Dynamic Multiobjective Optimization Problems: Test Cases, Approximations, and Applications

Marco Farina, Kalyanmoy Deb, and Paolo Amato

***Abstract*—After demonstrating adequately the usefulness of evolutionary multiobjective optimization (EMO) algorithms in finding multiple Pareto-optimal solutions for static multiobjective optimization problems, there is now a growing need for solving dynamic multiobjective optimization problems in a similar manner. In this paper, we focus on addressing this issue by developing a number of test problems and by suggesting a baseline algorithm. Since in a dynamic multiobjective optimization problem, the resulting Pareto-optimal set is expected to change with time (or, iteration of the optimization process), a suite of five test problems offering different patterns of such changes and different difficulties in tracking the dynamic Pareto-optimal front by a multiobjective optimization algorithm is presented. Moreover, a simple example of a dynamic multiobjective optimization problem arising from a dynamic control loop is presented. An extension to a previously proposed direction-based search method is proposed for solving such problems and tested on the proposed test problems. The test problems introduced in this paper should encourage researchers interested in multiobjective optimization and dynamic optimization problems to develop more efficient algorithms in the near future.**

***Index Terms*—Applications, dynamic fitness landscapes, evolutionary multiobjective optimization, test cases.**

I. Introduction

OVER THE past decade or so, multiobjective optimization literature has witnessed a radically different perspective in solving the problems using evolutionary computing methods compared with the classical methods. Since these problems involve a multitude of optimal solutions, known as Pareto-optimal solutions, evolutionary multiobjective optimization (EMO) methods attempt to find a widely distributed set of solutions as close to the true Pareto-optimal front (POF) as possible in a single simulation run. These approaches not only provide a good idea of the extent (ideal and nadir solutions) of the true POF but also provide information about the shape of the front [1] and existence of any "knee" solution [2]. They also allow users to investigate the obtained solutions to decipher any interesting properties of the optimal solutions [3]. Despite the usefulness of the EMO algorithms, there has been lukewarm interest in extending the ideas to solving dynamic multiobjective optimization problems. In this paper, we address this issue and suggest a suite of test problems for both continuous and discrete dynamic multiobjective optimization problems and also suggest a baseline algorithm for tackling such problems.

The EMO literature contains a large number of *static* (not changing during the course of optimization) test problems covering different types of difficulties which may be encountered by an EMO algorithm when converging toward the POF [4], [5]. As mentioned, these problems require a static optimization procedure, in which the task is to find a set of design variables to optimize the objective functions which are static, but several other important real-world applications require a time-dependent (on-line) multiobjective optimization, in which either the objective function and constraint or the associated problem parameters or both vary with time (iteration of the optimization process). In handling such problems, not many EMO algorithms exist, and certainly, there is a lack of test problems to adequately test a dynamic evolutionary multiobjective optimization (DEMO) algorithm.

Besides suggesting a set of test problems, in this paper we also discuss an adaptive control problem of a time-varying system, where the optimal controller is time-dependent because the system's properties are time-dependent. Moreover, we consider multiple objectives for the controller dynamic optimization, and we give a formulation of the resulting dynamic multiobjective optimization problem. Optimal design of controllers is a classical field of application for evolutionary computation (see [6] for a review of applications) and evolutionary multiobjective optimization. Once the closed-loop stability is assured, several additional criteria for performance enhancement can be considered such as maximum overshooting minimization, settling time minimization, and rise time minimization, in order to design stable and powerful controllers. Several examples of such an optimization procedure are available in the literature in the case of static problems, that is, when the optimization is to be performed offline and when the model of the system (the plant or the device) is not time dependent. Two early examples can be found in [7], where some controllers (among which an $\mathcal{H}_2/\mathcal{H}_\infty$ one) are optimized with an EMO algorithm. Another classical application of EMO for static controller optimization considers fuzzy rule set optimization for fuzzy controllers; some examples can be found in [8] and [9].

When considering dynamic single-objective optimization problems, the use of genetic algorithms (GAs) for a time-dependent fitness landscape [10] has been considered for single-objective problems, and several studies are available in the literature (see as examples [11]–[13]). Major modifications in the operators are required for a prompt reaction to time-dependent changing [13] because in the balance between convergence and exploration, a bigger weight is to be given to

Manuscript received July 3, 2003; revised December 17, 2003.

M. Farina and P. Amato are with STMicroelectronics, Agrate 20041, Italy (e-mail: marco.farina@st.com; paolo.amato@st.com).

K. Deb is with the Department of Mechanical Engineering, Kanpur Genetic Algorithms Laboratory (KanGAL), Indian Institute of Technology, Kanpur 208 016, India (e-mail: deb@iitk.ac.in).

Digital Object Identifier 10.1109/TEVC.2004.831456

the second feature, so that the algorithm can react promptly to time changes [14]–[17]. Moreover, several other strategies for dynamic optimization procedure for single-objective problems are also proposed in the literature. A promising method for dynamic optimization algorithm development seems to be the exploitation of an artificial life (A-life) paradigm (see [18] and [19] for a detailed overview) and the hybridization of it with evolutionary computation [20], [21]. The A-life is in fact a flexible and dynamic computational paradigm to mimic the natural search in dynamic environments [22]. Moreover, the hybridization between different computational paradigms [23] such as immune systems and GAs [24] seems to be particularly suited to the same scope. An additional example can be found in [25], where an A-life inspired algorithm was developed. However, when dynamic multiobjective optimization is concerned, very few studies are available in the literature [26], [27], and a complete formulation of the problem together with a set of adequate test problems is still missing.

In this paper, we make an attempt to fill this gap and suggest a test suite of five problems testing various aspects of tracking the Pareto-optimal set (POS) whenever there is a change in the problem. Hopefully, this paper will motivate researchers interested in dynamic optimization problems to develop and test their algorithms toward creating efficient dynamic multiobjective EAs.

II. Problem Definition

From the most general point of view, any dynamic multiobjective optimal control problem can be represented as the following parameterized multiobjective optimization problem.

Definition II.1: Let $\mathbf{V_O}$, $\mathbf{V_F}$, and $\mathbf{W}$ be n_O-dimensional, n_F-dimensional, and M-dimensional continuous or discrete vector spaces, $\mathbf{g}$ and $\mathbf{h}$ be two functions defining inequalities and equalities constrains, and $\mathbf{f}$ be a function from $\mathbf{V_O} \times \mathbf{V_F}$ to $\mathbf{W}$. A *parameterized multicriteria minimization problem* with M objectives is defined as

$$\begin{cases} \min\limits_{\mathbf{v_O} \in \mathbf{V_O}} \mathbf{f} = \{f_1(\mathbf{v_O}, \mathbf{v_F}), \ldots, f_M(\mathbf{v_O}, \mathbf{v_F})\} \\ \text{s.t. } \mathbf{g}(\mathbf{v_O}, \mathbf{v_F}) \leq 0,\ \mathbf{h}(\mathbf{v_O}, \mathbf{v_F}) = 0 \end{cases}.$$

In the problem defined above, some variables are available for optimization $(\mathbf{v}_O)$, and some others $(\mathbf{v}_F)$ are imposed parameters that are independent from the optimization variables. Both objective functions and constrains are parameter-dependent and can be nonlinear. A special case of the above problem is the following, where only one parameter—time t—is considered.

Definition II.2: Let t be the time variable, $\mathbf{V}$ and $\mathbf{W}$ be n-dimensional and M-dimensional continuous or discrete vector spaces, $\mathbf{g}$ and $\mathbf{h}$ be two functions defining inequalities and constraint equalities, and $\mathbf{f}$ be a function from $\mathbf{V} \times t$ to $\mathbf{W}$. A *dynamic multicriteria minimization problem* with M objectives is defined as

$$\begin{cases} \min\limits_{\mathbf{v} \in \mathbf{V}} \mathbf{f} = \{f_1(\mathbf{v}, t), \ldots, f_M(\mathbf{v}, t)\} \\ \text{s.t. } \mathbf{g}(\mathbf{v}, t) \leq 0,\ \mathbf{h}(\mathbf{v}, t) = 0 \end{cases}.$$

For the above problem, we define two sets of solutions which we shall mention throughout in this paper.

Definition II.3: We call the *POS at time* t $(\mathcal{S}_P(t))$ and the *POF at time* t $(\mathcal{F}_P(t))$ the set of Pareto-optimal solutions at time t in decision variable and objective spaces, respectively.

The ideal point (frequently called utopia point [28]) $\mathbf{U}$ is time-dependent in these problems and is defined as follows.

Definition II.4: Time-dependent utopia point

$$\mathbf{U}(t) = \left[\min_{\mathbf{v} \in \Omega_t} f_i(\mathbf{v}, t)\right] = [U_i(t)] \quad i = 1 : M \tag{1}$$

where $\Omega_t = \{\mathbf{v} \in \mathbf{V}, \text{s.t. } \mathbf{g}(\mathbf{v}, t) \leq 0, \mathbf{h}(\mathbf{v}, t) = 0\}$ is the time-dependent search space satisfying time-dependent constraint. The utopia point corresponds in the design space to the following $M \times N$ (N being the dimension of search space) matrix $[\mathbf{M}_v]$, where each line is the minimum in one objective and is, thus, defined as follows:

$$\mathbf{M}_v(i, :) = [\mathbf{v} \in \Omega_t, \text{ s.t. } f_i(\mathbf{v}) = U_i], \quad i = 1 : M. \tag{2}$$

Note that $[M_x]$ may have equal lines (e.g. lines i and j) if no clash holds between objective i and j. For two-objective problems, some primary information on bounds of the time-dependent POF $\mathcal{F}_P(t)$ may be easily obtained from the evaluation of the following $M \times M$ time-dependent payoff [28] matrix

$$[\mathbf{M}](t) = \begin{cases} U_i(t), & i = j \\ f_j(\mathbf{M}_v(i, :), t), & \text{otherwise} \end{cases}. \tag{3}$$

For the derivation of $\mathcal{F}_P(t)$ bounds from $[\mathbf{M}](t)$ in case of more than two objectives, refer to [28], where details are given for the static case. The extension to the dynamic case is also straightforward. Moreover, once the time-dependent payoff matrix is computed the time-dependent nadir point $\mathbf{R}(t)$ can be easily estimated through a straightforward extension of those formulas that are used in the static case [28]. As an example, the following approximated formula can be used:

$$\mathbf{R}_i(t) = \max_{j=1:M} \mathbf{M}(i, :), \quad i = 1 : M. \tag{4}$$

III. Test Problems

In this section, we suggest a number of test problems for dynamic multiobjective optimization for continuous and discrete search spaces. Unlike in the single-objective dynamic optimization problems, where the ordering criterion in decision space is trivial, here we are dealing with two distinct yet related spaces where an ordering criterion is to be considered: decision variable space and objective space. Such an increased complexity holds true for static problems and even more for dynamic problems, where there are four possible ways a problem can demonstrate a time-varying change.

- Type I) The POS (optimal decision variables) $\mathcal{S}_P$ changes, whereas the POF (optimal objective values) $\mathcal{F}_P$ does not change.
- Type II) Both $\mathcal{S}_P$ and $\mathcal{F}_P$ change.
- Type III) $\mathcal{S}_P$ does not change, whereas $\mathcal{F}_P$ changes.
- Type IV) Both $\mathcal{S}_P$ and $\mathcal{F}_P$ do not change, although the problem can change.

TABLE I
FOUR DIFFERENT TYPES OF A DEMO PROBLEM

	(POS) $\mathcal{S}_{\mathrm{P}}$	
(POF) $\mathcal{F}_{\mathrm{P}}$	No Change	Change
No Change	Type IV	Type I
Change	Type III	Type II

These four cases are summarized in Table I. There is, of course, a possibility that while the problem is changing, more types of above changes can occur simultaneously in the time scale. Moreover, in this paper, no attention is given to the decision maker's preferences, which may be time-dependent as well. Furthermore, new objectives may be added with iteration or deleted. Here, we concentrate on the first three types of changes, although we recognize that the Type IV change may also occur in some special cases. A Type IV change means a change in the system which does not make any change in the POS or POF.

Another aspect worth mentioning here is the rate of change of the problem with time. There may be a sudden change in the problem with a comparatively long *statis* thereafter, or there may be a gradual yet small change throughout the time scale. There is certainly a possibility of having both types of changes in a problem. Although the ability to track the POS for any of the above changes demands an adequate algorithm, here we concentrate on handling problems of the former kind in which there is a sudden change in a problem followed by a long stasis. Although the suggested test problems can be extended quite easily to develop other kinds of changes discussed above, we emphasize here that a completely different kind of algorithms than what is presented here would be required to track the corresponding time-varying POSs. We belabor these extensions for a future study.

A. Dynamic Test Problems from Static Test Problems

A straightforward extension of two-objective ZDT [29] and scalable DTLZ [5] test problems developed earlier is made here to construct dynamic test problems. Compared with other suggested test problems [30], these systematic static test problems allow researchers to investigate different hurdles, such as nonconvexity, discontinuity, deceptiveness, presence of local fronts, etc., which a real-world problem may have. When solving dynamic problems, such difficulties may transform themselves from one of the above features to another, providing an increasing difficulty to a multiobjective optimization algorithm.

A generic two-objective dynamic test problem can be presented using the construction procedure used in forming the static ZDT problems [4]

$$\begin{aligned}\text{minimize } f(\mathbf{x},t) = (&f_1(\mathbf{x}_I,t), g(\mathbf{x}_{II},t)\\ &\cdot h\left(\mathbf{x}_{III}, f_1(\mathbf{x}_I,t), g(\mathbf{x}_{II},t),t\right)) \quad (5)\end{aligned}$$

where $\mathbf{x}_I$, $\mathbf{x}_{II}$, and $\mathbf{x}_{III}$ are subsets of design variables set $\mathbf{x} \in \Omega_t$. In the above test problem, there are three functions f_1, g, and h. A typical functional form for each of the above for the static case is as follows:

$$\left.\begin{aligned} f_1\left(\mathbf{x}_I = \{x_1\}\right) &= x_1\\ g(\mathbf{x}_{II}) &= 1 + \sum_{x_i \in \mathbf{x}_{II}} x_i^2\\ h(\mathbf{x}_{III}, f_1, g) &= 1 - \sqrt{\frac{f_1}{g}} \end{aligned}\right\}. \quad (6)$$

Each of the above can be changed with time t. To make a more difficult problem, a mapping procedure in which the true decision variable vector $\mathbf{y}$ is mapped into the variable vector $\mathbf{x}$ should be followed before using the above equation [4]. Let us consider the following different scenarios before we present the test problem suite.

Scenario 1: The function g is redefined as follows:

$$g(\mathbf{x}_{II},t) = 1 + \sum_{x_i \in \mathbf{x}_{II}} \left(x_i - G(t)\right)^2, \quad x_{II}^{\min} \le G(t) \le x_{II}^{\max} \quad (7)$$

while f_1 and h are kept fixed as above. Equation (7) makes $x_i = G(t)$ as the optimal solution for all $x_i \in \mathbf{x}_{II}$. Here, each variable $x_i \in \mathbf{x}_{II}$ lies in $[x_{II}^{\min}, x_{II}^{\max}]$. Since $G(t)$ changes with time, $\mathcal{S}_P$ changes with time as well. However, the resulting POF $\mathcal{F}_P$ does not change. Thus, the above change in ZDT functions will cause a Type I test problem. In order to track the dynamic POF, the variable subset $\mathbf{x}_{II}$ has to converge to the new G value every time there is a change. The construction of g above is such that for any value of x_i for which $x_i \neq G(t)$ will correspond to a dominated solution with respect to the true POF.

However, if the g function is changed as follows: $g(\mathbf{x}_{II}) = 1 + G(t) + \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(t))^2$ for $x_{II}^{\min} \le G(t) \le x_{II}^{\max}$ and for $G(t) \ge 0$, the POF $\mathcal{F}_P$ also changes, and the resulting problem becomes a Type II test problem. Although the requirement for an algorithm to track both the current-best $\mathcal{F}_P$ and $\mathcal{S}_P$ is to make $x_i = G(t)$ for all $x_i \in \mathbf{x}_{II}$, a check on either $\mathcal{F}_P$ or $\mathcal{S}_P$ will enable one to determine if the algorithm is able to track the new POF.

Scenario 2: Next, the h function can be changed as follows:

$$h(\mathbf{x}_{III}, f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{\left(H(t) + \sum_{x_i \in \mathbf{x}_{III}} (\mathbf{x}_i - H(t))^2\right)^{-1}}$$
$$x_{III}^{\min} \le H(t) \le x_{III}^{\max}. \quad (8)$$

Here, a third set of variables $\mathbf{x}_{III}$ (with $x_i \in [x_{III}^{\min}, x_{III}^{\max}]$) is introduced to change h with time. Functions f_1 and g are not changed. The resulting $\mathcal{F}_P$ now changes through a change in $\mathbf{x}_{III}$ only (instead, through $\mathbf{x}_{II}$ required in scenario 1 above). This is also a Type II test problem.

Interestingly, if g and f_1 functions are not changed and h is changed as follows:

$$h(f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{H(t)}, \quad H(t) > 0 \quad (9)$$

the POS $\mathcal{S}_P$ does not change, but the POF $\mathcal{F}_P$ changes. This makes the corresponding problem a Type-III problem. Although the optimal $\mathcal{S}_P$ does not need to be changed for such problems to

remain on the new $\mathcal{F}_P$, there may be an effect of the distribution of the solution on the new POF for the old solutions. Therefore, if a well-spread set of solutions is desired every time there is a change in H function, a new decision variable vector $\mathbf{x}$ is required to be found. Thus, although this function is termed as a Type III problem, there will be some changes in the distribution of solutions in $\mathcal{S}_P$.

Scenario 3: The f_1 function can be changed as follows:

$$f_1(\mathbf{x}_I, t) = \sum_{x_i \in \mathbf{x}_I} x_i^{F(t)}, \quad F(t) > 0. \tag{10}$$

By varying $F(t)$, the density of solutions on the POF $\mathcal{F}_P$ can be varied. Since the objective of an multiobjective evolutionary algorithm (MOEA) would still be to find a well distributed set of Pareto-optimal solutions on the new front, the resulting $\mathbf{x}_I$ set (of finite size) needs to be defined in a different way from before. Since a change in $F(t)$ does not change the location of the POF, or $\mathcal{F}_P$, the resulting problem is a Type I problem.

Scenario 4: A more complex problem can be formed by changing all three functions by simply varying $G(t)$, $H(t)$, and $F(t)$. This will result in a Type II test problem. Since both $\mathcal{S}_P$ and $\mathcal{F}_P$ will be known in each case, it will be easier to test the performance of an MOEA. The functions suggested above can also be changed with more complex functions to construct a more difficult test problem.

We now discuss how the scalable static DTLZ test problems [5] can be used to construct scalable dynamic multiobjective test problems in a similar manner. For completeness, we first present a typical static M-objective, n-variable DTLZ function (say DTLZ2 problem) in the following:

$$\left.\begin{array}{ll} \text{Min.} & f_1(\mathbf{x}) = (1+g(\mathbf{x}_M))\cos\left(\frac{x_1\pi}{2}\right)\cdots\cos\left(\frac{x_{M-1}\pi}{2}\right) \\ \text{Min.} & f_2(\mathbf{x}) = (1+g(\mathbf{x}_M))\cos\left(\frac{x_1\pi}{2}\right)\cdots\sin\left(\frac{x_{M-1}\pi}{2}\right) \\ \vdots & \vdots \\ \text{Min.} & f_M(\mathbf{x}) = (1+g(\mathbf{x}_M))\sin\left(\frac{x_1\pi}{2}\right) \\ \text{with} & g(\mathbf{x}_M) = \sum_{x_i\in\mathbf{x}_M}(x_i-0.5)^2 \\ & 0 \le x_i \le 1, \text{ for } i = 1,2,\ldots,n \end{array}\right\}. \tag{11}$$

For a three-objective version of the above problem, the POF is an octant of a sphere of radius one residing on the first quadrant of the $f_1-f_2-f_3$ coordinate system and satisfying $\sum_i^M f_i^2 = 1$. Let us now discuss different ways to convert such a problem into a dynamic one.

Scenario 5: First, the $g(x_M)$ function can be changed as follows:

$$g(\mathbf{x}_M, t) = G(t) + \sum_{x_i \in \mathbf{x}_M} (x_i - G(t))^2, \quad 0 \le G(t) \le 1. \tag{12}$$

This will change both $\mathcal{S}_P$ and $\mathcal{F}_P$, thereby making the modified problem a Type II problem. For other DTLZ functions in which a different g function was used (such as in DTLZ1 and DTLZ3), they can also be changed accordingly. With the above change in the g function on DTLZ5 and DTLZ6 problems (which degenerated to have a curve as the POF, as opposed to a three-dimensional hypersphere), the $\mathcal{F}_P$ will get changed in an interesting way. Since the g function involves variables x_2 to x_{M-1}, the resulting POF may not lead to a curve, instead, for $G(t) > 0$, it will lead to a hypersurface, thereby causing a DEMO difficulty in expanding from a curve to a surface for a change in the g function.

Scenario 6: The change of x_i to $x_i^{F(t)}$ (where $F(t) \in (0,1]$) for all variables would be another interesting modification. Such a problem but with a fixed modification ($F(t) = 0.1$) was used in constructing the DTLZ4 function. This will give rise to a Type I test problem. Since this modification will cause a change of density of solutions over the POF and in the search space, the task of finding a well-distributed set of Pareto-optimal solutions every time there is a change would be a challenging task of a DEMO. The h function in DTLZ7 function can be modified, as described in ZDT functions above.

Scenario 7: The spherical shape of the POF in DTLZ2 to DTLZ6 can be changed to an ellipsoidal shape by changing the $(1+g(\mathbf{x}_M))$ term in every f_i expression by $(1+g(\mathbf{x}_M)+K_i(t))$, where $0 \le K_i(t) \le 1$ and $g(\mathbf{x}_M)$ is defined in (12). Such a change will lead to a Type II test problem. The POF will then be defined by $\sum_{i=1}^M [f_i/(1+K_i(t))]^2 = 1$. The change of a spherical hypersurface to ellipsoidal hypersurface will require a different set of solutions to maintain a good diversity and will remain a challenging task for a DEMO.

B. Test Suite for Continuous Search Space

Based on the above scenarios, we now suggest a test suite of five test problems involving ZDT and DTLZ test problems. However, these test problems can be used as a representative set of test problems in a study. Interested readers may construct other more interesting problems using the above construction procedure.

Definition III.1—FDA1 (Fig. 1): Type I, convex POF

$$\left\{\begin{array}{l} f_1(\mathbf{x}_I) = x_1 \\ g(\mathbf{x}_{II}) = 1 + \sum_{x_i\in\mathbf{x}_{II}} (x_i - G(t))^2 \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ G(t) = \sin(0.5\pi t), \quad t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_T} \right\rfloor \\ \mathbf{x}_I = (x_1) \in [0,1], \quad \mathbf{x}_{II} = (x_2,\ldots,x_n) \in [-1,1] \end{array}\right. . \tag{13}$$

This test problem is an instantiation of scenario 1 above.

Here, τ is the generation counter, τ_T is the number of generation for which t remains fixed, and n_t is the number of distinct steps in t. We suggest the following parameter values: $n = 20$, $\tau_T = 5$, and $n_T = 10$.

The task of a dynamic MOEA would be to find the same POF $f_2 = 1 - \sqrt{f_1}$ every time there is a change in t. A horizontal line in the top plot in Fig. 3 represents how the first two variables of all $\mathcal{S}_P(t)$ solutions vary in a particular time instantiation of the test problem. At the first time instant, $\mathcal{S}_P(t)$ corresponds to $x_i = 0$ for $\mathbf{x}_{II}$. With time t, the $\mathcal{S}_P$ is changed, and every variable in $\mathbf{x}_{II}$ must change a sinusoidal manner for a change in $G(t)$. Fig. 3 shows such a change for variable x_2 with time. Fig. 3 shows that although the $\mathcal{S}_P(t)$ change from one time instant to another, the corresponding $\mathcal{F}_P(t)$ does not change with time.

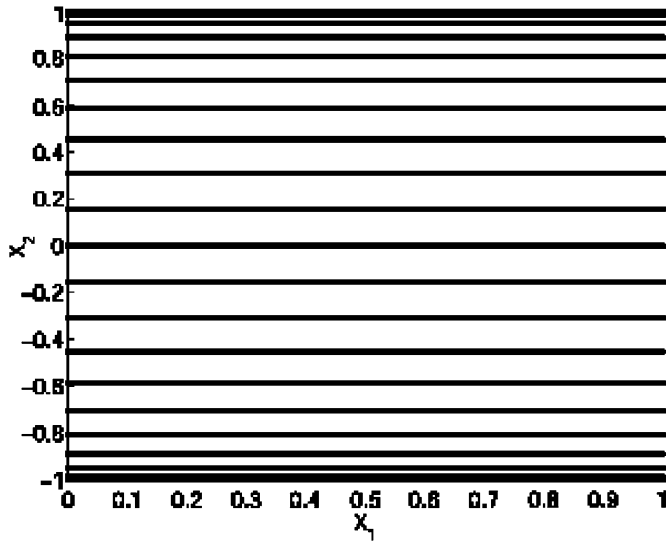


Fig. 1. $\mathcal{S}_P(t)$ for FDA1. Variations on only the first two decision variables are shown for 24 time steps. The corresponding $\mathcal{F}_P(t)$ is shown in Fig. 2.

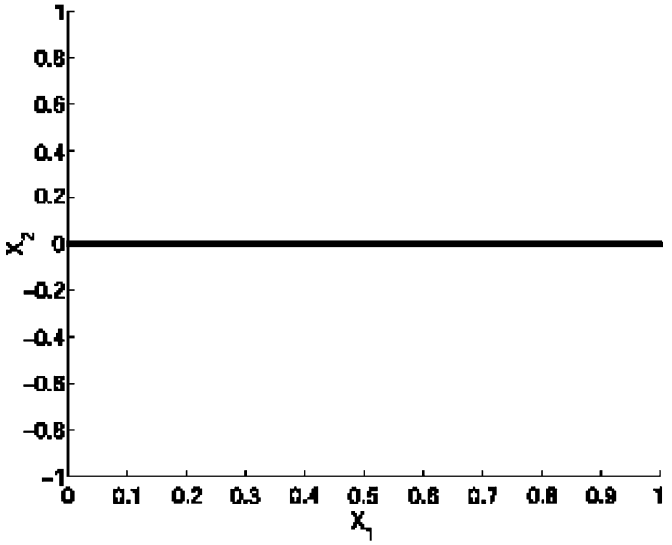


Fig. 3. $\mathcal{S}_P(t)$ for FDA2. Variations on only the first two decision variables are shown for 24 time steps. The corresponding $\mathcal{F}_P(t)$ is shown in Fig. 4.

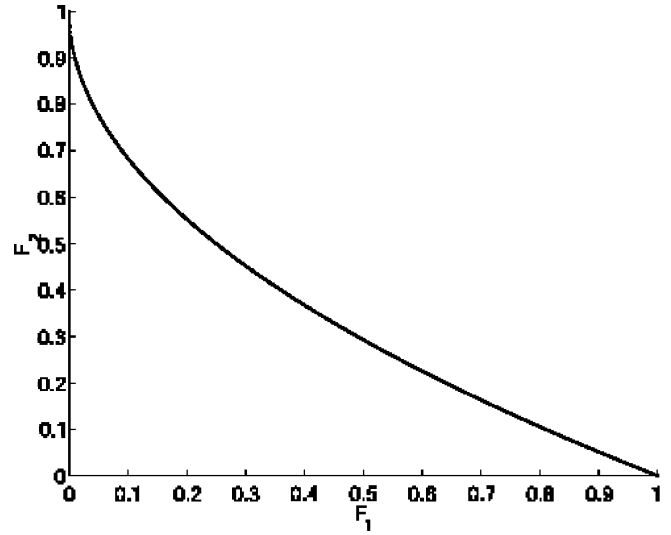


Fig. 2. $\mathcal{F}_P(t)$ for FDA1; the corresponding $\mathcal{S}_P(t)$ on only the first two decision variables are shown for 24 time steps in Fig. 1.

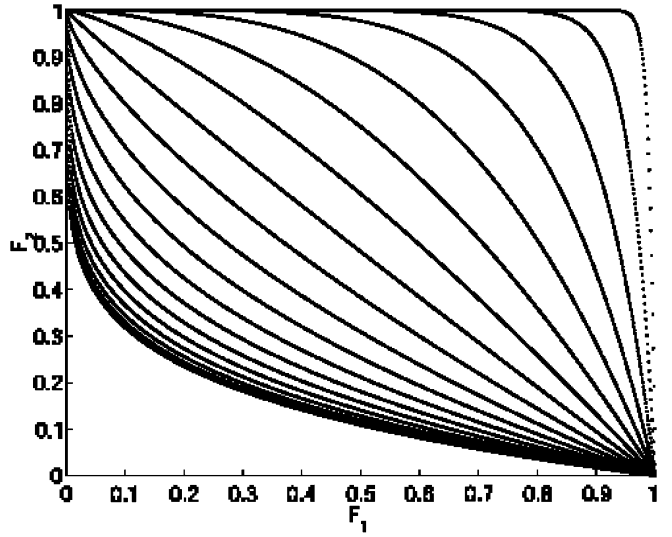


Fig. 4. $\mathcal{F}_P(t)$ for FDA2; the corresponding $\mathcal{S}_P(t)$ on only the first two decision variables are shown for 24 time steps in Fig. 3.

Definition III.2—FDA2: Type III, convex to nonconvex POFs

$$\begin{cases} f_1(\mathbf{x}_I) = x_1 \\ g(\mathbf{x}_{II}) = 1 + \sum\limits_{x_i \in \mathbf{x}_{II}} x_i^2 \\ h(\mathbf{x}_{III}, f_1, g) = 1 - \left(\frac{f_1}{g}\right) \left(H(t) + \sum\limits_{x_i \in \mathbf{x}_{III}} (\mathbf{x}_i - H(t))^2 \right)^{-1} \\ H(t) = 0.75 + 0.7 \sin(0.5\pi t), \quad t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_T} \right\rfloor \\ \mathbf{x}_I = (x_1) \in [0,1], \quad \mathbf{x}_{II}, \mathbf{x}_{III} \in [-1,1] \end{cases} . \tag{14}$$

This is an instantiation of scenario 2 discussed in the previous subsection. We recommend $|\mathbf{x}_{II}| = |\mathbf{x}_{III}| = 15$ in this test problem. Other parameters are the same as in FDA1. Here, the POF swings from a convex to a nonconvex shape due to the change in the $H(t)$ function, as shown in Fig. 4, while the corresponding $\mathcal{S}_P(t)$ (Fig. 3) remains unchanged. It is important to realize that a change in the shape of the $\mathcal{F}_P$ requires a change in the distribution of $\mathcal{S}_P$ (in this case, only in x_1 variable) to obtain a good distribution of solutions on the $\mathcal{F}_P$.

Definition III.3—FDA3 (Fig. 5): Type II, convex POFs

$$\begin{cases} f_1(\mathbf{x}_I) = \sum\limits_{x_i \in \mathbf{x}_I} x_i^{F(t)} \\ g(\mathbf{x}_{II}) = 1 + G(t) + \sum\limits_{x_i \in \mathbf{x}_{II}} (x_i - G(t))^2 \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ G(t) = |\sin(0.5\pi t)| \\ F(t) = 10^{2\sin(0.5\pi t)}, \quad t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_T} \right\rfloor \\ \mathbf{x}_I \in [0,1] \quad \mathbf{x}_{II} \in [-1,1] \end{cases} . \tag{15}$$

This is an instantiation of scenario 3 discussed earlier.

Here, we recommend $|\mathbf{x}_I| = 5$ and $|\mathbf{x}_{II}| = 25$. In this test problem, the density of solutions on the POF varies with t, as shown in Fig. 6. Here, both $\mathcal{S}_P(t)$ and $\mathcal{F}_P(t)$ change with time.

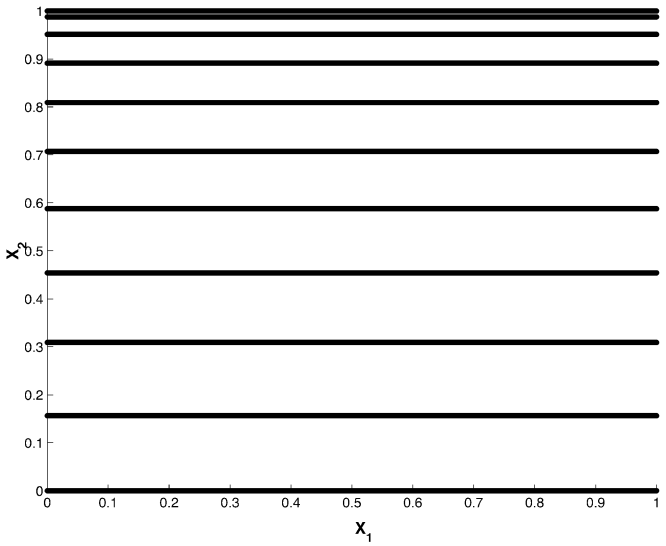


Fig. 5. $\mathcal{S}_P(t)$ for FDA3. Variations on only the first two decision variables are shown for 12 time steps. The corresponding $\mathcal{F}_P(t)$ is shown in Fig. 6.

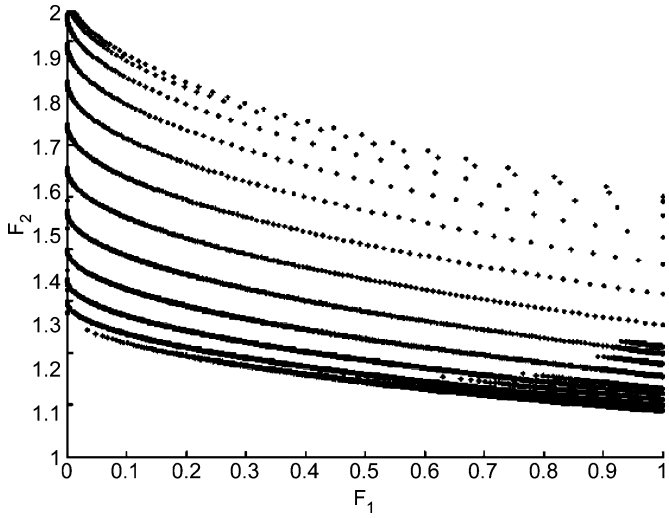


Fig. 6. $\mathcal{F}_P(t)$ for FDA3; the corresponding $\mathcal{S}_P(t)$ on only the first two decision variables are shown for 12 time steps in Fig. 5.

The task of an MOEA in FDA3 would be to find a widely distributed set of solutions every time there is a change in t, as shown in Fig. 6.

Definition III.4—FDA4 (Fig. 7): Type I, nonconvex POFs

$$\begin{cases} \min\limits_{\mathbf{x}} & f_1(\mathbf{x}) = (1+g(\mathbf{x}_{II})) \prod\limits_{i=1}^{M-1} \cos\left(\frac{x_i\pi}{2}\right) \\ \min\limits_{\mathbf{x}} & f_k(\mathbf{x}) = (1+g(\mathbf{x}_{II})) \left(\prod\limits_{i=1}^{M-k} \cos\left(\frac{x_i\pi}{2}\right)\right) \\ & \sin\left(\frac{x_{M-k+1}\pi}{2}\right), \quad k = 2:M-1 \\ \min\limits_{\mathbf{x}} & f_M(\mathbf{x}) = (1+g(\mathbf{x}_{II})) \sin\left(\frac{x_1\pi}{2}\right) \\ \text{where} & g(\mathbf{x}_{II}) = \sum\limits_{x_i \in \mathbf{x}_{II}} (x_i - G(t))^2 \\ & G(t) = |\sin(0.5\pi t)|, \quad t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_T} \right\rfloor \\ & \mathbf{x}_{II} = (x_M, \ldots, x_n), \\ & x_i \in [0,1] \quad i = 1:n \end{cases} . \quad (16)$$

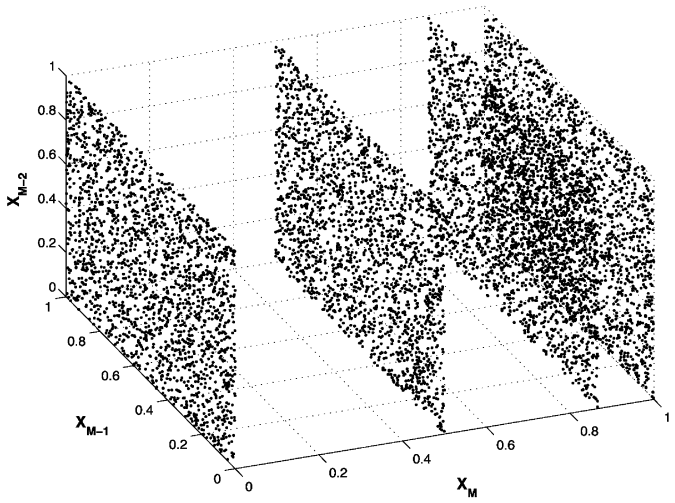


Fig. 7. $\mathcal{S}_P(t)$ for FDA4 with $(M-2)$th, $(M-1)$th, and Mth decision variables for four time steps. The corresponding $\mathcal{F}_P(t)$ is shown in Fig. 8.

This is an instantiation of scenario 5 discussed earlier.

We recommend $n = M + 9$, thereby keeping $|\mathbf{x}_{II}| = 10$. Fig. 8 shows the variation of $\mathcal{S}_P(t)$ and $\mathcal{F}_P(t)$ for this test problem. Since only $\mathcal{S}_P(t)$ changes with time, this is a Type I problem. Here, the task of a dynamic MOEA would be to find the same spherical surface (with radius one) whenever there is a change t. One aspect of such test problems is that just by observing whether the $\mathcal{F}_P(t)$ lies on the desired front, the optimality of the solutions can be ascertained.

Definition III.5—FDA5: Type II, nonconvex POFs

$$\begin{cases} \min\limits_{\mathbf{x}} & f_1(\mathbf{x}) = (1+g(\mathbf{x}_{II})) \prod\limits_{i=1}^{M-1} \cos\left(\frac{y_i\pi}{2}\right) \\ \min\limits_{\mathbf{x}} & f_k(\mathbf{x}) = (1+g(\mathbf{x}_{II})) \left(\prod\limits_{i=1}^{M-k} \cos\left(\frac{y_i\pi}{2}\right)\right) \\ & \sin\left(\frac{y_{M-k+1}\pi}{2}\right) \quad k = 2:M-1 \\ \min\limits_{\mathbf{x}} & f_M(\mathbf{x}) = (1+g(\mathbf{x}_{II})) \sin\left(\frac{y_1\pi}{2}\right) \\ \text{where} & g(\mathbf{x}_{II}) = G(t) + \sum\limits_{x_i \in \mathbf{x}_{II}} (x_i - G(t))^2 \\ & y_i = x_i^{F(t)} \quad \text{for } i = 1, \ldots, (M-1) \\ & G(t) = |\sin(0.5\pi t)| \\ & F(t) = 1 + 100\sin^4(0.5\pi t) \\ & t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_T} \right\rfloor \\ & \mathbf{x}_{II} = (x_M, \ldots, x_n), \quad x_i \in [0,1], \quad i = 1:n. \end{cases} \quad (17)$$

This is an instantiation of scenario 6 discussed earlier.

Identical parameter values as those in FDA4 can be used here. Here, the density of solutions on the POF changes with t, as shown in Fig. 10. The distribution achieved with a previously found good distribution will no more be a good one. The task of an dynamic MOEA would be to find a good distribution every time there is a change in the density of solutions.

C. Discrete Search Space Test Problems

A discrete version of the above test problems can be chosen by making all variables $(\mathbf{x})$ discrete. For example, in all problems, x_i can be assumed to take discrete values in steps of Δx

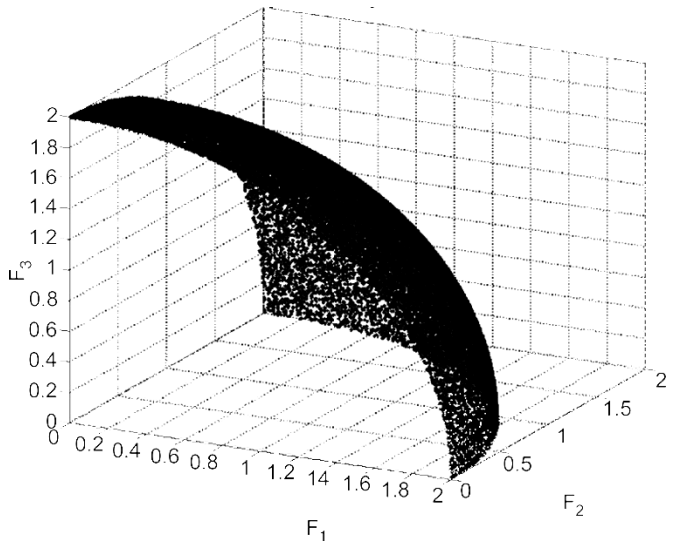


Fig. 8. $\mathcal{F}_P(t)$ for FDA. The corresponding $\mathcal{S}_P(t)$ plot with $(M-2)$th, $(M-1)$th, and Mth decision variables for four time steps is shown in Fig. 7.

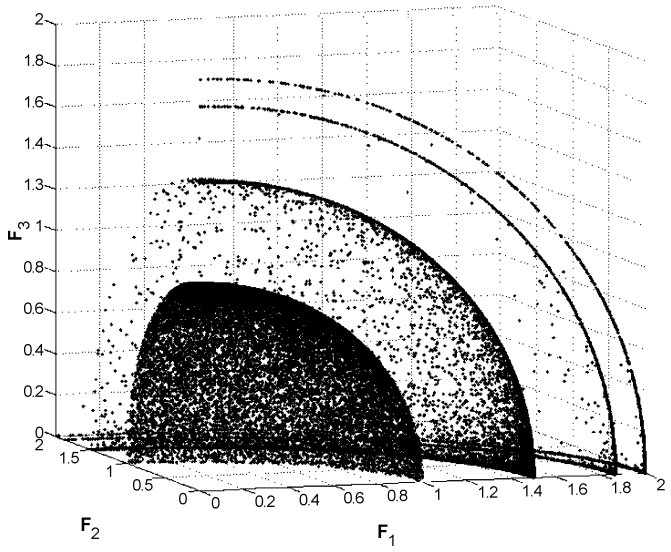


Fig. 10. $\mathcal{F}_P(t)$ (low) for FDA5 with $(M-2)$th, $(M-1)$th, and Mth decision variables for four time steps.

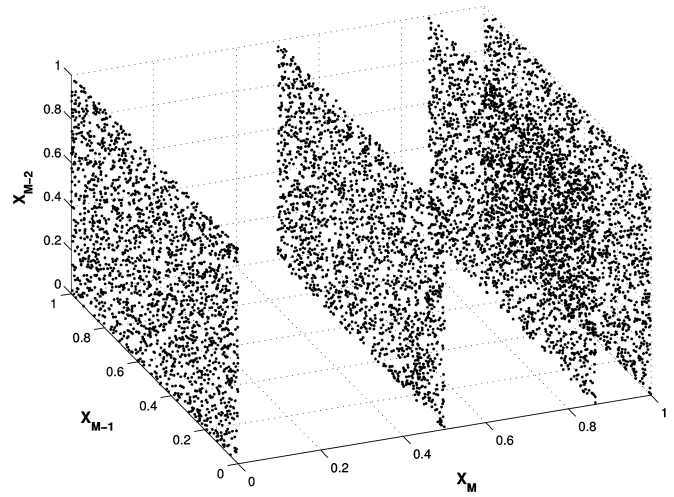


Fig. 9. $\mathcal{S}_P(t)$ for FDA5 with $(M-2)$th, $(M-1)$th, and Mth decision variables for four time steps.

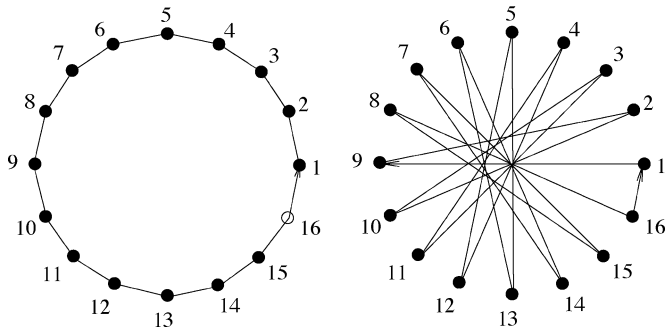


Fig. 11. Two extreme Pareto-optimal solutions.

(= 0.1 or 0.01 can be chosen) in the range [0,1]. To handle discrete variables, suitable genetic operators (recombination and mutation) must be used.

Static knapsack problems have been solved successfully using EMO techniques [31]. Dynamic knapsack problems having M knapsacks can also used as test problems

$$\begin{aligned} \text{Maximize} \quad & f_i(\mathbf{x},t) = \sum_{j=1}^{n} p_{ij}(t)x_j, \quad i = 1:M \\ \text{subject to} \quad & \sum_{j=1}^{n} w_{ij}(t)x_j \le c_i(t), \quad i = 1:M \\ & x_i \in \{0,1\}^n. \end{aligned} \tag{18}$$

Each decision variable x_i takes a value zero or one. If $x_i = 1$, the corresponding item is included in the knapsack. Each of the M knapsacks has a certain capacity c_i. The weight of item j in knapsack i is w_{ij}, and the corresponding profit is p_{ij}. In this problem, each of these quantities can be assumed to vary with time in a step-like manner, as discussed in the previous section. Typical values of the parameters which can be used in a test problem are as follows: $n \sim [250, 1000]$, w_{ij}, $p_{ij} \in [10, 100]$ and are integers, and $c_i \sim r(t)\sum_{j=1}^{n} w_{ij}$, where $r(t)$ is the fraction of total weight of the all n items. To start with, $r(t) = 0.5$ can be assumed and be changed thereafter in a predefined manner. It is intuitive to realize that a change in any or all of the quantities will make a change in the corresponding $\mathcal{F}_P$. However, an interesting problem can be created by just changing the c_i quantity for each knapsack after some time instant. An increase in c_i will cause more items to be included in the knapsacks, thereby making more profits and increasing the objective values. One difficulty in this problem is that the exact POF for each case cannot be derived easily, unlike the FDA test problems described earlier. However, for every new problem, a corresponding integer linear programming (ILP) problem can be solved using a standard and reliable software, and the obtained solutions can be used as known Pareto-optimal solutions.

The combinatorial optimization problems such as the traveling salesman problem (TSP) can also be considered as the prototype of optimal routing problems [32]. TSP problems are widely used for testing evolutionary algorithms and heuristic search strategies. In real-life problems, optimal routing may

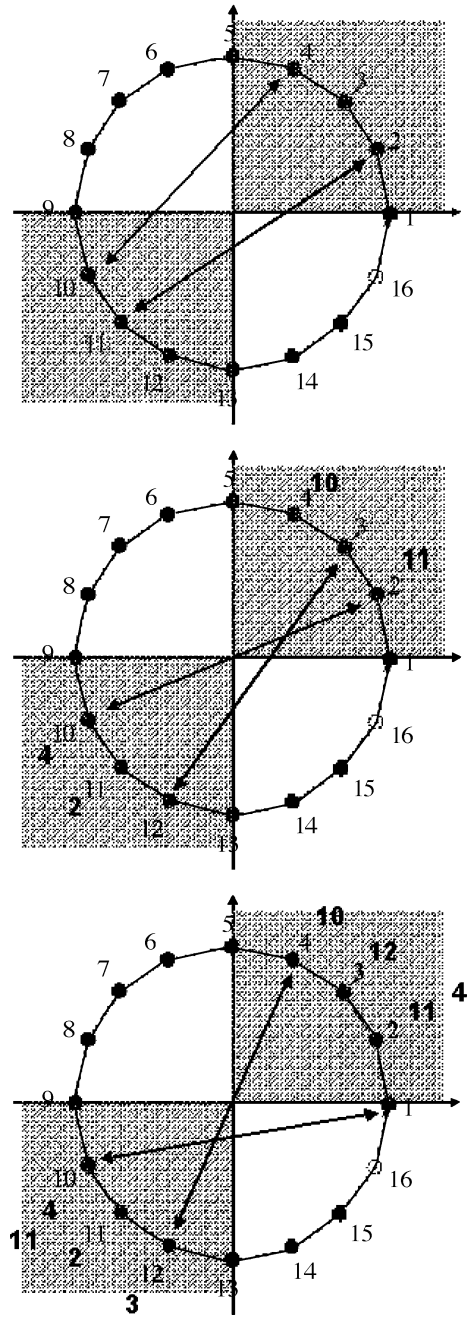


Fig. 12. Time changing of genotype for one particular Pareto-optimal solution. Arrows mean the changing in the next time step.

require several criteria to be considered and may be required in a dynamic situation, where features of the search space change during optimization. We, thus, consider an extension of the classical TSP as a benchmark for dynamic multiobjective optimization in discrete search spaces. The most general dynamic multiobjective n-city traveling salesman problem (DMTSP) (with $\mathbf{p}$ being a permutation of the cities) can be described as follows:

$$\begin{cases} \min\limits_{\mathbf{p}\in\mathbf{P^n}} \mathbf{f} = \{f_1(\mathbf{p},t),\ldots,f_M(\mathbf{p},t)\}, \quad \text{with} \\ f_j = \sum\limits_{i=1}^{n} w_{i,j}(p,t)\,\|\mathbf{x}(p_i,t)-\mathbf{x}(p_{i+1},t)\|, \quad p_{n+1}=p_1 \end{cases}. \tag{19}$$

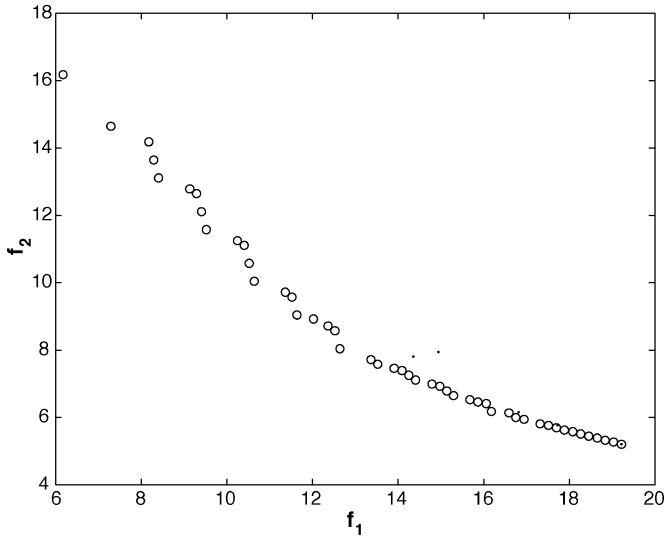


Fig. 13. Search space in objective domain and POF for an 11-city multiobjective TSP problem.

As can be seen from (19), both coefficients $w_{i,j}(p,t)$ and coordinates of the cities can be time-dependent; the first case may express different traffic conditions on the path (also depending on the permutation $\mathbf{p}$), while the second may represent a change in the location of the cities.

As a specific test case, we suggest a problem in which n cities are arranged equispaced on the circumference of a circle of radius one. This example is illustrated to paint a picture to the reader's mind about how a dynamic multiobjective TSP problem can be formulated with interesting yet tractable Pareto-optimal solutions. The following weight vectors are suggested:

$$\begin{aligned} w_{i1}(t) &= 1.0 \\ w_{i2}(t) &= \frac{1}{\|\mathbf{x}(p_i,t)-\mathbf{x}(p_{i+1},t)\|^2}. \end{aligned} \tag{20}$$

It is clear that the above two weight vectors will give rise to a set of Pareto-optimal solutions on which a successive arrangements of the cities and the diametric arrangement of cities are two extreme solutions (as shown in Fig. 11).

The initial arrangement of cities may be in any order. Thereafter, some cities in the first quadrant of the circle can get interchanged with the cities from the third quadrant. The number of cities and the choice of cities to be exchanged can be varied dynamically. Fig. 12 shows an exchange of two cities in three time steps changing the genotype of the schedule from one time instant to another. Since the weight vectors do not change, clearly, the POF ($\mathcal{F}_P$) also does not change. For example, the search space of all solutions (in dots) and the Pareto-optimal solutions (in circles) for a problem with 11 cities are shown in Fig. 13. The schedules of 48 different Pareto-optimal solutions are also shown in Fig. 14. Interestingly, two extreme solutions (shown in Fig. 11) exist in the set. The transition from one solution to the other through 46 other different solutions is quite interesting in this problem.

The above problem is a Type I DMTSP problem. Since the solution sequence changes with time, it would be the task of a DEMO to track the modified sequence of cities every time there

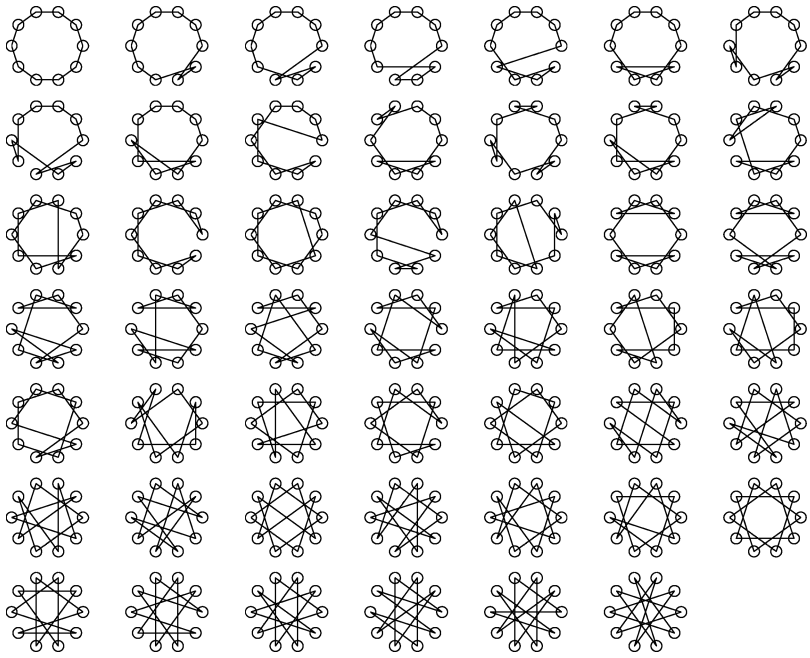


Fig. 14. Pareto-optimal routes for an 11-city multiobjective TSP problem: Routes are sorted using F_1.

is a change in arrangement of the cities. Since the POF does not change with time, it would be easy to test the performance of the DEMO.

D. Dynamic Multiobjective Optimal Control Problem

Finally, as a further example, closer to application, we discuss a dynamic controller optimization problem. Static controller optimization problems require the system to be known and fixed (but not time-dependent) during the entire optimal control procedure. This is quite often not true, and a simple example of this can be the aging of systems or the intrinsic randomness of some plants. In both cases, because the system changes with time, the controller is required to be adaptive in order for the closed-loop system performances to be satisfactory. As an example, consider the control of combustion in a rubbish burner [33], where the income properties (the rubbish to be burned) are typically random. We consider here a simplified test case of such a situation, that is, the control of a randomly varying system through a proportional-integral-derivative (PID) controller. The system's $(S(s))$ and controller's $(C(s))$ transfer functions are given as follows:

$$\begin{cases} S(s) = \frac{1.5}{50s^3+a_2(t)s^2+a_1(t)s+1} \\ C(s) = K_p(t) + K_i(t)\frac{1}{s} + K_d(t)s \end{cases} \tag{21}$$

where $a_1(t)$ and $a_2(t)$ are time-dependent parameters that can be varied in order to simulate the aforementioned aging or intrinsic random changes in the system. Moreover, in order to introduce some nonlinearity, two additional limits and rate blocks are inserted (see Fig. 15). The derivative coefficient in the PID controller has been fixed to $K_d = 8.3317$, and the other two coefficients $K_p(t)$ and $K_i(t)$ are to be varied in order for the closed-loop performances of the controlled system to be as similar as possible to a reference response in terms of a small rising time $R(t)$, a small maximum overshooting $O(t)$, and a small

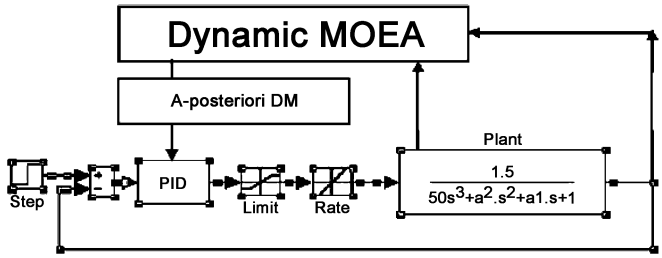


Fig. 15. Schema of the dynamic PID controller optimization test problem: three blocks are considered: the plant, the PID controller, and the dynamic multiobjective optimizer.

settling time $ST(t)$. The multiobjective optimization problem is, thus, the following:

$$\begin{cases} \min\limits_{(K_p(t),K_i(t))\in(0.5,5.0)\times(0.1,1.0)} \\ \{R\left(K_p(t),K_i(t)\right), O\left(K_p(t),K_i(t)\right), ST\left(K_p(t),K_i(t)\right)\} \end{cases}. \tag{22}$$

Fig. 15 shows the system and controller loop. The parameters $a_1(t)$ and $a_2(t)$ in the system transfer function can be varied in the following way:

$$a_1 = 3 + 30f(t), \quad a_2 = 43 + 30f(t) \tag{23}$$

where the function $-1 < f(t) < 1$ can be tuned for different time-dependent simulations. A sampling of the search space when only $O(t)$ and $R(t)$ are considered is shown in Fig. 16, together with the corresponding set of Pareto-optimal solutions for nine time steps with $f(t) = \sin(\pi t/18)$.

As can be seen, the shape of the POF changes quite significantly as time goes on. In snapshot one (upper left), the problem is even poorly multiobjective, while it becomes truly multiobjective in snapshot nine (lower right); moreover, the front moves from a nonconnected shape to a connected one. The dynamic

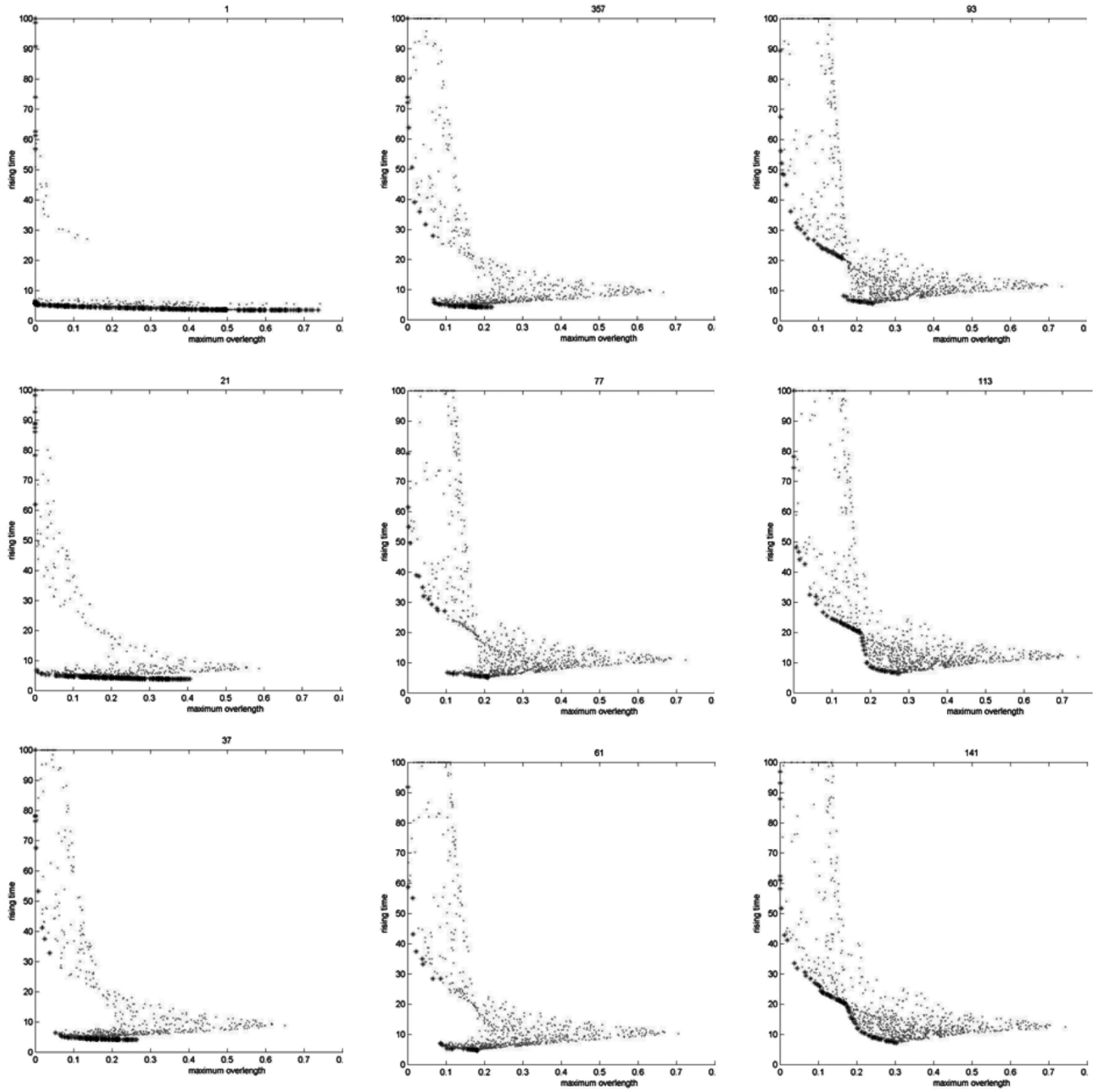


Fig. 16. Dynamic-POF [black ($\circ$)] and sampling [gray ($\cdot$)] for the dynamic PID controller optimization problem; snapshots at iteration 1, 21, 27, 38, 77, 81, 89, 113, and 141. The maximum overlength $O(K_p(t), K_d(t))$ and the rising time $R(K_p(t), K_d(t))$ are shown on the x and y axes, respectively. The shape of the POF changes quite significantly moving from a single-objective problem (step 1) to a disconnected multiobjective problem (steps 4–6) up to a nonconvex multiobjective one (steps 8 and 9).

MO optimizer has, thus, to be sufficiently flexible to approximate very different MO problems as time goes on.

We point out that, although sufficiently close to application for the scope of the paper, such a test case is still far from a concrete application of dynamic multiobjective optimization to dynamic control of real-life devices. The aim is mainly that of showing general principles rather than details on concrete application. Moreover, the use of PID control is not at all promoted as the best solution but is just an example, but the same general procedure may be applied to a different control strategy such as fuzzy control or H^{∞}. When thinking of a practical implementation of such a strategy with PID control, several problems are to be solved relating to the controller sensitivity and PID parameter tuning.

Another important issue (see Fig. 15) is the time-dependent *a posteriori* choice that is to be done in order to choose among PO solutions. This issue is similar to the static case, and the proper decision making rules are closely related to the physics and engineering of the device under consideration, and it is quite hard to give some problem-independent rules.

```
BEGIN
Start time t
Compute random starting pop.  pst(0)
WHILE t < t_S
    IF ε(t) > ε̃
        FOR i = 1 : M
            min.  f_i(t) with (1+1)-ES +
            GDA or NMA with pst(t)
            as starting population
        END
        compute U(t), R(t) and M̃(t)
        FOR k = 1 : n_s
            compute P^k(t) from P^{i=1:k-1}(t)
              and  S^{i=1:k-1*}(t)
            min.  f̃^k(t) with (1+1)ES +
              GBA or NMA with pst(t)
              as starting population
            obtain S^{k*}(t)
        END
        pst(t) = S*(t)
    END
    Compute ε(t)
END
```

Fig. 17. Pseudocode of the proposed strategy. $\widetilde{f}_j(t)$: Scalar objective functions corresponding to different directions, $P^k(t)$, $\widetilde{\mathbf{M}}(t)$, and $S^k(t)$: Parameters necessary for constructing $\widetilde{f}^k(t)$, GDA: Gradient-based single-objective optimization algorithm. NMA: Nelder Mead simplex single-objective optimization algorithm $pst(t)$: population at generation $t+1$.

IV. Dynamic EMO Algorithm: A Direction-Based Method

When EMO algorithms are considered for finding POS of time-dependent problems, the following two situations, or a combination of the two may happen.

- The time dependence is slow but continuous (mode A).
- The time dependence is seldom but sudden and random (mode B).

If the time dependence is of mode A, an extension to the multiobjective case of a single-objective evolutionary algorithm for variable fitness landscape [34] is to be considered (see [11], [12], and [14]–[16]), and some examples can be found in [26] and [27]. On the other hand, in the second case (mode B), a full evolution of an evolutionary algorithm is possible after the change has taken place and a time change monitoring strategy together with an update of the starting population may be sufficient for

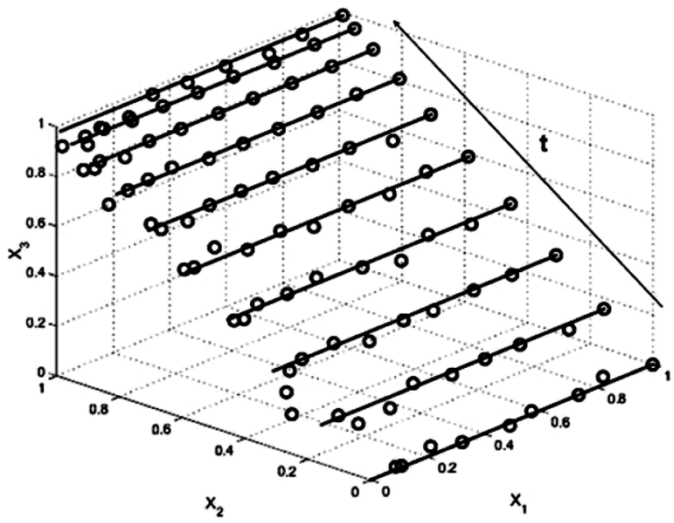


Fig. 18. Exact $\mathcal{S}_P$(dots) and approximated ($\circ$) with a search direction-based method on FDA1 (ten solutions).

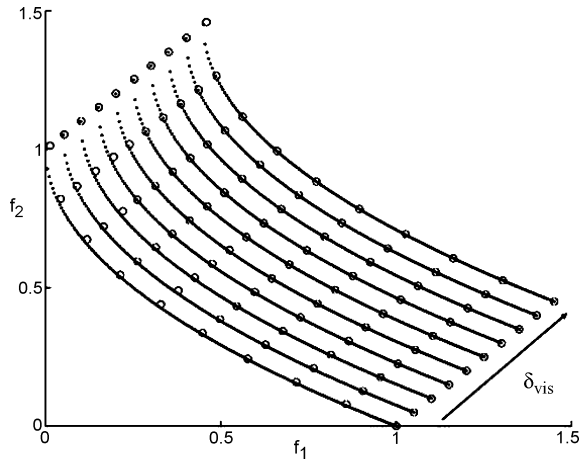


Fig. 19. Exact $\mathcal{F}_P$ (dots) and approximated ($\circ$) with the proposed search direction-based method on FDA1 (ten solutions).

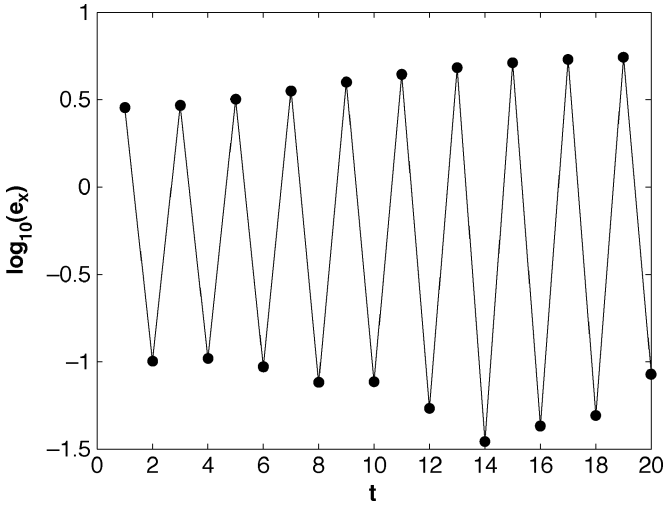


Fig. 20. FDA1: Log of POS convergence measure ($\log(e_x(t))$, (27)) versus time.

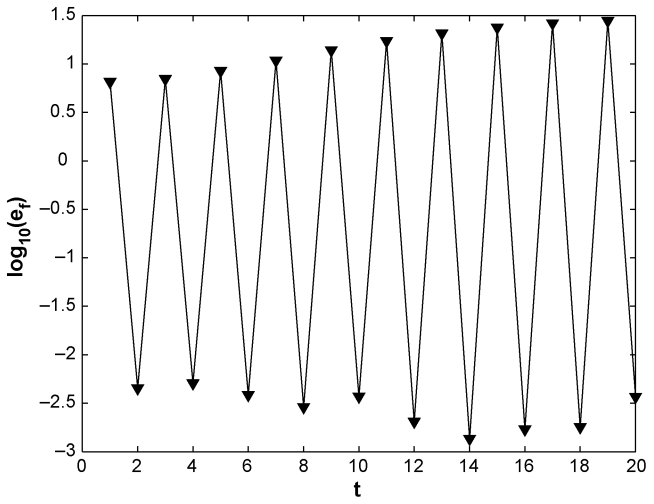


Fig. 21. FDA1: Log of POF convergence measure ($\log(e_f(t))$, (27)), versus time.

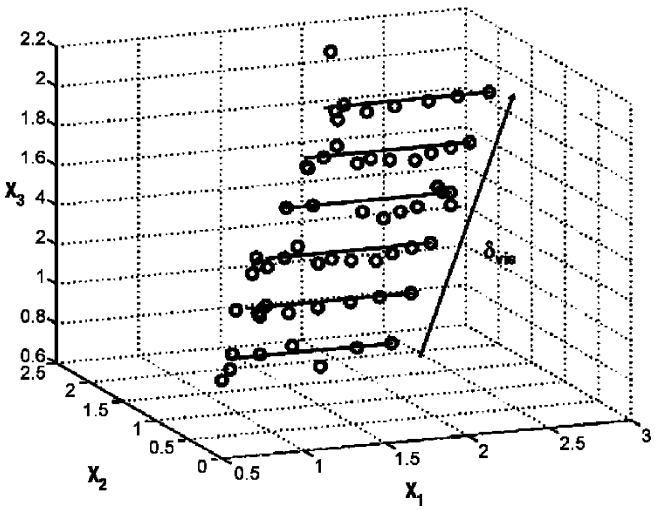


Fig. 22. Exact $\mathcal{S}_P$(dots) and approximated ($\circ$) with a search direction-based method on FDA2 (ten solutions).

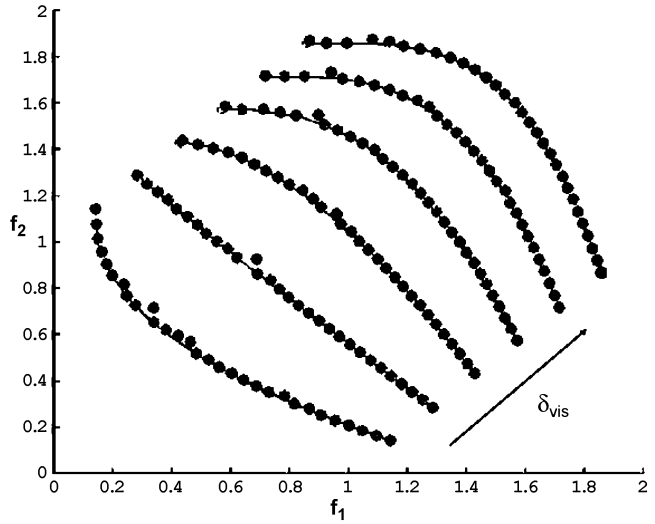


Fig. 23. Exact $\mathcal{F}_P$ (dots) and approximated ($\circ$) with the proposed search direction-based method on FDA2 (ten solutions).

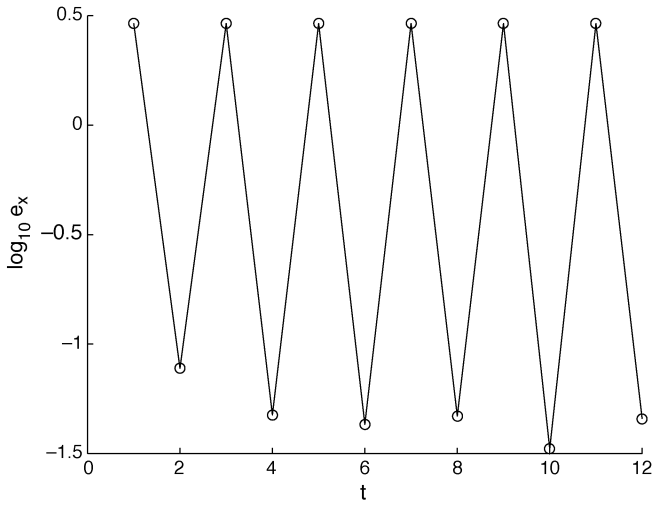


Fig. 24. FDA2: Log of POS convergence measure ($\log(e_x(t))$, (27)) versus time.

a proper approximation of the moving front and variable set. The proposed strategy is an immediate extension to the static search direction-based method described in [35]. We consider the case of a sudden random time-dependent change after which the system behaves as a static one for some time. A multiobjective search algorithm can, thus, be run in the time between one change to another. In order for the strategy to be fully automated, a check for any change in the system is added, using the following quantity:

$$\varepsilon(t) = \frac{\sum_{j=1}^{n_\varepsilon} \left\| \frac{(\mathbf{f}_j(X,t) - \mathbf{f}_j(X,t-1))}{\mathbf{R}(t) - \mathbf{U}(t)} \right\|}{n_\varepsilon} \tag{24}$$

where $\mathbf{R}(t)$ is the time-dependent nadir point, and $\mathbf{U}(t)$ is the time-dependent utopia point. A total of n_ε points in search space are chosen randomly for test problems or intentionally in case of real devices, where some working point may be particularly sensitive to time changes. If $\varepsilon(t) > \widetilde{\varepsilon}$ (a user-defined parameter) occurs, it means that a significant time change has taken place in the system, and a new search is to be carried out.

Such an automatic restarting procedure can be implemented with $10^{-3} < \widetilde{\varepsilon} < 10^{-2}$. More precise values for $\widetilde{\varepsilon}$ can be obtained only with some problem-dependent prediction on possible changes in the objective functions on a problem-to-problem basis. Moreover, a new computation of the utopia point U and nadir point R is required only when $\varepsilon(t) > \widetilde{\varepsilon}$.

A. Description of the Algorithm

A pseudocode of the whole strategy is shown in Fig. 17 (for more details, refer to [35]). The time is started $t = 0$, and a randomly distributed starting population $pst(0)$ is computed. After that, the following operations are performed iteratively until the final stopping time is reached ($t = t_S$). The quantity $\varepsilon(t)$ is computed using (24). If $\varepsilon(t) > \widetilde{\varepsilon}$ occurs (meaning that a significant change in the system has taken place), each objective function is minimized using a hybrid evolutionary-deterministic

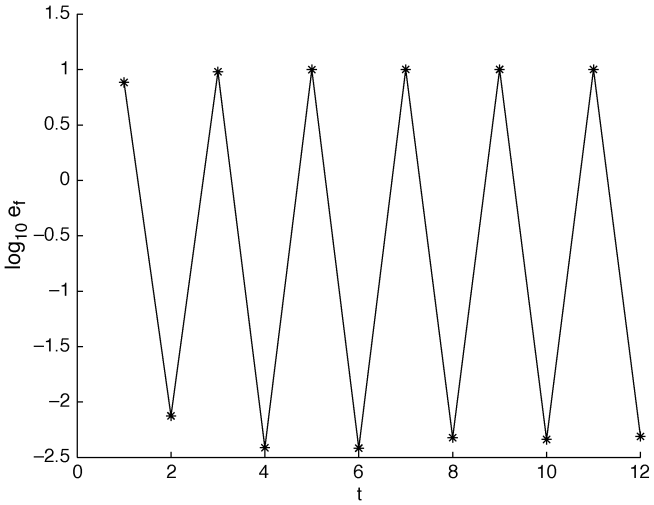


Fig. 25. FDA2: Log of POF convergence measure ($\log(e_f(t))$, (27)), versus time.

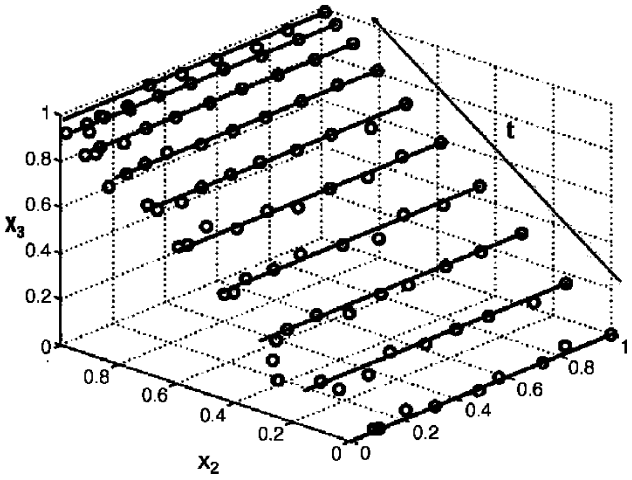


Fig. 26. Exact $\mathcal{S}_P$(dots) and approximated ($\circ$) with a search direction-based method on FDA3 (ten solutions).

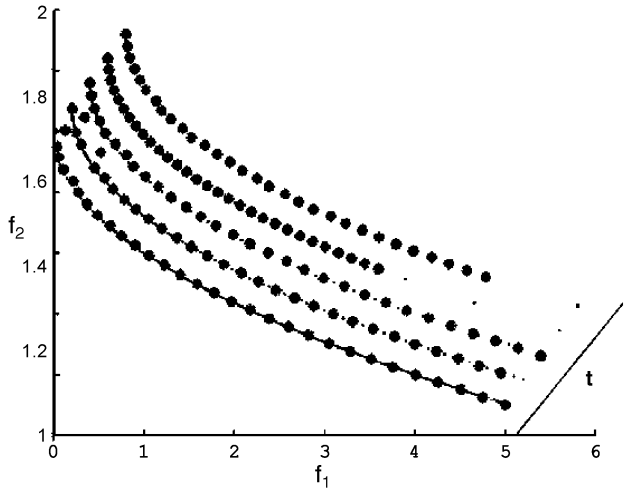


Fig. 27. Exact $\mathcal{F}_P$ (dots) and approximated ($\circ$) with the proposed search direction-based method on FDA3 (ten solutions).

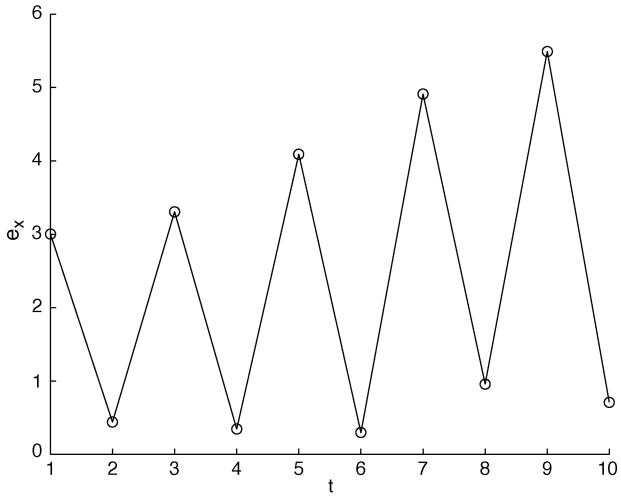


Fig. 28. FDA3: Log of POS convergence measure ($\log(e_x(t))$, (27)) versus time.

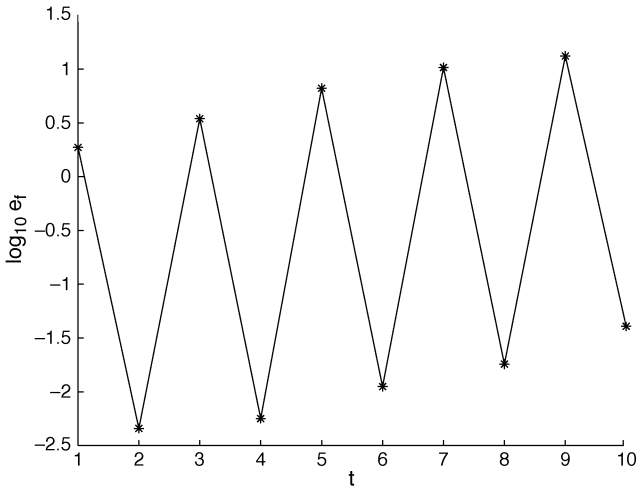


Fig. 29. FDA3: Log of POF convergence measure ($\log(e_f(t))$, (27)), versus time.

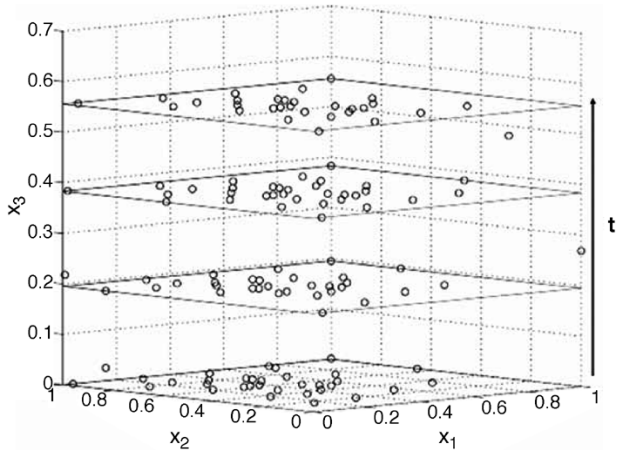


Fig. 30. Exact $\mathcal{S}_P$(wire-frame rectangle) and approximated ($\circ$) with a search direction-based method on FDA4 (ten solutions).

strategy with $pst(t)$ as a starting population. Such M independent optimizations help find the utopia point U, the nadir point R, and the payoff matrix $[\mathbf{M}]$. This is very similar to the practice usually followed in classical MCDM applications. Thereafter, n_s Pareto-optimal solutions are computed using n_s different

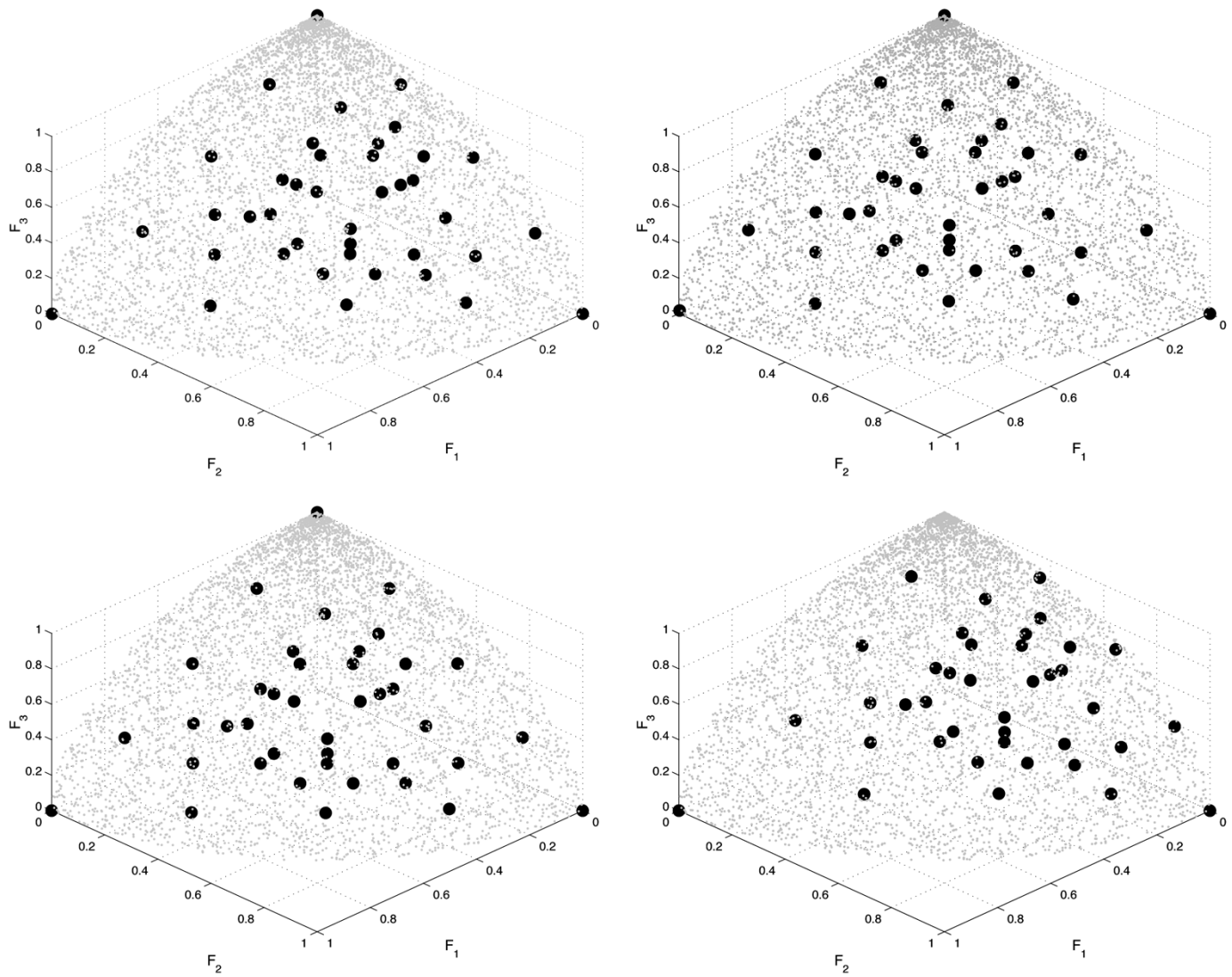


Fig. 31. Exact $\mathcal{F}_P$ (dots) and approximated ($\circ$) with the proposed search direction-based method on FDA4 (ten solutions).

scalar functions $\widetilde{f}^k$ (for the kth problem, where $k = 1 : n_s$) derived as follows:

$$\widetilde{f}^k(\mathbf{x}) = \max_{i=1:M} \left[w_i \frac{\left(f_i(\mathbf{x}) - P_i^k\right)}{R_i - U_i} \right] \tag{25}$$

where P_i^k is the ith objective value in the kth chosen center point, and w_i is a weight factor for the ith objective satisfying $\sum_{i=1}^{M} w_i = 1$. Here, we use $w_i = 1/M$. The above function is then minimized for each of the n_s center points independently. The diversity in obtained solutions is obtained by choosing a good distribution of centers P_i, described as follows. Once the utopia point U, nadir point R, and the matrix $\widetilde{\mathbf{M}}$ are computed, the line (two-objective problems) or the triangle (three-objectives problems) joining between the extremal point of the POF is considered, and a uniformly distributed set of points is built on it. For both the two- and three-objective problems, the following equation can be used to compute P^k vector for the kth center point

$$P^k = \sum_{j=1}^{M} w_{k,j} \widetilde{\mathbf{M}}(j,:), \quad \sum_{j=1}^{M} w_{k,j} = 1, \quad i = 1 : n_s. \tag{26}$$

Here, the vector $M(j,:)$ is the jth line of the matrix $\mathbf{M}$. Instead of prefixing the location of P^k center points, an iterative choice of centers P^k can also be considered (see the pseudocode in Fig. 17). For two-objective problems, we can start from the centroid of the two extreme points $\widetilde{M}_1$ and $\widetilde{M}_2$ as the first center P^1, and the objective solution S^{1^*} is obtained by maximizing the function given in (25). After that, two center points P^2 and P^3 are computed as centroids of the following pairs of points: $(\widetilde{M}_1, S^{1^*})$ and $(\widetilde{M}_2, S^{1^*})$. With these two new center points, two optimal solutions S^{2^*} and S^{3^*} are obtained using (25). In the case of three-objectives problems, a similar iterative strategy can be followed. The advantage of the iterative scheme is that latter optimizations will not require many iterations of the optimization process, as the center points may be already closer to the desired Pareto-optimal solutions. It is interesting to note that both the original scheme (26) and the iterative scheme described above can be used in parallel, thereby completing the task in a computationally faster manner. Both schemes should enable a good distribution of solutions to be obtained. Moreover, in the case of more than three objectives and when only a small number of Pareto-optimal

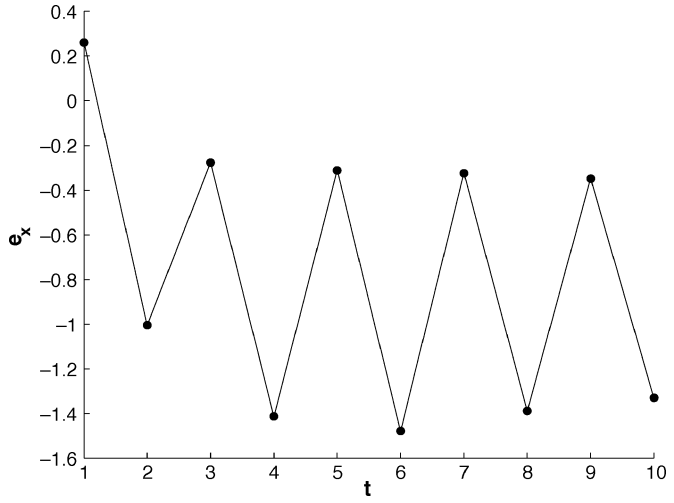


Fig. 32. FDA4: Log of POS convergence measure ($\log(e_x(t))$, (27)) versus time.

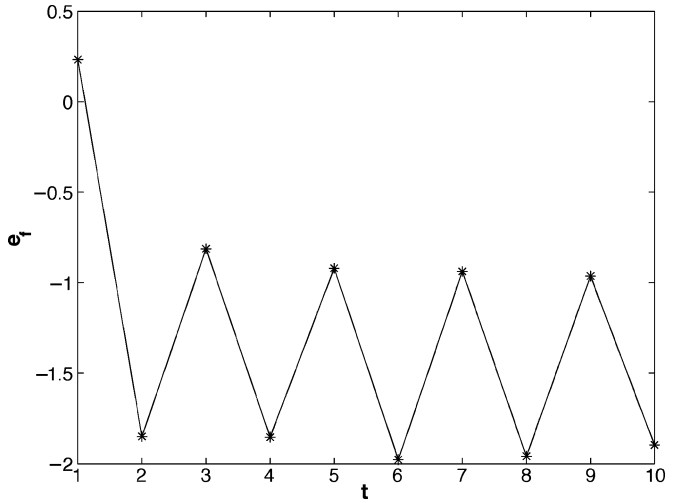


Fig. 33. FDA4: Log of POF convergence measure ($\log(e_f(t))$, (27)), versus time.

solutions are required, a design-of-experiments method may be used in the objective space for a appropriate choice of center points P^k. Finally, the population $pst(t+1)$ for the next iteration is formed using the set of current solutions S^{k^*}.

In general, any state-of-the-art EMO algorithm can also be used for solving mode B problems, and no particular modification to the operators may be necessary. Since these EMO methods are well capable of finding a diverse set of solutions on the current POF, the diversity present in the population may be adequate for these algorithms to locate the new POF quickly. However, this hypothesis needs testing, and we leave this important task for another study, but, the advantage of the proposed direction-based search strategy is that it requires smaller number of overall function evaluations (C_O). For the number of search directions (n_s) greater than two, this estimate is $C_O = n_s \times (ni_{(1+1)ES} + ni_L)$, where $n_{(1+1)ES}$ is the number of iterations used in each $(1+1)ES$ global search method, and ni_L is the number of iterations used in each local search method.

Because of the most immediate application of dynamic multiobjective optimization is dynamic control of time-varying systems, some careful time-analysis is to be considered, and the algorithm runtime is to be compared with the frequency of time changes. Once agai,n the analysis is strongly problem-dependent.

B. Convergence Measure

To measure the performance of a dynamic EMO, both the convergence and diversity of obtained solutions must be considered. Here, we simply use the following two terms for measuring convergence in decision and objective spaces:

$$e_x(t) = \frac{1}{np}\sum_{j=1}^{np} \min_{i=1:nh} \left\| \frac{\mathcal{F}_{P,i}(t) - \mathbf{F}_j^{sol}(t)}{\mathbf{R}(t) - \mathbf{U}(t)} \right\| \tag{27}$$

$$e_f(t) = \frac{1}{np}\sum_{j=1}^{np} \min_{i=1:nh} \left\| \mathcal{S}_{P,i}(t) - \mathbf{X}_j^{sol}(t) \right\| \tag{28}$$

where nh is the number of sampling points used to represent the known $\mathcal{F}_P$ and $\mathcal{S}_P$, np is the number of obtained nondominated solutions, and $\mathbf{F}^{\mathrm{sol}}(t)$ and $\mathbf{X}^{\mathrm{sol}}(t)$ are computed solutions in decision variable space and objective space, respectively. We point out that, although, from the mathematical point of view a design domain measure could be sufficient, from the design point of view a design domain measure may give useful information related to the design tolerance (it is useless to go with the optimization beyond the design variable tolerance). The operator $\|\cdot\|$ is the Euclidean distance. Although the above two metrics evaluate a convergence measure of the obtained solutions in both $\mathcal{S}_P$ and $\mathcal{F}_P$, we also emphasize using a diversity measure that is at least in the $\mathcal{F}_P$ space for evaluating the performance of a dynamic EMO algorithm. For brevity, we do not use a diversity measure here; instead, we show a visual presentation of the solutions through plots in both spaces in the following section.

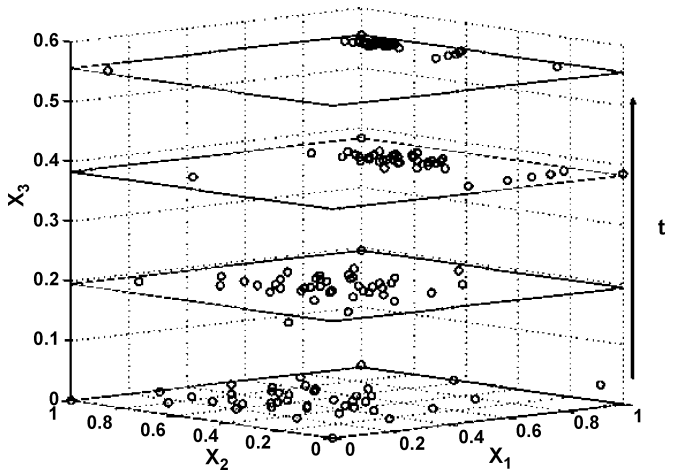


Fig. 34. Exact $\mathcal{S}_P$ (wire-frame rectangle) and approximated ($\circ$) with a search direction-based method on FDA5 (ten solutions).

V. Simulation Results on FDAs

We now present results of one simulation for each test problem FDA1 to FDA5. For each of them, we show the time-dependent solution set in both the decision variable space

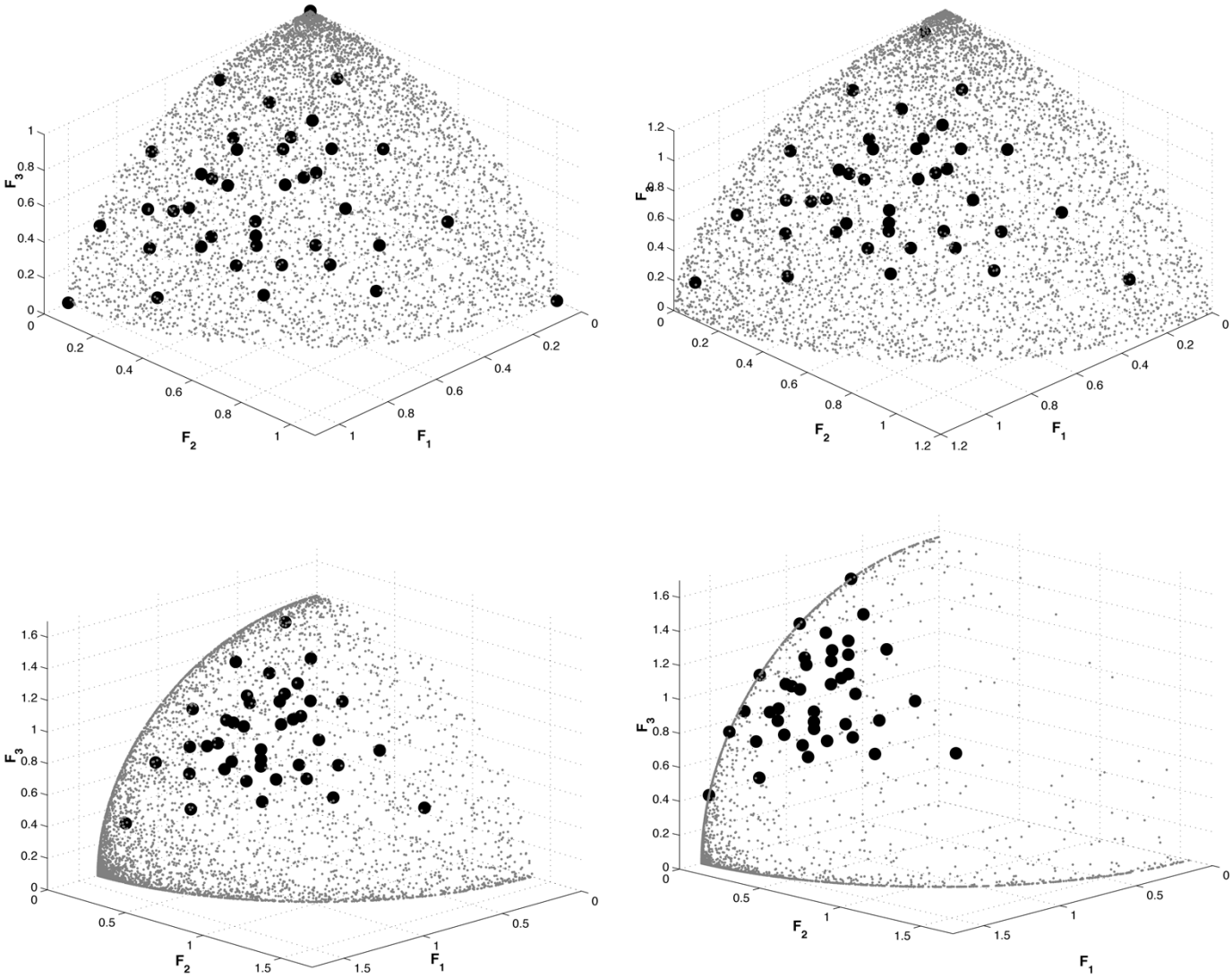


Fig. 35. Exact $\mathcal{F}_P$ (dots) and approximated ($\circ$) with the proposed search direction-based method on FDA5 (forty solutions).

and in the objective space. For a better visualization of a changed solution-set with time, we show the obtained solution-set translated on the space where it is changed and mark this translation by t. If the solution-set (either $\mathcal{S}_P$ or $\mathcal{F}_P$) does not change its location but only gets redistributed (as in Types I and III problems), we mark the changes with a δ_{vis}. For clarity, we do not show a complete n-dimensional decision variable space, instead, we show the variations by using at most three important variables. In both spaces, the known POF and set are shown with dots or lines. Obtained time-dependent solutions are shown with circles. Figs. 18 and 19 show the corresponding $\mathcal{S}_P$ and $\mathcal{F}_P$ variations for FDA1. This problem is a Type I problem, thus, we observe a time change in the $\mathcal{S}_P$ alone. Note that for better visualization purposes, $f_1 + t/2$ and $f_2 + t/2$ are shown on the x and y axes, respectively. The method encounters some difficulties when the $\mathcal{S}_P$ gets close to the bounding box region. To estimate the converging ability of the proposed algorithm, we also show the time-dependent convergence measures $e_x(t)$ and $e_f(t)$ in both decision variable space and objective space in Figs. 20 and 21, respectively. The convergence measure plots show that after each change, the proposed method is able to reduce the convergence metric down to a small value. When there is another time change, the convergence gets affected, but the algorithm is able to make another good convergence to the new optimum set. This happens as many times as there are changes in the problem. The time step of each oscillation in the above plots is, thus, not related to the iteration of each search but to each sudden time changes in the problem.

Similar performance of the proposed algorithm is observed for FDA2 as well and can be seen through Figs. 22–25. This problem is a Type III problem, and we observe a time change in the $\mathcal{F}_P$ alone.

Figs. 26 and 27 show the changes in $\mathcal{S}_P$ and $\mathcal{F}_P$ for FDA3. This is a Type II problem exhibiting variations in both spaces. The vulnerability of the proposed algorithm to such changes is evident from this problem. At the fourth time step, the distribution of solutions is poor. Fig. 26 shows the particularly poor convergence of solutions in the decision variable space. Figs. 28 and 29 show the corresponding convergence plots. Although the proposed method is able to reduce the convergence measure after each time change, $\mathcal{S}_P$ and $\mathcal{F}_P$ plots show that the reduction is not adequate.

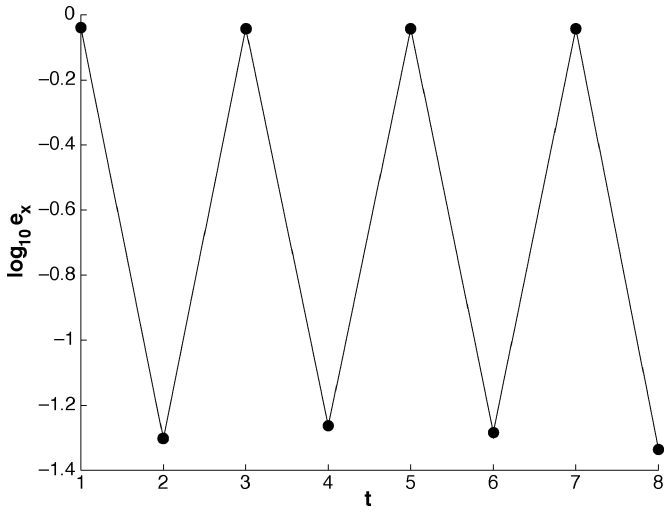


Fig. 36. FDA5: Log of POS convergence measure ($\log(e_x(t))$, (27)) versus time.

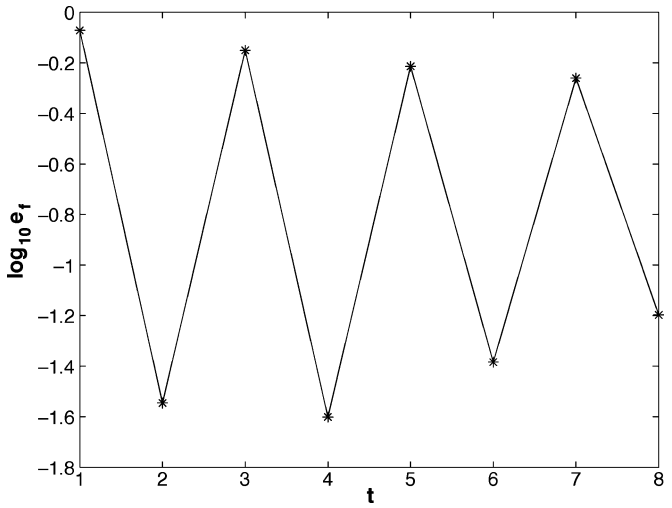


Fig. 37. FDA5: Log of POF convergence measure ($\log(e_f(t))$, (27)), versus time.

On FDA4, we obtain reasonably good convergence after each time change, as depicted in Figs. 30–33. However, with a change in density of solutions on the POF in FDA5 problem, we observe some difficulties in maintaining diversity by the proposed method (Figs. 34–37).

Test results for the controller design example are not shown because it is not a test case but an example toward more real-life problems. Moreover, results for dynamic multiobjective TSP have not been given because the algorithm shown in the paper is for real-valued variables, and the inclusion of such results would require the inclusion of an additional algorithm, which seems to be out of the scope of the paper.

VI. Conclusion

In this paper, the existing static multiobjective test problems are extended with time-dependent parameters to make them suitable test problems for dynamic multiobjective optimization studies. The test problems offer different complexities, such as nonconvexity, disconnectedness, deceptiveness and others, besides being time-dependent. Although continuous test problems, discrete test problems, and a real-world dynamic controller optimization problem are discussed, a test suite of five dynamic test problems (FDA1 to FDA5) are suggested for investigation. The task of a dynamic EMO algorithm in solving such problems is to not only handle the known search space difficulties but also to handle a number of such difficulties one after another in a time-varying manner. A successful dynamic EMO algorithm is therefore, required to be quick in its converging ability and should be able to produce any kind of diversity needed to get out from a converged set of solutions to converge to a new set.

As in the single-objective dynamic optimization, the case of random sudden changes and slow continuous changes are considered as extreme cases where different methods are expected to behave differently. In this paper, we have concentrated only on the first case of random sudden changes in a problem. However, any other scenario can also be implemented in the test problems through the suggested construction procedure. To demonstrate the difficulties offered by the suggested test problems, we have also suggested a directional search-based method and applied it on all five test problems. The initial results suggest the need for more efficient dynamic EMO algorithms, probably using state-of-the-art EMO methods such as NSGA-II, SPEA2, or PESA, etc., and an immediate need to perform a more rigorous simulation study. The present study has only dealt with tracking a set of dynamic Pareto-optimal solutions as and when the problem changes, but an important matter of choosing and tracking one compromised Pareto-optimal solution remains as even a harder task. This is because such a compromised solution will certainly depend on the shape of the POF at every time instant, and although the whole POF may change gradually the compromised solution may change its relative location on the front quite dramatically. Viewing all these possibilities for further meaningful research in dynamic evolutionary multiobjective optimization, this paper has only scratched the surface by providing some useful test problems and providing a baseline algorithm for the purpose. However, the study has certainly provided a platform to launch more detail studies. Only after more successful dynamic EMO algorithms are developed and tested, will they be ready to be applied to real-world problems which are multiobjective in nature and which are filled with parameters of changing nature.

References

[1] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, U.K.: Wiley, 2001.

[2] J. Branke, K. Deb, H. Dierolf, and M. Oswald, "Finding knees in multiobjective optimization," in *Proc. Int. Conf. Parallel Problem Solving from Nature*, KanGAL Rep. No. 2004010, 2004, [Online]. Available: http://www.iitk.ac.in/kangal/pub.htm.

[3] ——, "Unveiling innovative design principles by means of multiple conflicting objectives," *Eng. Opt.*, vol. 35, no. 5, pp. 445–470, 2003.

[4] ——, "Multi-objective genetic algorithms: problem difficulties and construction of test problems," *Evol. Comput. J.*, vol. 7, no. 3, pp. 205–230, 1999.

[5] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *Proc. Congress Evolutionary Computation*, 2002, pp. 825–830.

[6] P. J. Fleming and R. C. Purshouse, "Evolutionary algorithms in control systems engineering: a survey," *Control Eng. Practice*, vol. 10, pp. 1223–1241, 2002.

[7] C. M. M. de Fonseca, "Multiobjective genetic algorithms with applications to control engineering problems," Ph.D. dissertation, Dept. Autom. Control Syst. Eng., Univ. Sheffield, Sheffield, U.K., Sept. 1995.
[8] J. M. Anderson, T. M. Sayers, and M. G. H. Bell, "Optimization of a fuzzy logic traffic signal controller by a multiobjective genetic algorithm," in *Proc. 9th Int. Conf. Road Transport Information Control*, London, U.K., Apr. 1998, pp. 186–190.
[9] A. L. Blumel, E. J. Hughes, and B. A. White, "Fuzzy autopilot design using a multiobjective evolutionary algorithm," in *Proc. Congr. Evolutionary Computation*, vol. 1, July 2000, pp. 54–61.
[10] K. Trojanowsky and Z. Michalewicz, "Evolutionary algorithms for nonstationary environments," in *Proc. Intelligent Information Systems VIII Workshop Ustron Poland*, June 1999, pp. 229–240.
[11] C. Ronnewinkel, C. O. Wilke, and T. Martinetz, "Genetic algorithms in time-dependent environments," in *Theoretical Aspects of Evolutionary Computing*, L. Kallel, B. Naudts, and A. Rogers, Eds. Berlin, Germany: Springer-Verlag, 2000, pp. 263–288.
[12] F. Vavak, K. A. Jukes, and T. C. Fogarty, "Performance of a genetic algorithm with variable local search range relative to frequency of the environmental changes," in *Proc. 3rd Annu. Conf. Genetic Programming*, 1998, pp. 602–608.
[13] A. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 3, p. 124, 1999.
[14] J. Branke, "Evolutionary approaches to dynamic optimization problems—a survey," in *Evolutionary Algorithms for Dynamic Optimization Problems*, J. Branke and T. Baeck, Eds., 1999, vol. 13, pp. 134–137.
[15] J. J. Grefenstette, "Evolvability in dynamic fitness landscapes: a genetic algorithm approach," in *Proc. Congress Evolutionary Computation*, Washington, DC, 1999, pp. 2031–2038.
[16] ——, "Genetic algorithms for changing environments," in *Proc. 2nd Int. Conf. Parallel Problem Solving From Nature*, Brussels, Belgium, 1992.
[17] T. Back, "Self-adaptation in genetic algorithms," in *Proc. 1st Eur. Conf. Artificial Life*, Paris, France, 1991, pp. 263–271.
[18] C. G. Langton, *Artificial Life: An Overview*. Cambridge, MA: MIT Press, 1995.
[19] C. Adami, *Introduction to Artificial Life*. Berlin, Germany: Springer-Verlag, 1998.
[20] L. M. Rocha, "Evolutionary Systems and Artificial Life," in *Ssie 580b*. Los Alamos, NM: Los Alamos Nat. Lab., 1997.
[21] M. Mitchell and S. Forrest, "Genetic algorithms and artificial life," Santa Fe Institute Working Paper 93-11-072 (to appear in Artificial Life).
[22] H. W. Thimbleby, I. H. Witten, and D. J. Pullinger, "Concepts of cooperation in artificial life," *IEEE Trans. Syst., Man, Cybernetics*, vol. 25, pp. 1166–1171, July 1995.
[23] M. Kubat and G. Widmer, "Adapting to drift in continuous domains," in *Lecture Notes in Computer Science*, vol. 912, 1995, pp. 307–321.
[24] A. Gaspar *et al.*, "From GAs to artificial immune systems: Improving adaptation in time dependent optimization," in *Proc. Congr. Evolutionary Computation*, Washington, DC, 1999, pp. 1859–1866.
[25] P. Amato, M. Farina, G. Palma, and D. Porto, "An alife-inspired evolutionary algorithm for adaptive control of time-varying systems," in *Proc. EUROGEN Conf.*, Athens, Greece, Sept. 2001, pp. 222–227.
[26] Z. Bingul, A. Sekmen, and S. Zein-Sabatto, "Adaptive genetic algorithms applied to dynamic multi-objective problems," in *Proc. Artificial Neural Networks Engineering Conf.*, C. H. Dagli, A. L. Buczak, J. Ghosh, M. Embrechts, O. Ersoy, and S. Kercel, Eds., New York, 2000, pp. 273–278.
[27] K. Yamasaki, "Dynamic Pareto optimum GA against the changing environments," in *Proc. Genetic Evolutionary Computation Conf. Workshop Program*, San Francisco, CA, July 2001, pp. 47–50.
[28] K. Miettinen, *Nonlinear Multiobjective Optimization*. Dordrecht, The Nederlands: Kluwer, 1999.
[29] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evol. Comput. J.*, vol. 8, no. 2, pp. 125–148, 2000.
[30] D. Van Veldhuizen, "Multiobjective Evolutionary Algorithms: Classifications, Analyzes, and New Innovations," Ph.D. dissertation, Air Force Inst. Technol., Dayton, OH, 1999. Tech. Rep. AFIT/DS/ENG/99-01.
[31] E. Zitzler, "Evolutionary Agorithms for Multiobjective Optimization: Methods and Applications," Ph.D. dissertation, Swiss Federal Instit. Technol. (ETH), Zürich, Switzerland, 1999.
[32] "Interdisciplinary Center for Scientific Computing of the Ruprecht-Karls-University of Heidelberg," Library of TSP Problems, Heidelberg, Germany, 2003.
[33] M. Annunziato M, I. Bertini, A. Pannicelli, and S. Pizzuti, "Evolutionary control and optimization: An industrial application for combustion processes," in *Proc. EUROGEN*, Athens, Greece, Sept. 2001, pp. 367–372.
[34] D. Dasgupta, "Optimization in time-varying environments using structured genetic algorithms," Univ. Strathclyde, Dept. Comput. Sci., Glasgow, U.K., Tech. Rep. IKBS-17-93.
[35] M. Farina, "A minimal cost hybrid strategy for pareto optimal front approximation," *Evol. Opt.*, vol. 3, no. 1, pp. 41–52, 2001.

Marco Farina was born in Pavia, Italy, in 1973. He received the Laurea and Ph.D. degrees in electrical engineering from the University of Pavia, in 1997 and 2001, respectively.

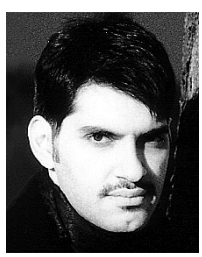
From 1999 to 2000, he was a Visiting Researcher at the University of Southampton, Southampton, U.K., and in 2001, he joined STMicroelectronics. Agrate, Italy, where he is now Project Leader in the SST Corporate R&D. He is the author of 15 papers on peer-reviewed international journals and more than 20 international conference papers. His professional activity and scientific interests are in the field of optimal industrial design, multiobjective optimization, natural computing, and computational electromagnetics.

Dr. Farina is a Reviewer for the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and *Soft Computing Journals*. He is a Member of the Program Committees of the GECCO'03, CEC'04, EvoSTOC'04, and EMO'05 International Conferences. He is Member of the Scientific and Organizing Committee of the European School on Complex System Management, University of Pavia. He gave invited seminars in several international academic institutions such as the University of Berkeley and the University of Jyvaskyla, Finland.

Kalyanmoy Deb received the B.Tech. degree in mechanical engineering from the Indian Institute of Technology, Kharagpur, in 1985, and the M.S. and Ph.D. degrees in engineering mechanics from the University of Alabama, Tuscaloosa, in 1989 and 1991, respectively.

He is a Professor of Mechanical Engineering at the Indian Institute of Technology Kanpur. His current research interests are in the field of evolutionary computation, particularly in the areas of multicriterion and real-parameter evolutionary algorithms. His recent book *Multi-Objective Optimization Using Evolutionary Algorithms* (Chichester, U.K.: Wiley, 2001) is the first comprehensive monograph on the subject. He has also published *Optimization for Engineering Design* (New Delhi, India: Prentice-Hall, 1995). Besides a number of book chapters, he has published over 140 research papers in international journals and conferences.

Dr. Deb is an Associate Editor of the IEEE TRANSACTION ON EVOLUTIONARY COMPUTATION and the *Evolutionary Computation Journal*. He is on the Editorial Board of the *Engineering Optimization Journal* and *Genetic Programming and Evolvable Machines Journal*. He is a Fellow of the International Society of Genetic and Evolutionary Computation (ISGEC).



Paolo Amato was born in Desio, Italy, in 1973. He received the Laurea degree in computer science from the University of Milan, Milan, Italy, in 1997.

Since 1998, he has been with SST Corporate R&D, STMicroelectronics, Agrate, Italy. He is currently the R&D Manager of the Methodology for Complexity Team, a team aimed at developing methods, algorithms, and software tools for complex system management. His main research interest are in the areas of many-valued logic, optimal industrial design, multiobjective optimization, natural computing, and cryptography.

Mr. Amato is a Reviewer for the *Soft Computing Journal* and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS. He is a member of program committees of several international conferences. He is Member of the International Advisory Board of the European School for Advanced Studies in Methods for Management of Complex Systems, University of Pavia, Pavia, Italy.