can adapt more easily to changes than a fully converged one. Random Immigrants GA (RIGA) (Grefenstette 1992; Tinós and Yang 2007b; Yang 2008) is an example of this strategy. Every generation RIGA replaces part of the population by randomly generated individuals. This introduces new genetic material in every time step and avoids the convergence of the whole population to a narrow region of the search space.

For PSO this may be achieved by integrating a sort of repulsion. There are several techniques such as charged PSO (CPSO) (Blackwell 2003) based on mutually repelling particles that orbit chaotically a nucleus of neutral particles, quantum particles (Blackwell and Branke 2004) based on a quantum rather than classical picture of an atom, the replacement of global by local neighborhoods

**Table 3** Methods applied to dynamic optimization and references to papers where they are used

| Method | Abbr. | References |
|---|---|---|
| Ant colony optimization | ACO | (Fernandes et al. 2007, 2008; Guntsch et al. 2001; Hanshar and Ombuki-Berman 2007; Montemanni et al. 2003) |
| Cooperative strategies | CS | (González et al. 2010; Lung and Dumitrescu 2007, 2009; Pelta et al. 2009b) |
| Cultural algorithms | CA | (Peng and Reynolds 2004; Saleem and Reynolds 2000) |
| Evolutionary techniques | | |
| Evolution strategies | ES | (Arnold and Beyer 2002, 2006; Boumaza 2005; Mendes and Mohais 2005; Rocco and Salazar 2007; Rohlfshagen and Yao 2009a, b; Rossi et al. 2008; Weicker and Weicker 1999) |
| Evolutionary algorithms | EA | (Barrico and Antunes 2007; Bosman 2007, 2005; Branke 1999; Branke and Schmeck 2003; Branke et al. 2006a, b; Droste 2003; Eriksson and Olsson 2002, 2004; Goh and Tan 2007; Handa et al. 2007; Jin and Branke 2005; Jin and Sendhoff 2004; Karaman et al. 2005; Li and Yang 2008a, b; Lim et al. 2007; Morrison 2004; Neri and Mäkinen 2007; Quintão et al. 2007; Richter 2005; Richter and Yang 2009; Rocco and Salazar 2007; Rohlfshagen et al. 2009a, b; Rossi et al. 2008; Smierzchalski and Michalewicz 2000; Tenne and Armfield 2007; Tumer and Agogino 2007; Ursem et al. 2002; Woldesenbet and Yen 2009; Yan et al. 2004; Yang and Yao 2005, 2008; Zou et al. 2004) |
| Evolutionary programming | EP | (Li and Yang 2008a; Tinos and Yang 1823) |
| Genetic algorithms | GA | (Abbass et al. 2004; Bui et al. 2005a, b; Dam et al. 2007; Deb and Nain 2007; Fan et al. 2007; Jin and Branke 2005; Kramer and Gallagher 2003; Laredo et al. 2008; Ling et al. 2007; Mack et al. 2007; Mattfeld and Bierwirth 2004; Mori and Kita 2000a, b; Mori et al. 2000a, b; Pankratz 2005; Rand and Riolo 2005; Rand and Riolo 2006; Rocco and Salazar 2007; Ronnewinkel and Martinetz 2001; Simões and Costa 2003; Stanhope and Daida 1999; Tezuka et al. 2007; Tinós and Yang 2007a, b, c; Ursem 2000; Wang et al. 2009a, b; Weicker 2002; Wineberg and Oppacher 2000; Yang 2006a, b, 2006a, b, 2007, 2008; Yang and Tinós 2007a, b; Yang and Yao 2008) |
| Memetic algorithms | MA | (Handa et al. 2007; Tenne and Armfield 2007) |
| Self-organizing scouts | SOS | (Branke and Schmeck 2003; Branke et al. 2000; Lung and Dumitrescu 2007, 2009; Zeng et al. 2007) |
| Other evolutive algorithms | OEA | (Ayvaz et al. 2006; Hu et al.2007) |
| Estimation-of-distribution algorithms | EDA | (Bosman 2005, 2005; Fernandes et al. 2008; Ghosh and Mühlenbein 2004; Yang 2007) |
| Immune-based algorithms | IBA | (Hart and Ross 1999; Olivetti de França et al. 2005; Simões and Costa 2003; Trojanowski and Wierzchon 2009; Yang 2006b; Yang and Yao 2008) |
| Neural networks | NN | (Deb and Nain 2007; Tinós and Yang 2007a) |
| Swarm intelligence | PSO | (Blackwell 2007; Blackwell and Branke 2004, 2006; Blackwell 2003, 2005; Carlisle and Dozier 2000; Du and Li 2008; Eberhart and Shi 2001; Esquivel and Coello Coello 2004, 2006; Hu and Eberhart 2002; Janson and Middendorf 2004; Li and Yang 2008a, b; Li et al. 2006; Mendes and Mohais 2005; Meyer et al. 2006; Novoa et al. 2009; Parrott and Li 2004; Reyes-Sierra and Coello 2007; Shi and Eberhart 2001; Venayagamoorthy 2004) |
| Other | O | (Aydin and Öztemel 2000; Dam et al. 2007; Kobliha et al. 2006; Meyer et al. 2006; Michalewicz et al. 2007; Moser and Hendtlass 2007; Yen et al. 2001; Zeng et al. 2007) |

(Li 2004) or hierarchical neighborhood structures (Janson and Middendorf 2004).

For cooperative strategies, Pelta et al. (2009a, b) use the grid of solutions per-se as an implicit diversity mechanism. Because there are more solutions than agents, then it is like having a sort of backup of solutions that are not being manipulated by any agent. Such solutions may be helpful in the future to get closer to new optimums. Also, they implement a simple diversification mechanism in two stages: perturbation of a certain percentage of solutions and a random re-positioning of the agents.

A combination of ACO and Estimation of Distribution Algorithm is presented by Fernandes et al. (2008) where a new strategy is able to respond to a change. This method uses vectors that emulate ACOs pheromone maps and act as a kind of memory, allowing to incorporate information from prior distributions into their current parameters.

An important problem in the use of this technique is that the continuous focus on maintenance of diversity can slow down the optimization process.

*React on changes* The detection of changes is also a key component of most dynamic optimization methods. In general, this detection is performed reevaluating the best solution, the second best or using some solutions as "sentinels". In any case, when a change in the environment occurs, the best fitness areas are probably others and the

proper detection of such changes can allow to track the good solutions or to vary the intensification of the search giving more weight to other areas.

The way in which reaction to change is managed is well exemplified in the following works. For example, Cobb (1990) used hypermutation technique as an adaptive operator in GAs, keeping the whole population after a change but increasing population diversity by drastically increasing the mutation rate for some number of generations.

Abbass et al. (2004) introduced the extended compact genetic algorithm (ECGA) to solve problems in dynamic environments. Their approach is based on random restarts of the population at each change so that diversity in the population can be increased at the beginning of each new environment. Another way of improvement comes through parameter self-adjustment. Tinos and Yang (1823) used a evolutionary programming algorithm with self adaptation of the value of $q$ parameter that controls the shape of a $q$-Gaussian mutation distribution. Parameter $q$ is increased after environmental changes, which allows for a higher number of long jumps to help the population escape from the local optimum or to converge faster to the global optimum. On the contrary, the latest stages within the same environmental change showed that $q$ reaches small values which improves the local search and intensification once a good search space location has been found.

Determining the useful amount of diversity is the main problem in these cases: too much diversity will resemble a restart and too little does not solve the convergency.

*Use of Memory* A natural way to enhance the strategies for DOPs is using memory schemes that work by storing useful information from the current scenario (either implicitly or explicitly) and reusing it at a later stage. Memory can help to preserve good solutions from the past that could be useful again at a later time, to reuse good solution components in newer solutions, to avoid visiting regions of the search space whose fitness is going down, and so on. The use of these techniques is only useful when periodic changes occur. In other words, when what is being observed now, can occur in the future.

An implicit memory scheme, for example, is an algorithm that uses representations containing more information than necessary and basically has some memory where good (partial) solutions may be stored and reused later as necessary. An explicit memory scheme is an algorithm that uses an extra storage space with explicit rules for storing and retrieving information.

Interesting surveys about memory-based evolutionary algorithms approaches for dynamic optimization problems appear in Branke (1999, 2001). Authors suggested the use of explicit memory through a memory population and search population. The memory population is responsible for remembering good solutions, maintaining a minimum quality and initiating jumps. The search population searches for new good solutions and submit these to the memory, but will not retrieve any information from it, and it is re-initialized at random after every change in the environment.

Karaman et al. (2005) propose a memory indexing evolutionary algorithm (MIA) that uses concepts originating from explicit memory-based dynamic evolutionary algorithms, the hyper-mutation mechanism and estimation of distribution approaches. MIA uses a distribution array (DA) that can be seen as a form of an external memory, where DA is a list of ratios representing the frequency of each allele at each gene position in the population.

Richter and Yang (2009) implemented an abstract memory scheme where the abstraction of good solutions is preserved in the memory to improve future problem solving. Abstraction means to select, evaluate, and code information before storing. In this case they use their approximate location in the search space to deduce a probabilistic model for the spatial distribution of good solutions.

Yang et al. (2010) applied a combination of a genetic algorithm with immigrants and an explicit memory scheme to solve Dynamic Shortest Path Routing Problems in wireless Mobile Ad Hoc Networks. They show how the immigrants improve the results in acyclic dynamic environments while the memory schemes outperform other techniques in cyclic environments.

Yang (2006a) presented an associative memory scheme for genetic algorithms where the environmental information is stored and associated with the current best individual of the population, and when the environment changes this information is used to create new individuals.

Pelta et al. (2009b) using cooperative strategies presented a dynamic fuzzy rule that made use of a history of previously seen costs to decide on the quality of solutions leading a successful optimization strategy. The definition of the fuzzy set is static but the domain where it is defined changes over time.

*Multiple populations* Employing several populations has been one of the most successful strategies to enhance the diversity in dynamic environments.

In EAs, a multi-population algorithm named self organizing scouts (SOS) presented by Branke et al. (2000) showed excellent results in some dynamic test problems. Multi-objective GAs have also been proposed for solving single objective functions. Also Bui et al. (2005b) apply multiobjective GAs for solving single-objective functions to dynamic optimization and test it on the moving peaks benchmark. Mendes and Mohais (2005) experimented with a multi-population approach of differential evolution (DE). Moser and Hendtlass (2007) used a multi-individual

version of extremal optimization (EO) applied to the moving peaks benchmark.

For the case of PSO, multi-population means many swarms. In that sense, Parrot and Li (2004) created a speciation-based PSO (SPSO), which dynamically adjust the number and the sizes of swarms through an ordered list of particles. SPSO was tested with good results in the benchmark defined by Morrison and De Jong (1999). Blackwell and Branke inspired by multipopulation EA approaches developed a multiswarm PSO (Blackwell and Branke 2004, 2006) with additional diversity mechanisms. The approaches were evaluated on the moving peaks benchmark showing that they performed well over a wide range of problem characteristics. Li and Yan (2008b) presented a fast multi-swarm algorithm (FMSO) where a parent swarm keeps exploring the entire search space using a global search operator, while many child swarms are dynamically created around the best solutions found by this parent swarm. In the experimental studies conducted for FMSO, the authors observed a greater robustness of this algorithm than that of several instances of MPB. Novoa et al. (2009) also used a multi-swarm PSO algorithm but adding an operator for controlling particle failures (CPF) and tested it on the moving peaks benchmark obtaining very good results.

### 4.1 Problems and methods

After reviewing problems and methods, it is interesting to analyze the relation between these two aspects. Table 6 shows a matrix where each row is a problem and each column a method. As stated before, genetic algorithms are clearly the most frequent approach while the moving peaks is the usual benchmark (many methods were already used to deal with it).

The most interesting aspect to observe is that the table is sparse, thus indicating that much work could be potentially done using current methods on different problems. This is specially noticeable on "real world" applications where there is a one-to-one relation in most of the cases. However, this situation poses additional challenges that are going to be discussed later.

## 5 Comparison and analysis of results for DOPs

When solving an optimization problem, there is always a need to assess the quality of solutions (or populations of solutions). This is useful to compare different solutions or to rank different algorithms or different versions of the same algorithm according to their performance. The use of appropriate measures, metrics, and statistical tests is an important part for the usefulness and meaning of this process. It is also a key point for being able to assess the quality of research results and to allow for a comparison of new research data with previously published results.

Therefore, it is important to discuss what measures and metrics have been used for DOPs and which ones are more appropriate for the former purposes.

### 5.1 Measures and metrics for assessing the results

Regarding what to measure or how to compare the performance of a set of algorithms over DOPs, the review performed indicates that there is no unified criteria. This is illustrated in Table 4 where a summary of the most used measures is presented. These measures range from totally method/problem-specific to some more generic proposals.

In a static context when solving a problem instance it is often enough to just report the best solution achieved by a method at the end of its execution. In some cases the costs in terms of memory/time are also considered, but no other factors are normally taken into account. But when we move to DOPs, reporting this simple values is not enough because we are normally interested not only in the solution quality at single points of time but also in many other aspects. These aspects include how well the methods are able to detect the problem changes and move to new better areas of the search space as they appear or how well they can track the existing good solutions as they move through the search space. It may be possible, for instance, that an algorithm is able to find the optimal solution at a given problem state (change) but it is totally lost for the rest of the execution. If we used a simple value like the best solution found, that algorithm may be ranked higher than another algorithm that keeps obtaining near-optimal solutions through all the optimization process and all the problem changes. On the contrary, the user will probably prefer this second algorithm over the occasionally optimal one. These concerns and the complexity of these tasks make the evaluation of the performance of methods in DOPs a research area *per se* and many different desirable goals and measures have been proposed to evaluate DOP optimization methods.

The usual approach taken in many contributions is depicted in Fig. 3. In a single run (top of the figure), the evolution of the error (calculated with respect to a known optimum) or the best solution found against evaluations (or time) looks like the one depicted in the plot. The small circles represents the "cost" (error or best) of the best solution found before the new change arrived. These costs are collected and some aggregation of the values is performed, typically the mean or median.

**Table 4** Measures for dynamic optimization and references to the papers where they are used

| Measure | References |
|---|---|
| Average best function value (ABFV) | (Abbass et al. 2004; Eberhart and Shi 2001; Montemanni et al. 2003; Schönemann 2004, 2007) |
| Average error | (Bui et al. 2005a, b; Kramer and Gallagher 2003) |
| Current best | (Aydin and Öztemel 2000; Bosman 2005, 2007; Dam et al. 2007; Eriksson and Olsson 2002; Esquivel and Coello Coello 2004; Hanshar and Ombuki-Berman 2007; Laredo et al. 2008; Mattfeld and Bierwirth 2004; Michalewicz et al. 2007; Neri and Mäkinen 2007; Olivetti de França et al. 2005; Shi and Eberhart 2001; Tenne and Armfield 2007; Tumer and Agogino 2007; Venayagamoorthy 2004) |
| Current best evolution | (Dam et al. 2007; Eberhart and Shi 2001; Fernandes et al. 2007; Jin and Sendhoff 2004; Mori et al. 2000a, b; Rand and Riolo 2005, 2006; Stanhope and Daida 1999; Tinos and Yang 1823) |
| Current best-of-generation evolution | (Morrison 2003, 2004; Quintão et al. 2007; Richter 2005; Saleem and Reynolds 2000; Schönemann 2007; Simões and Costa 2003; Tinós and Yang 2007a, b; Wineberg and Oppacher 2000; Yang 2003, 2005, 2006a, b, 2006a, b, 2007, 2008; Yang and Tinós 2007a, b; Yang and Yao 2005, 2008; Yang et al. 2010) |
| Collective mean fitness (CMF) | (Kobliha et al. 2006; Morrison 2003, 2004; Richter 2005) |
| Cumulative fitness | (Bosman 2005, 2007; Pankratz 2005) |
| Distance to known best | (Arnold and Beyer 2006; Carlisle and Dozier 2000; Rossi et al. 2008; Tenne and Armfield 2007; Ursem 2000; Weicker and Weicker 1999) |
| Diversity | (Bui et al. 2005a; Fernandes et al. 2008; Morrison 2004; Simões and Costa 2003; Wineberg and Oppacher 2000; Yang 2008; Yang and Tinós 2007a, b; Yang and Yao 2008) |
| Mean best-of-generation | (Esquivel and Coello Coello 2006; Fernandes et al. 2008; Laredo et al. 2008; Tinós and Yang 2007b, c, 2008; Wang et al. 2009b; Yang 2006b; Yang and Yao 2005, 2008) |
| Mean fitness error (MFE) | (Esquivel and Coello Coello 2006; Richter 2005; Richter and Yang 2009; Rossi et al. 2008) |
| Off-line error | (Ayvaz et al. 2006; Blackwell 2007; Branke and Schmeck 2003; Du and Li 2008; Li and Yang 2008b; Li et al. 2006; Lung and Dumitrescu 2007, 2009; Mendes and Mohais 2005; Moser and Hendtlass 2007; Olivetti de França et al. 2005; Pelta et al. 2009a, b; Trojanowski and Wierzchon 2009; Zeng et al. 2007) |
| Off-line error evolution | (Boumaza 2005; Branke et al. 2006a, b; Woldesenbet and Yen 2009) |
| Off-line performance | (Branke and Schmeck 2003; Eriksson and Olsson 2004; Jin and Branke 2005; Li and Yang 2008a; Novoa et al. 2009; Yang 2005, 2006a, 2007, 2008; Yang and Tinós 2007a, b; Yang et al. 2010; Zou et al. 2004) |
| Off-line performance evolution | (Branke 1999; Branke et al. 2000; Karaman et al. 2005; Li and Yang 2008a; Wang et al. 2009a) |
| Pareto optimality | (Deb and Nain 2007; Goh and Tan 2007; Lim et al. 2007; Mack et al. 2007; Reyes-Sierra and Coello 2007; Rocco and Salazar 2007) |
| Problem specific measures | (Elshamli et al. 2004; Handa et al. 2007; Hart and Ross 1999; Mack et al. 2007; Pankratz 2005; Peng and Reynolds 2004; Tezuka et al. 2007; Yan et al. 2004; Yen et al. 2001) |
| Robustness | (Goh and Tan 2007; Handa et al. 2007; Hu et al. 2007) |
| Standard deviation | (Mendes and Mohais 2005; Olivetti de França et al. 2005) |
| Time | (Ayvaz et al. 2006) |
| Other measures | (Arnold and Beyer 2002; Deb and Nain 2007; Ghosh and Mühlenbein 2004; Guntsch et al. 2001; Hu and Eberhart 2002; Meyer et al. 2007; Pelta et al. 2009a; Ronnewinkel and Martinetz 2001; Ursem 2000; Weicker 2002; Wineberg and Oppacher 2000) |

Usually, several runs are performed, so such values are further averaged as shown in Fig. 3 (bottom)

Many of the typical measures like current best and current average (Weicker 2002), Collective Mean Fitness (Morrison 2003), Off-line performance and offline error (Branke and Schmeck 2003), and Mean Fitness Error (Richter and Yang 2009) are based on similar ideas.

A step beyond was proposed by Weicker (2002) who presented an in-depth analysis on this topic, mainly focusing on EAs for dynamic environments; he considered the evaluation of three characteristics in a dynamic optimization process:

- *Accuracy*: The optimization accuracy at time $t$ is defined as

$$\text{accuracy}^t = \frac{\text{best}^t - \text{Min}^t}{\text{Max}^t - \text{Min}^t}$$

where $\text{best}^t$ is the fitness of the best candidate solution in the population at time $t$, $\text{Max}^t \in \mathbb{R}$ is the best fitness value in the search space and $\text{Min}^t \in \mathbb{R}$ is the worst fitness value in the search space.
- *Stability*: In the context of dynamic optimization, an adaptive algorithm is called stable if changes in the environment do not affect the optimization accuracy

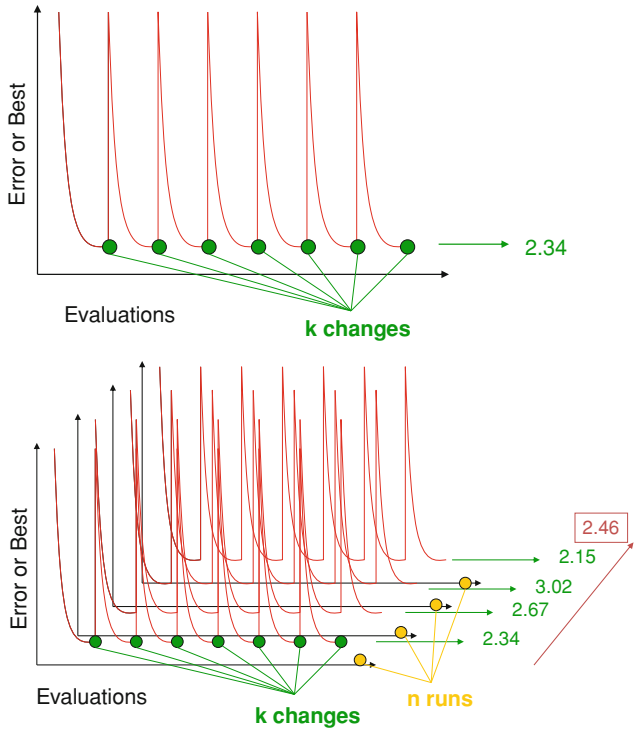


**Fig. 3** Basic schemes for calculating the performance of an algorithm in a dynamic environment. *Circles* represents the fitness of the best solution found before the new change arrived

severely. Even in the case of drastic changes an algorithm should be able to limit the respective fitness drop. The stability of an optimization algorithm at time $t$ is defined as

$$\text{stability}^t = \max\left\{0, \text{accuracy}^t - \text{accuracy}^{(t-1)}\right\}$$

- The stability cannot be relied as the only criterion to compare two algorithms since it makes no statement on the accuracy level. It can be seen as a consistency index of how reliably the algorithm keeps getting good results through the whole optimization process.
- *Reactivity*: An additional aspect that can be considered as a goal in a dynamic optimization process is the ability of an adaptive algorithm to react quickly to changes. The $\varepsilon$-reactivity of an algorithm at time $t$ is

$$\text{react}_{\varepsilon}^{(t)} = \min\left\{(t'-t) \;\middle|\; \frac{\text{accuracy}^{t'}}{\text{accuracy}^{t}} \geq (1-\varepsilon)\right\} \cup \{\text{maxgen} - t\}$$

where $t, t' \in \mathbb{N}$ and $t < t' \leq \text{maxgen}$, with maxgen referring to the number of generations in generation-based algorithms such as evolutionary algorithms. Lower values for the reactivity mean a better and faster reaction to changes.

While these proposals were mainly oriented to evaluate EAs, they are usually applicable to any other type of algorithm, or they are easily modifiable to be adapted to other algorithms. In this way, the best of the population could be a single algorithm solution if the algorithm does not use populations, or the generations can be substituted by other progression steps like the number of problem changes or fitness function evaluations.

Despite the insights that the analysis of the last two characteristics may provide, most authors have focused only on accuracy measures, not considering at all the stability or the reactivity. The accuracy is always a useful value but if the overall best fitness is not known the accuracy cannot be computed. Since the best fitness value changes there is no common basis for assessment of the quality of a solution, because a good fitness value at one time may be bad at another. Therefore, there is a need for alternative performance/quality measures so a different knowledge is required such as the position of the optimum (usually only available in academic benchmarks), the knowledge of the best fitness values or even no global knowledge at all.

### 5.2 Statistical tests

Given a set of algorithms and their corresponding performance values, there is a need to assess which is the "best" alternative. From the pioneer works of Hooker (1995) or Golden and Stewart (1985) until recent papers (García et al. 2009; Rardin and Uzsoy 2001) the use of statistical tests are advocated as a way of introducing validation into empirical investigations of algorithms.

When dealing with static problems, proper statistical validations is nowadays a *sine qua non* condition for further consideration of the work.

In dynamic optimization papers, the situation is similar to the one observed in the early days of optimization with metaheuristics. After analyzing the papers in the repository, we find that many works make a comparison of results using only mean and standard deviation values while others resort to parametric tests (like Student's t-test) without assessing if the set of performance values follows a normal distribution. It should also be remarked that several authors are starting to consider non-parametric statistical test in order to confirm if the observed differences in performance can be obtained by chance. Table 5 provides a small showcase of papers and the test they have applied to compare the algorithms.

In our opinion, if one wants to compare the behavior of a set of algorithms over a dynamic optimization problem, and the performance obtained by a single algorithm in an individual run is resumed as a single value denoting its performance, then the comparison should be performed using non-parametric statistical testing. Of course, parametric tests can also be applied whenever the conditions for

**Table 5** Tests for dynamic optimization and references to the papers where they are used

| Test | References |
|---|---|
| ANOVA | (Novoa et al. 2009; Pelta et al. 2009b) |
| Mann–Whitney *U* test | (Pelta et al. 2009b; Woldesenbet and Yen 2009) |
| Student's *t* test | (Du and Li 2008; Fernandes et al. 2008; Karaman et al. 2005; Laredo et al. 2008; Tezuka et al. 2007; Tinós and Yang 2007a, b, 2008; Wang et al. 2009a, b; Weicker 2002; Yang 2005, 2006a, b, 2006a, b, 2007, 2008; Yang and Tinós 2007a, b; Yang and Yao 2005, 2008; Yang et al. 2010) |
| Tamhane test | (Novoa et al. 2009; Pelta et al. 2009b) |
| Wilcoxon's non-parametric test | (Mendes and Mohais 2005) |

their application are verified first. See, for example, Sheskin (2004) for a general discussion on testing. In the context of dynamic optimization as understood here, we can suggest to follow the guidelines and indications provided in (García et al. 2009), where the authors discuss how to made single-problem analysis and multiple-problem analysis in the context of evolutionary algorithms for real parameter optimization.

## 6 Summary and suggestions for future research

This paper presents an overview of the related works on dynamic optimization problems emphasizing what has been done in the past decade. A large number of papers were first collected from several search engines and then carefully analyzed and categorized. The result is now available at http://www.dynamic-optimization.org where the information can be explored through several dimensions. Three points of view were mainly considered: methods, measures, and problems.

Regarding the methods analyzed, the use of population-based techniques represents the most usual approach. From an almost exclusive use of evolutionary algorithms in the beginning, there is now an increasing interest in other optimization techniques such as particle swarm optimization, ant colony optimization, cooperative strategies, immune based algorithms, etc. It is clear that, besides more or less appealing names, the way to go are self-adaptive systems that can "sense" the environment and react properly, doing some self-tuning or even reconfiguring (changing components) by themselves. As stated in Plexousakis (2006) regarding software intensive systems:

> The challenge is to develop algorithms, methods, tools, and theoretical foundations that enable effective design of adaptive systems which harness and control properties that emerge through the interaction of systems and their environment.

Regarding the problems, it is clear that many synthetic test problems and DOPs generators are available but what is not clear at all is what features of "real world" these generators should model. In this sense, it is interesting to note that most of the research is done considering dynamism in the objective function. We are not aware of other works considering time varying restrictions, or dimensionality changes that may perfectly represent real world problem features.

We should also highlight that there is a clear drawback with the reproducibility of published experiments and results as the concept of "problem instance" is not as clear as in the static optimization case. The "dynamic" part is lost from the definition of the problem instance and, being true that the *type of change* is perfectly defined in artificial problems, this fact would be critical in other scenarios.

For example, in the so-called "real-world" problems, the main open question is if one wishes to apply a new algorithm to such problem, what information about it should be provided? In other words, how the dynamism of the problem should be represented? For example, if we consider a dynamic vehicle routing problem with time windows, where the customers can be added or deleted dynamically from the schedule, we will have both a hard problem to solve and no simple way to share such information about timing. This could be an interesting line of research.

In terms of the measures and tests, we observed that most of the used measures need a precise knowledge of the problem at hand, and they reduce a whole dynamic run to a single value. The use of proper statistical testing to compare different results/algorithms, instead of simple comparisons based on reported means and standard deviations, should be promoted. In particular, non-parametric tests should be applied whenever the conditions for the application of their parametric versions do not hold. The literature on the topic is wide and there is no reason for not using them.

In our opinion, there is a need for new measures that take into account what happens "during" the run. In this sense, perhaps the field of time series theory may provide some ideas. At the end, we will want to observe and be able to compare temporal patterns (or averages of them).

After this review, we consider that the following aspects should deserve further consideration:

- On problems:
  - Consider other types of dynamism, like dimensionality change.

**Table 6** Dynamic optimization problems (DOPs) and methods cross-references to the papers where they are used

| Problem | Methods (abbreviations are taken from Table 3) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ACO | CS | CA | EDA | ES | EA | EP | GA |
| Real world applications | | | | | | | | |
| Aerospace design | | | | | | | | (Mack et al. 2007) |
| Car distribution system | | | | | | | | |
| Evolutionary robotics | | | | | | | | (Tinós and Yang 2007a, b) |
| Financial optimization problems | | | | | | | | (Tezuka et al. 2007) |
| Mobile ad hoc networks | | | | | | | | (Yang et al. 2010) |
| Modeling of ship trajectory | | | | | | (Smierzchalski and Michalewicz 2000) | | |
| Path planning | | | | | | | | (Elshamli et al. 2004) |
| Pollution control | | | | | | | | |
| Pose problem | | | | | | (Rossi et al. 2008) | | |
| Rober problem | | | | | | (Tumer and Agogino 2007) | | |
| Robust design | | | | | | (Barrico and Antunes 2007; Goh and Tan 2007; Handa et al. 2007; Lim et al. 2007; Rocco and Salazar 2007) | | (Fan et al. 2007; Ling et al. 2007; Rocco and Salazar 2007) |
| Salting route optimization | | | | | | (Handa et al. 2007) | | |
| Structural optimization | | | | | | (Neri and Mäkinen 2007) | | |
| Varied-line-spacing holographic grating | | | | | | | | (Ling et al. 2007) |
| Wireless sensor networks | | | | | | (Quintão et al. 2007) | | |
| Synthetic dynamic problems | | | | | | | | |
| DF1 generator (cones) | | | (Peng and Reynolds 2004; Saleem and Reynolds 2000) | | | (Eriksson and Olsson 2002; Morrison 2004) | | |
| Dyn. Ackley function | | | | | (Schönemann 2004, 2007) | | | |

**Table 6** continued

| Problem | Methods (abbreviations are taken from Table 3) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ACO | CS | CA | EDA | ES | EA | EP | GA |
| Dyn. bit matching | | | | | | (Jin and Branke 2005) | | (Jin and Branke 2005; Mori and Kita 2000a, b; Stanhope and Daida 1999) |
| Dyn. deceptive functions | | | | (Yang 2007) | | (Yang and Yao 2005, 2008) | | (Tinós and Yang 2007a, b; Wang et al. 2009a, b; Yang 2003, 2005, 2006a, b, 2007; Yang and Yao 2008) |
| Dyn. knapsack problem | | | | | | (Branke et al. 2006a, b; Jin and Branke 2005; Karaman et al. 2005; Yang and Yao 2005) | | (Jin and Branke 2005; Mori and Kita 2000a, b; Simões and Costa 2003; Wang et al. 2009a, b; Yang 2008; Yang and Tinós 2007a, b) |
| Dyn. Onemax function | (Fernandes et al. 2008) | | | (Fernandes et al. 2008; Yang 2007) | | (Droste 2003; Yang and Yao 2008) | | (Wang et al. 2009a, b; Yang 2003, 2005, 2007, 2008; Yang and Tinós 2007a, b; Yang and Yao 2008) |
| Dyn. plateau functions | | | | | | | | (Wang et al. 2009a, b; Yang 2006a, b, 2008) |
| Dyn. problem generator | | | | | (Jin and Sendhoff 2004) | (Morrison 2004; Ursem et al. 2002; Yang and Yao 2005) | (Li and Yang 2008a, b) | (Rand and Riolo 2005, 2006; Tinos and Yang 2007) |
| Dyn. Rastrigin function | | | | | (Schönemann 2004, 2007) | | (Tinos and Yang 1823) | |
| Dyn. royal road function | (Fernandes et al. 2007, 2008) | | | (Fernandes et al. 2008; Yang 2007) | | (Yang and Yao 2005) | | (Tinós and Yang 2007a, b; Wang et al. 2009a, b; Yang 2003, 2007; Yang and Tinós 2007a, b) |
| Dyn. scheduling | | | | | | | | (Mattfeld and Bierwirth 2004) |

**Table 6** continued

| Problem | Methods (abbreviations are taken from Table 3) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | ACO | CS | CA | EDA | ES | EA | EP | GA |
| Dyn. sphere | | | | | (Arnold and Beyer 2002, 2006; Boumaza 2005; Schönemann 2004, 2007) | | | |
| Dyn. vehicle routing problem (DVRP) | (Hanshar and Ombuki-Berman 2007; Montemanni et al. 2003) | | | | | | | (Hanshar and Ombuki-Berman 2007; Pankratz 2005) |
| Griewank function | | | | | | | | (Wineberg and Oppacher 2000) |
| Moving parabola | | | | | | | | (Abbass et al. 2004; Mori and Kita 2000a, b) |
| Moving peaks benchmark (MPB) | | (Lung and Dumitrescu 2007, 2009; Pelta et al. 2009a, b) | | | (Mendes and Mohais 2005) | (Ayvaz et al. 2006; Branke and Schmeck 2003; Eriksson and Olsson 2004; Jin and Branke 2005; Li and Yang 2008a, b; Lung and Dumitrescu 2007, 2009; Zou et al. 2004) | | (Bui et al. 2005a, b; Jin and Branke 2005; Kramer and Gallagher 2003; Mori and Kita 2000a, b; Ronnewinkel and Martinetz 2001; Ursem 2000) |
| Rosenbrock function | | | | | | | | |
| Shaky ladder hyperplane-defined functions | | | | | | | | (Rand and Riolo 2005, 2006) |
| Time-linkage numerical problems | | | | (Bosman 2005, 2007) | | (Bosman 2005, 2007) | | |
| Trap function based synthetic problems | | | | | | | | (Abbass et al. 2004; Laredo et al. 2008) |
| XOR-based synthetic dynamic problems | | | | | | (Rohlfshagen et al. 2009a, b) | | (Yang 2003, 2005, 2006a, b) |
| Other | (Fernandes et al. 2007; Guntsch et al. 2001) | | | (Ghosh and Mühlenbein 2004) | (Weicker and Weicker 1999) | (Branke 1999; Tenne and Armfield 2007; Woldesenbet and Yen 2009; Yan et al. 2004) | | (Mori and Kita 2000a, b; Mori et al. 2000a, b) |

**Table 6** continued

| Problem | Methods (abbreviations are taken from Table 3) | | | | | | |
|---|---|---|---|---|---|---|---|
| | IBA | MA | SOS | OEA | NN | PSO | O |
| Real world applications | | | | | | | |
| Aerospace design | | | | | | | |
| Car distribution system | | | | | | | |
| Evolutionary robotics | | | | | | | (Michalewicz et al. 2007) |
| Financial optimization problems | | | | | (Tinós and Yang 2007a, b) | | |
| Mobile ad hoc networks | | | | | | | |
| Modeling of ship trajectory | | | | | | | |
| Path planning | | | | | | | |
| Pollution control | | | | | | | (Michalewicz et al. 2007) |
| Pose problem | | | | | | | (Michalewicz et al. 2007) |
| Rober problem | | | | | | | |
| Robust design | | | | | | | |
| Salting route optimization | (Hu et al. 2007) | (Handa et al. 2007) | | (Hu et al. 2007) | | | |
| Structural optimization | | (Handa et al. 2007) | | | | | |
| Varied-line-spacing holographic grating | | | | | | | |
| Synthetic dynamic problems | | | | | | | |
| DF1 generator (cones) | | | | | | (Esquivel and Coello Coello 2004, 2006) | |
| Dyn. Ackley function | | | | | | | |
| Dyn. bit matching | | | | | | | |
| Dyn. deceptive functions | (Yang 2006a, b; Yang and Yao 2008) | | | | | | |

**Table 6** continued

| Problem | Methods (abbreviations are taken from Table 3) | | | | | | |
|---|---|---|---|---|---|---|---|
| | IBA | MA | SOS | OEA | NN | PSO | O |
| Dyn. knapsack problem | (Simões and Costa 2003) | | | | | | |
| Dyn. Onemax function | (Yang and Yao 2008) | | | | | | |
| Dyn. plateau functions | (Yang 2006a, b) | | | | | | |
| Dyn. problem generator | (Trojanowski and Wierzchon 2009) | | | | | (Li and Yang 2008a, b) | |
| Dyn. Rastrigin function | (Olivetti de França et al. 2005) | | | | | (Du and Li 2008; Shi and Eberhart 2001; Venayagamoorthy 2004) | |
| Dyn. royal road function | | | | | | | |
| Dyn. scheduling | (Hart and Ross 1999) | | | | | | (Aydin and Öztemel 2000) |
| Dyn. sphere | (Olivetti de França et al. 2005) | | | | | | |
| Dyn. vehicle routing problem (DVRP) | | | | | | | |
| Griewank function | (Olivetti de França et al. 2005) | | | | | (Shi and Eberhart 2001; Venayagamoorthy 2004) | |
| Moving parabola | | | | | | (Eberhart and Shi 2001; Hu and Eberhart 2002) | |
| Moving peaks benchmark (MPB) | (Trojanowski and Wierzchon 2009) | | (Branke and Schmeck 2003; Branke et al. 2000; Lung and Dumitrescu 2007, 2009) | (Ayvaz et al. 2006) | | (Blackwell 2007; Blackwell and Branke 2004, 2006; Blackwell 2003, 2005; Du and Li 2008; Janson and Middendorf 2004; Li and Yang 2008a, b; Li et al. 2006; Lung and Dumitrescu 2007, 2009; Mendes and Mohais 2005; Meyer et al. 2006; Novoa et al. 2009; Parrott and Li 2004) | (Meyer et al. 2006; Moser and Hendtlass 2007) |

**Table 6** continued

| Problem | Methods (abbreviations are taken from Table 3) | | | | | | |
|---|---|---|---|---|---|---|---|
| | IBA | MA | SOS | OEA | NN | PSO | O |
| Rosenbrock function | (Olivetti de França et al. 2005) | | | | | (Hu and Eberhart 2002; Shi and Eberhart 2001; Venayagamoorthy 2004) | |
| Shaky ladder hyperplane-defined functions | | | | | | | |
| Time-linkage numerical problems | | | | | | | |
| Trap function based synthetic problems | | | | | | | |
| XOR-based synthetic dynamic problems | | | | | | | |
| Other | | (Tenne and Armfield 2007) | | | (Deb and Nain 2007) | (Carlisle and Dozier 2000) | (Dam et al. 2007; Yen et al. 2001) |

- Further explore multi-objective dynamic optimization.

– On methods:

- Explore self-adaptive algorithms as well as other successful population-based algorithms like scatter search, that could also perform well on DOPs.
- The assumption that trajectory methods will not work well for DOPs should be reconsidered. The results on González et al. (2010) where several trajectory solvers are coupled with a cooperative strategy suggest that these methods could greatly outperform evolutionary like algorithms on some cases. This approach combined with some sort of cooperation could possibly lead to very good results on some DOPs.
- Suggest the authors to provide their algorithms source code or executables for checking and reproducibility purposes as it is done in other areas, like bioinformatics.

– On analyzing performance:

- Define a computational experimentation protocol.
- Development of new measures beyond "single valued" ones.
- Promote the use of non-parametric testing.

Moreover, it will be very interesting to have a broader experimentation done on the different issues described on this paper (methods, problems, measures, tests, and so on) as well as the interactions between them. This could start to provide insights into how to select a good algorithm for a given DOP or which measures to use for a given problem with respect to its severity of change, if the optimum is known or not, the shape of the search landscape or any other problem characteristics. A good understanding of the interactions between all these issues will surely be helpful for tackling more complex real-world problems and to focus the research into the most promising topics while reducing the efforts on the less promising ones.

To conclude, we would like to recognize that this review has been a hard task. Any review process is also a filtering process, so we would like to apologize to the readers and researchers for any possible omissions. We will be happy to receive their feedback in order to improve the resources available in our web repository. If this study motivates students and researchers to use and investigate on these problems, then the aim of the paper will be fulfilled.

**Acknowledgments** This work has been partially funded by projects TIN2008-01948 from the Spanish Ministry of Science and Innovation, and P07-TIC-02970 from the Andalusian Government.

## References


Abbass HA, Sastry K, Goldberg DE (2004) Oiling the wheels of change: the role of adaptive automatic problem decomposition in non-stationary environments. IlliGAL report no. 2004029. Technical report, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory (IlliGAL)

Angeline PJ (1997) Tracking extrema in dynamic environments. In: Evolutionary programming VI. Lecture notes in computer science, vol 1213. Springer, Berlin, pp 335–345

Arnold DV, Beyer H-G (2002) Random dynamics optimum tracking with evolution strategies. In: Parallel problem solving from nature VII. Springer, Berlin, pp 3–12

Arnold DV, Beyer H-G (2006) Optimum tracking with evolution strategies. Evol Comput 14(3):291–308

Aydin ME, Öztemel E (2000) Dynamic job-shop scheduling using reinforcement learning agents. Robot Auton Syst 33(2–3): 169–178

Ayvaz D, Topcuoglu H, Gurgen F (2006) A comparative study of evolutionary optimisation techniques in dynamic environments. In: Proceedings of the genetic and evolutionary computation conference. ACM, New York, pp 1397–1398

Barrico C, Antunes C (2007) An evolutionary approach for assessing the degree of robustness of solutions to multi-objective models. In: Studies in computational intelligence, vol 51. Springer, New York, pp 565–582

Blackwell TM (2003) Swarms in dynamic environments. In: Proceedings of the genetic and evolutionary computation conference. Lecture notes in computer science, vol 2723. Springer, Berlin, pp 1–12

Blackwell TM (2005) Particle swarms and population diversity. Soft Comput: A Fusion Found Methodol Appl 9(11):793–802

Blackwell T (2007) Particle swarm optimization in dynamic environments. In: Studies in computational intelligence, vol 51. Springer, New York, pp 29–49

Blackwell T, Branke J (2004) Multi-swarm optimization in dynamic environments. In: Applications of evolutionary computing. Lecture notes in computer science, vol 3005. Springer, Berlin, pp 489–500

Blackwell T, Branke J (2006) Multiswarms, exclusion, and anti-convergence in dynamic environments. IEEE Trans Evol Comput 10(4):459–472

Bosman PAN (2005) Learning, anticipation and time-deception in evolutionary online dynamic optimization. In: Proceedings of the 2005 workshops of the genetic and evolutionary computation conference. ACM, New York, pp 39–47

Bosman P (2007) Learning and anticipation in online dynamic optimization. In: Studies in computational intelligence, vol 51. Springer, New York, pp 129–152

Boumaza A (2005) Learning environment dynamics from self-adaptation: a preliminary investigation. In: Proceedings of the 2005 workshops of the genetic and evolutionary computation conference. ACM, New York, pp 48–54

Branke J (1999) Memory enhanced evolutionary algorithms for changing optimization problems. In: Angeline PJ, Michalewicz Z, Schoenauer M, Yao X, Zalzala A (eds) Proceedings of the IEEE Congress on evolutionary computation, vol 3. IEEE Press, pp 1875–1882

Branke J (2001) Evolutionary optimization in dynamic environments. In: Genetic algorithms and evolutionary computation, vol 3. Kluwer Academic Publishers, Dordrecht

Branke J (2005) Editorial: special issue on dynamic optimization problems. Soft Comput: A Fusion Found Methodol Appl 9(11):777

Branke J, Jin Y (2006a) Guest editorial special issue on evolutionary computation in the presence of uncertainty. IEEE Trans Evol Comput 10(4):377–379

Branke J, Schmeck H (2003) Designing evolutionary algorithms for dynamic optimization problems. In: Advances in evolutionary computing: theory and applications, pp 239–262

Branke J, Kaubler T, Schmidt C, Schmeck H (2000) A multi-population approach to dynamic optimization problems. In: Adaptive computing in design and manufacture, pp 299–308

Branke J, Orbayi M, Uyar S (2006) The role of representations in dynamic knapsack problems. In: Applications of evolutionary computing. Lecture notes in computer science, vol 3907. Springer, Berlin, pp 764–775

Bui L, Abbass H, Branke J (2005a) Multiobjective optimization for dynamic environments. In: Proceedings of the IEEE Congress on evolutionary computation, vol 3, pp 2349–2356

Bui LT, Branke J, Abbass HA (2005b) Diversity as a selection pressure in dynamic environments. In: Proceedings of the genetic and evolutionary computation conference. ACM, New York, pp 1557–1558

Carlisle A, Dozier G (2000) Adapting particle swarm optimization to dynamic environments. In: Proceedings of the international conference on artificial intelligence (ICAI), pp 429–434

Cobb HG (1990) An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuouis, time-dependent nonstationary environments. Technical report AIC-90-001, Naval Research Laboratory

Dam H, Lokan C, Abbass H (2007) Evolutionary online data mining: an investigation in a dynamic environment. In: Studies in computational intelligence, vol 51. Springer, New York, pp 153–178

Dasgupta D, Mcgregor DR (1992) Nonstationary Function Optimization Using the Structured Genetic Algorithm. In R. Manner and B. Manderick, editors, Parallel Problem Solving from Nature. Elsevier, pp 145–154

Deb K, Nain P (2007) An evolutionary multi-objective adaptive meta-modeling procedure using artificial neural networks. In: Studies in computational intelligence, vol 51. Springer, New York, pp 297–322

Droste S (2003) Analysis of the (1+1) EA for a dynamically bitwise changing OneMax. In: Cantu-Paz E (ed) Lecture notes in computer science, vol 2723. Springer, New York, pp 909–921

Du W, Li B (2008) Multi-strategy ensemble particle swarm optimization for dynamic optimization. Inf Sci 178(15):3096–3109

Eberhart R, Shi Y (2001) Tracking and optimizing dynamic systems with particle swarms. In: Proceedings of the IEEE Congress on evolutionary computation, vol 1, pp 94–100

Elshamli A, Abdullah H, Areibi S (2004) Genetic algorithm for dynamic path planning. In: Canadian conference on electrical and computer engineering

Eriksson R, Olsson B (2002) On the behavior of evolutionary global-local hybrids with dynamic fitness functions. In: Parallel problem solving from nature VII. Springer, New York

Eriksson R, Olsson B (2004) On the performance of evolutionary algorithms with life-time adaptation in dynamic fitness landscapes. In: Proceedings of the IEEE Congress on evolutionary computation, vol 2, pp 1293–1300

Esquivel S, Coello Coello C (2004) Particle swarm optimization in non-stationary environments. In: Advances in artificial intelligence—IBERAMIA 2004. Springer, New York

Esquivel SC, Coello Coello CA (2006) Hybrid particle swarm optimizer for a class of dynamic fitness landscape. Eng Optim 38:873–888

Fan Z, Wang J, Wen M, Goodman E, Rosenberg R (2007) An evolutionary approach for robust layout synthesis of MEMS. In: Studies in computational intelligence, vol 51. Springer, New York, pp 519–542

Fernandes CM, Rosa AC, Ramos V (2007) Binary ant algorithm. In: Proceedings of the genetic and evolutionary computation conference. ACM, New York, pp 41–48

Fernandes CM, Lima C, Rosa AC (2008) UMDAs for dynamic optimization problems. In: Proceedings of the genetic and evolutionary computation conference. ACM, New York, pp 399–406

Fogel LJ, Owens AJ, Walsh MJ (1966) Artificial intelligence through simulated evolution. Wiley, New York

García S, Molina D, Lozano M, Herrera F (2009) A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. J Heuristics 15:617–644

Ghosh A, Mühlenbein H (2004) Univariate marginal distribution algorithms for non-stationary optimization problems. Int J Knowl Intell Eng Syst 8:129–138

Goh C, Tan K (2007) Evolving the tradeoffs between pareto-optimality and robustness in multi-objective evolutionary algorithms. Studies in computational intelligence, vol 51. Springer, New York, pp 457–478

Goldberg DE, Smith RE (1987) Nonstationary function optimization using genetic algorithm with dominance and diploidy. In: Grefensette JJ (ed) Proceedings of the second international conference on genetic algorithms and their application. Lawrence Erlbaum Associates Inc., pp 59–68

Golden B, Stewart W (1985) Empirical evaluation of heuristics. In: Lawler E, Lenstra J, Kan AR, Shmoys D (eds) The traveling salesman problem: a guided tour of combinatorial optimization. Wiley, New York

González JR, Masegosa AD, García IJ (2010) A cooperative strategy for solving dynamic optimization problems. Memet Comput (in press). doi:10.1007/s12293-010-0031-x

Grefenstette JJ (1992) Genetic algorithms for changing environments. In: Männer R, Manderick B (eds) Proceedings of 2nd international conference on parallel problem solving from nature. Elsevier, pp 137–144

Guntsch M, Middendorf M, Schmeck H (2001) An ant colony optimization approach to dynamic TSP. In: Spector L et al (eds) Proceedings of the genetic and evolutionary computation conference. Morgan Kaufmann, Massachusetts, pp 860–867

Handa H, Chapman L, Yao X (2007) Robust salting route optimization using evolutionary algorithms. In: Studies in computational intelligence, vol 51. Springer, New York, pp 497–517

Hanshar FT, Ombuki-Berman BM (2007) Dynamic vehicle routing using genetic algorithms. Appl Intell 27(1):89–99

Hart E, Ross P (1999) An immune system approach to scheduling in changing environments. In: Proceedings of the genetic and evolutionary computation conference. Morgan Kaufmann, Massachusetts, pp 1559–1565

Hooker JN (1995) Testing heuristics: we have it all wrong. J Heuristics 1(1):33–42

Hu X, Eberhart RC (2002) Adaptive particle swarm optimization: detection and response to dynamic systems. In: Proceedings of the IEEE Congress on evolutionary computation, vol 2, pp 1666–1670

Hu J, Li S, Goodman E (2007) Evolutionary robust design of analog filters using genetic programming. Studies in computational intelligence, vol 51. Springer, New York, pp 479–496

Janson S, Middendorf M (2004) A hierarchical particle swarm optimizer for dynamic optimization problems. In: Applications of evolutionary computing. Lecture notes in computer science, vol 3005. Springer, Berlin, pp 513–524

Jin Y, Branke J (2005) Evolutionary optimization in uncertain environments: a survey. IEEE Trans Evol Comput 9(3):303–317

Jin Y, Sendhoff B (2004) Constructing dynamic optimization test problems using the multi-objective optimization concept. In: Raidl G.R. (ed) Lecture notes in computer science, vol 3005. Springer, New York, pp 525–536

Karaman A, Uyar S, Eryigit G (2005) The memory indexing evolutionary algorithm for dynamic environments. In: Applications on evolutionary computing. Lecture notes in computer science, vol 3449. Springer, Berlin, pp 563–573

Kobliha M, Schwarz J, Oenáek J (2006) Bayesian optimization algorithms for dynamic problems. In: Applications of evolutionary computing. Lecture notes in computer science, vol 3907. Springer, Berlin, pp 800–804

Kramer G, Gallagher J (2003) Improvements to the *CGA enabling online intrinsic evolution in compact EH devices. In: Proceedings of the NASA/DoD conference on evolvable hardware, pp 225–231

Laredo JL, Castillo PA, Mora AM, Merelo JJ, Rosa A, Fernandes C (2008) Evolvable agents in static and dynamic optimization problems. In: Proceedings of the 10th international conference on parallel problem solving from nature. Springer, New York, pp 488–497

Li C, Yang S (2008a) A generalized approach to construct benchmark problems for dynamic optimization. In: Simulated evolution and learning. Lecture notes in computer science, vol 5361. Springer, Berlin, pp 391–400

Li C, Yang S (2008b) Fast multi-swarm optimization for dynamic optimization problems. In: Fourth international conference on natural computation, vol 7. IEEE Computer Society, pp 624–628

Li X (2004) Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization. In: Proceedings of the genetic and evolutionary computation conference. Lecture notes in computer science, vol 3102. Springer, Berlin, pp 105–116

Li X, Branke J, Blackwell T. (2006) Particle swarm with speciation and adaptation in a dynamic environment. In: Proceedings of the genetic and evolutionary computation conference, vol 1. ACM, New York, pp 51–58

Lim D, Ong Y-S, Lim M-H, Jin Y (2007) Single/multi-objective inverse robust evolutionary design methodology in the presence of uncertainty. In: Studies in computational intelligence, vol 51. Springer, New York, pp 437–456

Ling Q, Wu G, Wang Q (2007) Deterministic robust optimal design based on standard crowding genetic algorithm. In: Studies in computational intelligence, vol 51. Springer, New York, pp 583–598

Lung RI, Dumitrescu D (2007) A new collaborative evolutionary-swarm optimization technique. In: Proceedings of the genetic and evolutionary computation conference. ACM, New York, pp 2817–2820

Lung RI, Dumitrescu D (2009) Evolutionary swarm cooperative optimization in dynamic environments. Nat Comput 9(1):83–94

Mack Y, Goel T, Shyy W, Haftka R (2007) Surrogate model-based optimization framework: a case study in aerospace design. In: Studies in computational intelligence, vol 51. Springer, New York, pp 323–342

Mattfeld DC, Bierwirth C (2004) An efficient genetic algorithm for job shop scheduling with tardiness objectives. Eur J Oper Res 155(3):616–630

Mendes R, Mohais A (2005) DynDE: a differential evolution for dynamic optimization problems. In: Proceedings of the IEEE Congress on evolutionary computation, vol 3, pp 2808–2815

Meyer KD, Nasuto SJ, Bishop M (2006) Stochastic diffusion search: partial function evaluation in swarm intelligence dynamic optimisation. In: Stigmergic optimization. Studies in computational intelligence, vol 31. Springer, Berlin, pp 185–207

Michalewicz Z, Schmidt M, Michalewicz M, Chiriac C (2007) Adaptive business intelligence: three case studies. In: Studies in computational intelligence, vol 51. Springer, New York, pp 179–196

Montemanni R, Gambardella L, Rizzoli A, Donati A (2003) A new algorithm for a dynamic vehicle routing problem based on ant colony system. In: Second international workshop on freight transportation and logistics, pp 27–30

Mori N, Kita H (2000a) Genetic algorithms for adaptation to dynamic environments: a survey. In: IEEE industrial electronics conference, IECON, vol 4, pp 2947–2952

Mori N, Kude T, Matsumoto K (2000b) Adaptation to a dynamical environment by means of the environment identifying genetic algorithm. In: IEEE industrial electronics conference, IECON 2000

Morrison RW (2003) Performance measurement in dynamic environments. In: GECCO Proceedings of workshop on evolutionary algorithms for dynamic optimization problems, pp 5–8

Morrison RW (2004) Designing evolutionary algorithms for dynamic environments. Springer, New York

Morrison R, De Jong K (1999) A test problem generator for non-stationary environments. In: Proceedings of the IEEE Congress on evolutionary computation, vol 3, pp 2047–2053

Moser I, Hendtlass T (2007) A simple and efficient multi-component algorithm for solving dynamic function optimisation problems. In: Proceedings of the IEEE Congress on evolutionary computation, pp 252–259

Neri F, Mäkinen R (2007) Hierarchical evolutionary algorithms and noise compensation via adaptation. In: Studies in computational intelligence, vol 51. Springer, New York, pp 345–369

Novoa P, Pelta DA, Cruz C, del Amo IG (2009) Controlling particle trajectories in a multi-swarm approach for dynamic optimization problems. In: International work-conference on the interplay between natural and artificial computation, IWINAC 2009. Lecture notes in computer science, vol 5601. Springer, Berlin, pp 285–294

Olivetti de França F, Von Zuben FJ, Nunes de Castro L (2005) An artificial immune network for multimodal function optimization on dynamic environments. In: Proceedings of the genetic and evolutionary computation conference. ACM, New York, pp 289–296

Pankratz G (2005) Dynamic vehicle routing by means of a genetic algorithm. Int J Phys Distrib Logist Manag 35(5):362–383

Parrott D, Li X (2004) A particle swarm model for tracking multiple peaks in a dynamic environment using speciation. In: Proceedings of the IEEE Congress on evolutionary computation, vol 1, pp 98–103

Pelta D, Cruz C, Gonzalez JR (2009a) A study on diversity and cooperation in a multiagent strategy for dynamic optimization problems. Int J Intell Syst 24:844–861

Pelta D, Cruz C, Verdegay JL (2009b) Simple control rules in a cooperative system for dynamic optimisation problems. Int J Gen Syst 38(7):701–717

Peng B, Reynolds R (2004) Cultural algorithms: knowledge learning in dynamic environments. In: Proceedings of the IEEE Congress on evolutionary computation, pp 1751–1758

Plexousakis D (2006) Beyond the horizon: anticipating future and emerging information society technologies. Technical report, European Research Consortium for Informatics and Mathematics. http://beyond-the-horizon.ics.forth.gr/

Quintão F, Nakamura F, Mateus G (2007) Evolutionary algorithms for combinatorial problems in the uncertain environment of the wireless sensor networks. In: Studies in computational intelligence, vol 51. Springer, New York, pp 197–222

Rand W, Riolo R (2005) Shaky ladders, hyperplane-defined functions and genetic algorithms: systematic controlled observation in dynamic environments. In: Applications on evolutionary computing. Lecture notes in computer science, vol 3449. Springer, Berlin, pp 600–609

Rand W, Riolo R (2006) The effect of building block construction on the behavior of the GA in dynamic environments: a case study using the shaky ladder hyperplane-defined functions. In: Applications of evolutionary computing. Lecture notes in computer science, vol 3907. Springer Berlin, pp 776–787

Rardin RL, Uzsoy R (2001) Experimental evaluation of heuristic optimization algorithms: a tutorial. J Heuristics 7(3):261–304

Reyes-Sierra M, Coello C (2007) A study of techniques to improve the efficiency of a multi-objective particle swarm optimizer. In: Studies in computational intelligence, vol 51. Springer, New York, pp 269–296

Richter H (2005) A study of dynamic severity in chaotic fitness landscapes. In: Proceedings of the IEEE Congress on evolutionary computation, vol 3, pp 2824–2831

Richter H, Yang S (2009) Learning behavior in abstract memory schemes for dynamic optimization problems. Soft Comput: A Fusion Found Methodol Appl 13(12):1163–1173

Rocco C, Salazar D (2007) A hybrid approach based on evolutionary strategies and interval arithmetic to perform robust designs. In: Studies in computational intelligence, vol 51. Springer, New York, pp 543–564

Rohlfshagen P, Yao X (2009) The dynamic knapsack problem revisited: a new benchmark problem for dynamic combinatorial optimisation. In: Applications of evolutionary computing, pp 745–754

Rohlfshagen P, Lehre PK, Yao X (2009) Dynamic evolutionary optimisation: an analysis of frequency and magnitude of change. In: Proceedings of the genetic and evolutionary computation conference, pp 1713–1720

Ronnewinkel C, Martinetz T (2001) Explicit speciation with few a priori parameters for dynamic optimization problems. In: GECCO workshop on evolutionary algorithms for dynamic optimization problems. Morgan Kaufmann, Massachusetts, pp 31–34

Rossi C, Abderrahim M, César Díaz J (2008) Tracking moving optima using Kalman-based predictions. Evol Comput 16(1):1–30

Saleem S, Reynolds R (2000) Cultural algorithms in dynamic environments. In Proceedings of the Congress on evolutionary computation, vol 2, pp 1513–1520

Schönemann L (2004) The impact of population sizes and diversity on the adaptability of evolution strategies in dynamic environments. In: Proceedings of the IEEE Congress on evolutionary computation, vol 2, pp 1270–1277

Schönemann L (2007) Evolution strategies in dynamic environments. In: Studies in computational intelligence, vol 51. Springer, New York, pp 51–77

Sheskin DJ (2004) Handbook of parametric and nonparametric statistical procedures. CRC Press, Boca Raton

Shi Y, Eberhart R (2001) Fuzzy adaptive particle swarm optimization. In: Proceedings of the IEEE conference on evolutionary computation

Simões A, Costa E (2003) An immune system-based genetic algorithm to deal with dynamic environments: diversity and memory. In: Pearson DW, Steele NC, Albrecht R (eds) Proceedings of the sixth international conference on neural networks and genetic algorithms (ICANNGA03). Springer, New York, pp 168–174

Smierzchalski R, Michalewicz Z (2000) Modeling of ship trajectory in collision situations by an evolutionary algorithms. IEEE Trans Evol Comput 4:227–241

Stanhope S, Daida J (1999) (1+1) Genetic algorithm fitness dynamics in a changing environment. In Proceedings of the IEEE Congress on evolutionary computation, vol 3, pp 1851–1858

Tenne Y, Armfield S (2007) A memetic algorithm using a trust-region derivative-free optimization with quadratic modelling for optimization of expensive and noisy black-box functions. In: Studies in computational intelligence, vol 51. Springer, New York, pp 389–415

Tezuka M, Munetomo M, Akama K (2007) Genetic algorithm to optimize fitness function with sampling error and its application

to financial optimization problem. In: Studies in computational intelligence, vol 51. Springer, New York, pp 417–434
Tinós R, Yang S (2007a) Genetic algorithms with self-organizing behaviour in dynamic environments. In: Studies in computational intelligence, vol 51. Springer, New York, pp 105–127
Tinós R, Yang S (2007b) A self-organizing random immigrants genetic algorithm for dynamic optimization problems. Genet Program Evolvable Mach 8(3):255–286
Tinós R, Yang S (2007c) Continuous dynamic problem generators for evolutionary algorithms. In: Proceedings of the IEEE Congress on evolutionary computation, pp 236–243
Tinós R, Yang S (2008) Evolutionary programming with q-Gaussian mutation for dynamic optimization problems. In: Proceedings of the IEEE Congress on evolutionary computation, pp 1823–1830
Trojanowski K, Wierzchon ST (2009) Immune-based algorithms for dynamic optimization. Inf Sci 179(10):1495–1515
Tumer K, Agogino A (2007) Evolving multi rover systems in dynamic and noisy environments. In: Studies in computational intelligence, vol 51. Springer, New York, pp 371–387
Ursem RK (2000) Multinational GAs: multimodal optimization techniques in dynamic environments. In: Proceedings of the genetic and evolutionary computation conference. Morgan Kaufmann, Massachusetts, pp 19–26
Ursem RK, Krink T, Jensen M, Michalewicz Z (2002) Analysis and modeling of control tasks in dynamic systems. IEEE Trans Evol Comput 6(4):378–389
Venayagamoorthy G (2004) Adaptive critics for dynamic particle swarm optimization. In: IEEE international symposium on intelligent control
Wang H, Wang D, Yang S (2009a) A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems. Soft Comput: A Fusion Found Methodol Appl 13(8-9): 763–780
Wang H, Yang S, Ip W, Wang D (2009b) Adaptive primal-dual genetic algorithms in dynamic environments. IEEE Trans Syst Man Cybernet B 39(6):1348–1361
Weicker K (2002) Performance measures for dynamic environments. In: Parallel problem solving from nature VII. Lecture notes in computer science, vol 2439. Springer, New York, pp 64–73
Weicker K (2003) Evolutionary algorithms and dynamic optimization problems. Der Andere Verlag
Weicker K, Weicker N (1999) On evolution strategy optimization in dynamic environments. In: Proceedings of the IEEE Congress on evolutionary computation, pp 2039–2046
Wineberg M, Oppacher F (2000) Enhancing the GA's ability to cope with dynamic environments. In: Proceedings of the genetic and evolutionary computation conference. Morgan Kaufmann, Massachusetts, pp 3–10
Woldesenbet YG, Yen GG (2009) Dynamic evolutionary algorithm with variable relocation. IEEE Trans Evol Comput 13(3):500–513
Yan X-S, Kang L-S, Cai Z-H, Li H (2004) An approach to dynamic traveling salesman problem. In: International conference on machine learning and cybernetics
Yang S (2003) Non-stationary problem optimization using the primal-dual genetic algorithm. In: Proceedings of the IEEE Congress on evolutionary computation, vol 3. IEEE Press, pp 2246–2253
Yang S (2005) Memory-based immigrants for genetic algorithms in dynamic environments. In: Proceedings of the genetic and evolutionary computation conference. ACM, New York, pp 1115–1122
Yang S (2006a) Associative memory scheme for genetic algorithms in dynamic environments. In: Applications of evolutionary computing. Lecture notes in computer science, vol 3907. Springer, Berlin, pp 788–799
Yang S (2006b) A comparative study of immune system based genetic algorithms in dynamic environments. In: Proceedings of the genetic and evolutionary computation conference. ACM, New York, pp 1377–1384
Yang S (2007) Explicit memory schemes for evolutionary algorithms in dynamic environments. In: Studies in computational intelligence, vol 51. Springer, New York, pp 3–28
Yang S (2008) Genetic algorithms with memory- and elitism-based immigrants in dynamic environments. Evol Comput 16(3): 385–416
Yang S, Tinós R (2007) A hybrid immigrants scheme for genetic algorithms in dynamic environments. Int J Autom Comput 4(3): 243–254
Yang S, Yao X (2005) Experimental study on population-based incremental learning algorithms for dynamic optimization problems. Soft Comput: A Fusion Found Methodol Appl 9(11):815–834
Yang S, Yao X (2008) Population-based incremental learning with associative memory for dynamic environments. IEEE Trans Evol Comput 12(5):542–561
Yang S, Ong Y-S, Jin Y (2006) Editorial to special issue on evolutionary computation in dynamic and uncertain environments. Genet Program Evolvable Mach 7(4):293–294
Yang S, Ong Y-S, Jin Y (eds) (2007) Evolutionary computation in dynamic and uncertain environments. In: Studies in computational intelligence, vol 51. Springer, Berlin
Yang S, Cheng H, Wang F (2010) Genetic algorithms with immigrants and memory schemes for dynamic shortest path routing problems in mobile ad hoc networks. IEEE Trans Syst Man Cybernet C: Appl Rev 40(99):52–63
Yen G, Yang F, Hickey T, Goldstein M (2001) Coordination of exploration and exploitation in a dynamic environment. In: International joint conference on neural networks. Institute of Electrical and Electronics Engineers
Zeng S, Shi H, Kang L, Ding L (2007) Orthogonal dynamic hill climbing algorithm: ODHC. In: Studies in computational intelligence, vol 51. Springer, New York, pp 79–104
Zou X, Wang M, Zhou A, Mckay B (2004) Evolutionary optimization based on chaotic sequence in dynamic environments. In: IEEE international conference on networking, sensing and control, pp 1364–1369