Int. J. Appl. Math. Comput. Sci., 2013, Vol. 23, No. 1, 35–45
DOI: 10.2478/amcs-2013-0004

A MULTIVARIABLE MULTIOBJECTIVE PREDICTIVE CONTROLLER

FATEN BEN AICHA, FAOUZI BOUANI, MEKKI KSOURI

LR-ACCS-ENIT
National Engineering School of Tunis, BP 37, Le Belvedere 1002 Tunis, Tunisia
e-mail: {faten.benaicha,bouani.faouzi}@yahoo.fr,
mekki.ksouri@enit.rnu.tn

Predictive control of MIMO processes is a challenging problem which requires the specification of a large number of tuning parameters (the prediction horizon, the control horizon and the cost weighting factor). In this context, the present paper compares two strategies to design a supervisor of the Multivariable Generalized Predictive Controller (MGPC), based on multiobjective optimization. Thus, the purpose of this work is the automatic adjustment of the MGPC synthesis by simultaneously minimizing a set of closed loop performances (the overshoot and the settling time for each output of the MIMO system). First, we adopt the Weighted Sum Method (WSM), which is an aggregative method combined with a Genetic Algorithm (GA) used to minimize a single criterion generated by the WSM. Second, we use the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) as a Pareto method and we compare the results of both the methods. The performance of the two strategies in the adjustment of multivariable predictive control is illustrated by a simulation example. The simulation results confirm that a multiobjective, Pareto-based GA search yields a better performance than a single objective GA.

Keywords: closed loop performance, coupled multivariable system, generalized predictive control, multiobjective optimization, weighted sum method, NSGA-II.

1. Introduction

Industrial processes are essentially coupled multivariable systems. The Generalized Predictive Control (GPC) strategy offers an effective way to deal with problems in multivariable control systems by including the process model in the computation of control actions. Without any doubt, the ability of GPCs regarding this kind of processes attracted notable attention in industry over the past few years (Qin and Badgwell, 2003).

In order to control a multivariable system, some works used multivariable predictive controllers in a completely decentralized fashion (Zenghui *et al*., 2006; Al-Gherwi *et al*., 2010). To realize that, it is necessary to design decoupling compensators to eliminate the interactions, and then the MIMO system is treated as a set of independent SISO systems and individual GPC controllers for the different subsystems are operated. In such cases, satisfactory results are obtained if the interactions are negligible. But in many cases, industrial systems are strongly coupled and more than one input variable is coupled with the outputs. Consequently, closed-loop performances may be significantly decreased since some or all interactions are ignored (Al-Gherwi *et al*., 2010). In this case, the system must be considered truly multivariable and some type of multivariable control has to be applied to achieve satisfactory performances.

It is evident that distributed GPC strategies have a simpler structure but their performance is expected to be generally poorer as compared with centralized GPC strategies which will be applied in this work. The design of multivariable predictive controllers requires the specification of synthesis parameters, namely, prediction horizons, control horizons and cost weighting factors. But there are not exact rules giving the values of the required parameters. In this context, some works (Bemporada and Munoz de la Penab, 2009; Muldera *et al*., 2009; Królikowski and Jerzy, 2001) were interested in tuning of the MPC parameters using multiobjective optimization.

A new Model Predictive Control (MPC) scheme based on multiparametric multiobjective linear programming was presented (Bemporada and Munoz de la Penab, 2009). In the work of Muldera *et al*. (2009), a method for the synthesis of a simultaneous linear and anti-windup controller using multiobjective convex optimization is described and tested using a benchmark problem. Also, some works deal with the automatic tuning of MPC, e.g., Ben Abdennour *et al.* (1998) present an on-line adjustment of GPC synthesis parameters using fuzzy logic. Therefore the objective of this work is to adjust the synthesis parameters of multivariable predictive controllers to make a compromise between closed loop performances. To realize this the optimization problem treated in this work is a multiobjective one and the performance criteria of this problem correspond to the closed loop performances.

In this paper, we make use of Multi-Objective Optimization (MOO) based on the minimization of all performance criteria simultaneously. MOO leads to a set of optimal solutions, i.e., Pareto optimal solutions or non-dominated solutions (Colette and Siarry, 2002). In this context, many works (e.g., Bemporada and Munoz de la Penab, 2009; Muldera *et al.*, 2009; Popov *et al.*, 2005; Yang and Pedersen, 2006; Behroozsarand and Shaffei, 2010) were focused on the synthesis of controllers based on multiobjective optimization, which has received growing interest.

In this work, we suggest two strategies to achieve our purpose. The first one is a combination of an aggregative method, namely, the weighted sum method, allowing the transformation of the criteria in only one criterion, and a GA whose role is to minimize the generated single criterion by the WSM. The second strategy uses NSGA-II as a multiobjective Pareto-based GA. NSGA-II is a modified version of the popular NSGA to rectify all the above issues of the latter. In fact, NSGA-II has a better sorting algorithm ($O(MN^2)$ instead of $O(MN^3)$, where M is the number of objectives and N is the population size), incorporates elitism and no sharing parameter needs to be chosen *a priori* (Deb, 2002; Srinivas and Deb, 1995). The performance criteria to be simultaneously minimized are the overshoot and the settling time for each MIMO system output.

This paper is organized as follows. The problem is formulated in Section 2, where the key elements needed to formulate a centralized MIMO predictive control law are given. Both strategies allowing the design of the multivariable multiobjective predictive controller are described in Section 3. The obtained simulation results and discussion are presented in Section 4. Conclusions and perspectives are given in the last section.

2. Centralized multivariable generalized predictive control

This section is an extension of the GPC proposed by Clarke *et al.* (1987) to multivariable systems.

2.1. Multivariable system representation.

We consider a multivariable linear system with n inputs $u_l(k), l = 1,\dots,n$ and m outputs $y_j(k), k = 1,\dots,m$ given by

$$\mathbf{y}(k) = \mathbf{H}(z^{-1})\mathbf{u}(k), \tag{1}$$

where $\mathbf{u}(k) = [u_1(k), u_2(k), \dots, u_n(k)]^T \in \mathbb{R}^n$ is the control vector, $\mathbf{y}(k) = [y_1(k), y_2(k), \dots, y_m(k)]^T \in \mathbb{R}^m$ is the output vector and

$$\mathbf{H}(z^{-1}) = \begin{pmatrix} H_{11}(z^{-1}) & \cdots & H_{1n}(z^{-1}) \\ \vdots & \ddots & \vdots \\ H_{m1}(z^{-1}) & \cdots & H_{mn}(z^{-1}) \end{pmatrix}$$

is the $m \times n$ transfer function matrix.

In this section, the basic principles of the multivariable GPC control design are discussed, based on ARMA dynamics models. The system given by (1) can be written in the matrix form (Camacho and Bordons, 1995) as Eqn. (2) with

$$\begin{aligned} A_{jj}(z^{-1}) &= 1 + a_{jj}(1)z^{-1} + a_{jj}(2)z^{-2} + \dots \\ &\quad + a_{jj}(n_{A_{jj}})z^{-n_{A_{jj}}}, \quad j = 1,2,\dots,m, \end{aligned} \tag{3}$$

$$\begin{aligned} B_{jl}(z^{-1}) &= b_{jl}(0) + b_{jl}(1)z^{-1} + \dots \\ &\quad + b_{jl}(n_{B_{jl}})z^{-n_{B_{jl}}}, \quad l = 1,2,\dots,n. \end{aligned} \tag{4}$$

2.2. Objective function.

MGPC is based on the minimization of a quadratic criterion, with a first term corresponding to the difference between the predicted output sequence and the future set points sequence and a second term corresponding to the future weighted control increments (Clarke *et al.*, 1987; Richalet *et al.*, 2005; Mohtadi *et al.*, 1987; Kinnaert, 1989) given by the following expression:

$$\begin{aligned} J_{GPC} &= \sum_{j=1}^{m}\sum_{i=1}^{H_{p_j}} \left(r_j(k+i) - \hat{y}_j(k+i/k)\right)^2 \\ &\quad + \sum_{l=1}^{n} \rho_l \sum_{i=0}^{H_{c_l}-1} \left(\Delta u_l(k+i)\right)^2, \end{aligned} \tag{5}$$

where H_{p_j} is the prediction horizon for the scalar output y_j , H_{c_l} is the control horizon for the input u_l and ρ_l is the weighting factor of control increments or, equivalently,

$$J_{GPC} = (\hat{r} - \hat{y})^T(\hat{r} - \hat{y}) + \Delta u^T \mathbf{\Lambda} \Delta u, \tag{6}$$

where

$$\hat{r} = \left[\hat{r}_1^T \ \hat{r}_2^T \ \dots \ \hat{r}_m^T\right]^T \in \mathbb{R}^{Hp}$$

is an augmented set-point vector composed of

$$H_p = \sum_{j=1}^{m} H_{p_j}$$

$$\begin{bmatrix} A_{11}(z^{-1}) & 0 & \dots & 0 \\ 0 & A_{22}(z^{-1}) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A_{mm}(z^{-1}) \end{bmatrix} \begin{bmatrix} y_1(k) \\ y_2(k) \\ \vdots \\ y_m(k) \end{bmatrix} = \begin{bmatrix} B_{11}(z^{-1}) & B_{12}(z^{-1}) & \dots & B_{1n}(z^{-1}) \\ B_{21}(z^{-1}) & B_{22}(z^{-1}) & \dots & B_{2n}(z^{-1}) \\ \vdots & \vdots & \ddots & \vdots \\ B_{m1}(z^{-1}) & B_{m2}(z^{-1}) & \dots & B_{mn}(z^{-1}) \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \\ \vdots \\ u_n(k-1) \end{bmatrix}, \tag{2}$$

rows obtained by stacking the sub-vectors

$$\hat{r}_j = \left[r_j(k+1)\, r_j(k+2)\, \dots\, r_j(k+H_{p_j})\right]^T \in \mathbb{R}^{H_{p_j}}, \quad j = 1, 2, \dots, m,$$

which specify the desired future set-point trajectories for each output

$$\hat{y} = \left[\hat{y}_1^T\, \hat{y}_2^T\, \dots\, \hat{y}_m^T\right]^T \in \mathbb{R}^{Hp}$$

is the augmented vector of predicted future outputs which contains the predicted future values of the m outputs, and is composed of the sub-vectors

$$\hat{y}_j = \left[\hat{y}_j(k+1/k)\ \dots\ \hat{y}_j(k+H_{p_j}/k)\right]^T \in \mathbb{R}^{H_{p_j}}, \quad j = 1, 2, \dots, m.$$

The augmented vector

$$\Delta\hat{u} = \left[\Delta\hat{u}_1^T\, \Delta\hat{u}_2^T\, \dots\, \Delta\hat{u}_n^T\right]^T \in \mathbb{R}^{H_c},$$

composed of

$$H_c = \sum_{l=1}^{n} H_{cl}$$

rows, contains all the input-increment values required to produce, and it contains the sub-vectors

$$\Delta\hat{u}_l = [\Delta u_l(k)\ \dots\ \Delta u_l(k+H_{c_l}-1)]^T \in \mathbb{R}^{H_{c_l}}, \quad l = 1, 2, \dots, n.$$

The input-weighting matrix

$$\mathbf{\Lambda} \in \mathbb{R}^{H_c x H_c}$$

is given by

$$\mathbf{\Lambda} = \operatorname{diag}\{\rho_1\, I_{Hc_1},\ \rho_2 I_{Hc_2},\ \dots,\ \rho_n I_{Hc_n}\}$$

with scalars $\rho_l \geq 0,\ l = 1, 2, \dots, n.$

In this work, only the weighting coefficients for future control increments and their influence on the closed loop performance are taken into account. We can also introduce weights for predicted control errors in the performance function to illustrate their influence on overshoots and settling times.

2.3. Design of a MIMO predictor via Diophantine equations. We consider m MISO systems resulting from (1),

$$A_{jj}(z^{-1})y_j(k) = \sum_{l=1}^{n} B_{jl}u_l(k-1),, \tag{7}$$

$j = 1, \dots, m.$ A predictor form can be derived from (7) by solving the set of Diophantine equations

$$F_{j,i}(z^{-1})\Delta A_{jj}(z^{-1}) + z^{-i}G_{j,i}(z^{-1}) = 1 \tag{8}$$

and

$$F_{j,i}(z^{-1})B_{jl}(z^{-1}) = Q_{jl,i}(z^{-1}) + z^{-i}R_{jl,i}(z^{-1}). \tag{9}$$

The predictor is written in this vector-matrix form as

$$\hat{y} = \hat{y}^0 + Q\Delta\hat{u}. \tag{10}$$

The dynamic matrix appearing in (10) is of the form

$$\mathbf{Q} = \begin{bmatrix} & \mathbf{Q}_{12} & \dots & \mathbf{Q}_{1n} \\ & \mathbf{Q}_{22} & \dots & \mathbf{Q}_{2n} \\ & \vdots & \ddots & \vdots \\ \mathbf{Q}_{m1} & \mathbf{Q}_{m2} & \dots & \mathbf{Q}_{mn} \end{bmatrix} \in \mathbb{R}^{H_p \times H_c}, \tag{11}$$

where

$$\mathbf{Q}_{jl} = \begin{bmatrix} q_{jl}(0) & 0 & \dots & 0 \\ q_{jl}(1) & q_{jl}(0) & \ddots & \vdots \\ q_{jl}(2) & q_{jl}(1) & \ddots & 0 \\ \vdots & \vdots & \ddots & q_{jl}(0) \\ \vdots & \vdots & \ddots & \vdots \\ q_{jl}(H_{p_j}-1) & q_{jl}(H_{p_j}-2) & \dots & q_{jl}(H_{p_j}-H_{c_l}) \end{bmatrix} \in \mathbb{R}^{H_{p_j} \times H_{cl}} \tag{12}$$

The vector $\hat{\mathbf{y}}^0$ is given by

$$\hat{y}^0 = \mathbf{G}(z^{-1})y(k) + z^{-1}R(z^{-1})\Delta u(k), \tag{13}$$

where

$$\mathbf{G}_j(z^{-1}) = \begin{bmatrix} G_{j,1}(z^{-1}) \\ G_{j,2}(z^{-1}) \\ \vdots \\ G_{j,H_{p_j}}(z^{-1}) \end{bmatrix} \in \mathbb{R}^{H_{p_j}\times 1} \tag{14}$$

and

$$\mathbf{R}_j = \begin{bmatrix} R_{j1,1}(z^{-1}) & R_{j2,1}(z^{-1}) & \dots & R_{jn,1}(z^{-1}) \\ R_{j1,2}(z^{-1}) & R_{j2,2}(z^{-1}) & \dots & R_{jn,1}(z^{-1}) \\ \vdots & \vdots & \ddots & \vdots \\ R_{j1,H_{p_j}}(z^{-1}) & R_{j2,H_{p_j}}(z^{-1}) & \dots & R_{jn,H_{p_j}}(z^{-1}) \end{bmatrix} \in \mathbb{R}^{H_{p_j}\times n}. \tag{15}$$

2.4. Optimal control solution. The solution of the problem described by (6) is given by

$$\Delta\hat{u}_{\text{opt}} = \mathbf{Z}(\hat{r} - \hat{y}^0), \tag{16}$$

where

$$\mathbf{Z} = (\mathbf{Q}^T\mathbf{Q} + \mathbf{\Lambda})^{-1}\mathbf{Q}^T \in \mathbb{R}^{H_c\times H_p} \tag{17}$$

Using the receding horizon principle, only the elements of the first row of $\Delta\hat{u}_{\text{opt}}$ which corresponds to the current instant k are used. These elements are collected in the vector $\Delta u(k) = [\Delta u_1(k)\ \Delta u_2(k)\ \dots \Delta u_n(k)] \in \mathbb{R}^n$. Then we have the vector matrix MIMO predictive law

$$\Delta u(k) = \mathbf{K}(\hat{r} - \hat{y}^0), \tag{18}$$

where the gain matrix $\mathbf{K} \in \mathbb{R}^{n\times H_p}$ contains the rows of $\mathbf{Z}$ that are labelled with 1, 1+H_{c1} , 1+H_{c1}+H_{c2} and so on.

3. Design of the multiobjective tuning of GPC parameters

MOO can be defined as the problem of finding a vector of parameters $X = [x_1, \dots, x_e]^T$, which optimizes a vector of objective functions $(J_1(X), \dots, J_f(X))$ (Berro, 2001; Talbi, 2001; Coello Coello *et al.*, 2002). In general, the MOO problem can be formulated as follows:

$$\min J = (J_1(X), J_2(X), \dots, J_f(X))$$

subject to

$$\begin{aligned} g_i(X) &\le 0 \qquad \text{for} \quad i = 1, \dots, n_g, \\ h_j(X) &= 0 \qquad \text{for} \quad j = 1, \dots, n_h, \end{aligned} \tag{19}$$

where $J \in F$ (the objective vector field), $X \in \Omega$ is a vector of decision variables, f is the number of objective functions, n_g is the number of inequality constraints and n_h is the number of equality constraints.

In single objective optimization problems, the aim is to determine the global optimal solution, if it exists. Unlike in single objective optimization, in optimization with conflicting objectives there is no single optimal solution and it is often necessary to determine a set of points that all fit a predetermined definition for the optimum. Usually, for multiobjective optimality the following definition given by Pareto is accepted (Gambier, 2008).

Definition 1. A point $X^* \in \Omega \in \mathbb{R}^e$ is *Pareto optimal* with respect to Ω iff there does not exist another point $X \in \Omega$ such that $\mathbf{J}(X) \le \mathbf{J}(X)$ and $J_i(X) \le J_i(X^*)$ for at least one function, i.e., there is no way to improve upon a Pareto optimal point without increasing the value of at least one of the other objective functions.

MOO leads to a set of solutions known as a Pareto set. This set is also called that of non-dominated solutions. When the non-dominated solutions are collectively plotted in the criterion space, they constitute the Pareto front (Gambier, 2008; Veldhuizen and Lamont, 2000). All points of the Pareto front are an equally acceptable solution for the problem.

The determination of the Pareto set is only a first step in the solution of multiobjective problems, which needs secondly the choice of a solution from this optimal set according to the decision-maker's preferences to be able to implement the controller. This choice takes generally three forms: *a priori*, posterior and interactive (Talbi, 2001). In this work, we opt for the posterior decision, which consists in choosing a single optimal solution from among the non dominated solutions after generating all optimal ones. This approach is useful if the cardinality of the Pareto set is reduced (Berro, 2001).

Thus, the work developed in this paper can be divided into two main phases as given in Fig. 1: the first phase named the "evaluation phase" is a phase that allows off-line generation of optimal solutions. In this phase, we make use of the model describing the real system, MGPC and an MOO method. For each set of the MGPC parameters we evaluate the closed loop performances. Then, the synthesis parameters giving the minimum values of objective functions are generated and form the Pareto front. Once all non-dominated solutions are obtained, we proceed to the second phase called the "implementation phase (real time)". In this phase, the

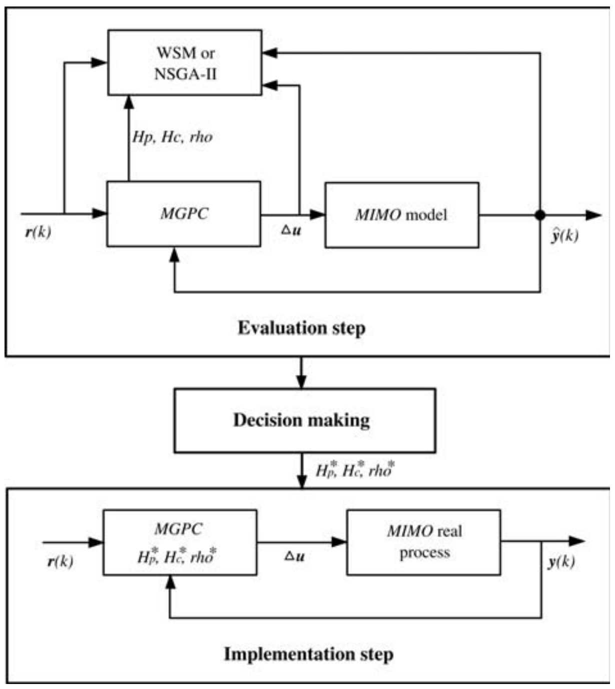

Fig. 1. Principal steps to design the multivariable multiobjective controller.

decision-maker selects a unique solution according to the user preferences in order to implement the multivariable predictive controller and control the real system.

At present, a very huge number of methods to solve MOO problems can be found in the literature (Berro, 2001; Talbi, 2001; Coello Coello *et al.*, 2002; Gambier, 2008; Veldhuizen and Lamont, 2000). In our work, we compare two multiobjective methods to design the multivariable predictive controller. The first one is the Weighted Sum Method (WSM), which is an aggregative one, and the second is NSGA-II, which is a Pareto method.

The quality of a control applied to a multivariable process is generally evaluated by the obtained closed loop performance indices for each output. Among these performance, we choose as objective functions to optimize the following ones:

- The overshoot $D_{j\%}$ corresponding to the j-th output

$$D_{j\%} = 100\frac{y_{j\max} - r_j}{r_j}, \tag{20}$$

  where $y_{j\max}$ is the maximum value of the j-th output and r_j is the j-th set-point value.

- The settling time T_{s_j}: it is the first instant after which the j-th system output does not exceed $\pm 5\%$ of the set-point value.

3.1. Weighted sum method. This method allows the transformation of the objective function vector into a single objective function. It is known for its efficiency and suitability to generate a strongly non-dominated solution that can be used as an initial solution for other techniques (Gambier, 2007). The single criterion is obtained by the sum of the weighted criteria as follows (Gambier, 2007):

$$\sum_{i=1}^{n} w_i J_i(X), \tag{21}$$

where the weights are chosen such that

$$\sum_{i=1}^{n} w_i = 1, \qquad 0 \le w_i \le 1. \tag{22}$$

Thus, applying the WSM, we obtain the following criterion:

$$\begin{aligned} J = {} & w_1 D_{1\%} + w_2 T_{s_1} + w_3 V_{u_1} + \ldots + w_{3m-2} D_{m\%} \\ & + w_{3m-1} T_{s_m} + w_{3m} V_{u_m}, \end{aligned} \tag{23}$$

such that $w_1 + w_2 + \cdots + w_{3m} = 1$ and $0 \le w_i \le 1, i = 1, \ldots, 3m$, with m being the number of outputs of the multivariable system.

Then, to determine the multivariable GPC synthesis parameters, we make use of genetic algorithms to minimize (23) (Mohtadi *et al.*, 1987). Consequently, this proposed method described by Algorithm 1 consists essentially of the combination of the WSM and the GA to determine the GPC synthesis parameters by minimizing the closed loop performance indices stated previously. The GA population is formed by the synthesis parameters (H_{p_j}, H_{c_l}, ρ_l). The initial population is produced by arbitrary values, such as $1 \le H_{p_j} \le 20$, $1 \le H_{c_l} \le 3$ and $0 \le \rho_l \le 10$.

After fixing an interval of iterations, for each individual of the population, we use the multivariable process model and the multivariable generalized predictive control in order to compute, for a given set point, the output sequence. Then, we evaluate the performance indices ($D_{j\%}$ and T_{s_j}) and the fitness given by (23) obtained over this interval as shown in (2). To obtain the new population, we use the roulette wheel as a selection operator. To acquire more information in the new population, the crossover and mutation operators are needed. This procedure will be repeated until a stop criterion (e.g., a maximal number of generation) is reached. Then, we obtain the best individual (optimal values of H_{p_j}, H_{c_l} and ρ_l) that minimizes the performance indexes.

The steps used to compute the best synthesis parameters are given in Algorithm 1. These steps are repeated for each set of weights. In this algorithm, we design by ‘max-gen’ the maximum number of generations and by ‘max-pop’ the maximum number of individuals.

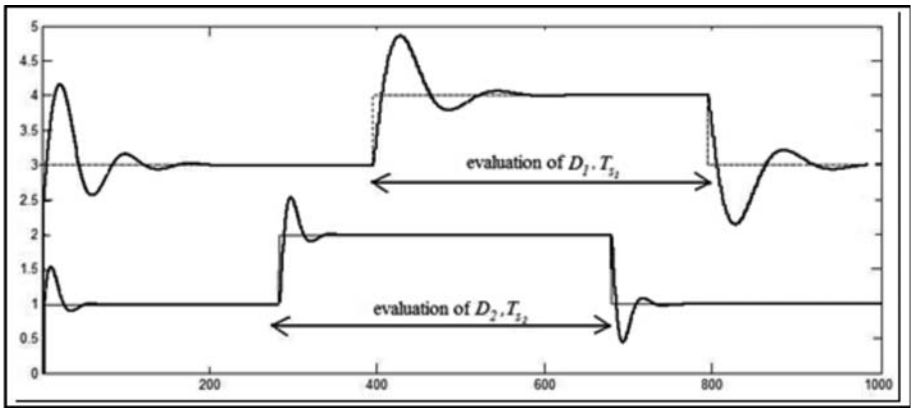

Fig. 2. Evaluation of closed loop performances.

Algorithm 1. First method principal steps to find the best multivariable predictive controller parameters.

```
Require: Form the initial population
 1: for $j = 1$ to max-gen do
 2:   for $i = 1$ to max-pop do
 3:     Take the $i$-th individual of population
 4:     Use the multivariable GPC with the multivariable ARMA process model
 5:     Compute the model outputs and the control signals
 6:     Evaluate the criteria $D_{j\%}$ and $T_{s_j}$
 7:     Evaluate the fitness using (23)
 8:   end for
 9:   Use the GA operators (selection, crossover and mutation) to form the new population.
10: end for
11: Take the best individual $(H_{p_j}, H_{c_l}, \rho_l)$.
```

3.2. Pareto method (NSGA-II). The Non-dominated Sorting Genetic Algorithm (NSGA) proposed by Deb (2002) was one of most successful methods to solve multiobjective optimization problems. The main criticism of the NSGA is the high computational complexity of non-dominated sorting, the lack of elitism and the need for specifying a sharing parameter. Therefore, an improved version of NSGA, called NSGA-II, was suggested by Deb (2002) to address all the above issues. The proposed NSGA-II algorithm is an elitist one, in which the sharing function approach is replaced by the crowding-distance so there is no need for the specification of the sharing parameter. This algorithm requires only computations.

NSGA-II begins as usual with the creation of a random parent population of size N. After initializing the population, the individuals are sorted based on non-domination into each front and each solution is assigned a fitness (or rank) equal to its non-domination level. Individuals in the first front are given a fitness value of 1 and individuals in the second front are assigned a fitness value of 2, and so on. The binary tournament selection, recombination, and mutation operators are used to create an offspring population of size N. After the first generation, the offspring population is combined with the current generation population. Since all the previous and current best individuals are added in the population, elitism is ensured (Deb, 2002).

The remaining members of the population are chosen from subsequent non-dominated fronts in the order of their ranking. The solutions from the set are chosen first, followed by solutions from the set, and so on, until the population size exceeds the current population size. Since it is necessary to choose exactly N solutions, the crowding distance, which is a measure of density of solutions in the neighborhood, is calculated and the solutions of the last front are sorted using the crowding distance in descending order. To select an individual, we use the binary tournament selection, but the selection criterion is based on the rank and the crowding distance while an individual is selected if its rank is less than that the other individual or if the crowding distance is greater than of the other one. To create a new population, the crossover and mutation operators are used. Since the population composed of the current one and current offspring is formed, the obtained population is sorted again based on the non-domination and only the best N individuals are selected based on the rank and the crowding distance on the last front. For the real-coded NSGA-II applied in this work, we need to use the simulated binary crossover (SBX) operator and the polynomial mutation (Deb, 2002).

Simulated binary crossover. The simulated binary crossover operator is designed to simulate the operation of a single-point binary crossover on real variables. Many works suggest that real-coded GAs with the SBX operator are able to perform as well as, or better than, binary-coded GAs with the single-point crossover (Deb and Agrawal, 1995). A spread factor $\beta \geq 0$ is defined as the ratio of the absolute difference in children values to that of the parent values:

$$\beta = \frac{c_i^2 - c_i^1}{p_i^2 - p_i^1}, \tag{24}$$

where c_i^1 and c_i^2 are the generated offspring and p_i^1 and p_i^2 are the selected parents. With the above definition of the spread factor, crossovers are classified as

- contracting crossovers ($\beta < 1$): the parent points enclose the children points;
- expanding crossovers ($\beta > 1$): the children points enclose the parent points;
- stationary crossovers ($\beta = 1$): the children points are the same as the parent points.

The probability distribution function of β in SBX is defined by

$$p(\beta_q) = \begin{cases} 0.5(\eta_c+1)\beta_q^{\eta_c} & \text{if } 0 \le \beta_q \le 1, \\ 0.5(\eta_c+1)\dfrac{1}{\beta_q^{\eta_c+2}} & \text{if } \beta_q > 1, \end{cases} \tag{25}$$

with η_c being the distribution index for crossover which determines how well spread the children will be from their parents. To get a value of β, noted as β_q, first a random number x between 0 and 1 is created. From the specified distribution function given by (25), the ordinate β_q is found so that the area under the probability curve from 0 to β_q is equal to the chosen random number x and β_q is calculated as follows:

$$\beta_q = \begin{cases} (2u)^{\frac{1}{\eta+1}} & \text{if } u \le 0.5, \\ \left[\dfrac{1}{2(1-u)}\right]^{\frac{1}{\eta+1}} & \text{otherwise,} \end{cases} \tag{26}$$

After obtaining β_q from the above probability distribution, the children solutions are calculated as follows:

$$c_1^i = 0.5[(1-\beta_q)p_i^1 + (1+\beta_q)p_i^2], \tag{27}$$

$$c_2^i = 0.5[(1+\beta_q)p_i^1 + (1-\beta_q)p_i^2]. \tag{28}$$

Polynomial mutation. The polynomial mutation is given by

$$c_k = p_k + (p_k^u - p_k^l)\delta_k, \tag{29}$$

where c_k is the child and p_k is the parent, with p_k^u and p_k^l being respectively the upper and the lower bound on the parent component, and δ_k is a small variation which is calculated from a polynomial distribution using the equations below:

$$\delta_k = \begin{cases} (2r_k)^{\frac{1}{\eta_m+1}} - 1 & \text{if } r_k < 0.5, \\ 1 - [2(1-r_k)]^{\frac{1}{\eta_m+1}} & \text{if } r_k \ge 0.5, \end{cases} \tag{30}$$

where r_k is a random number from the interval $(0, 1)$ and η_m is the mutation distribution index. In the present work, the efficiency of the NSGA-II algorithm is illustrated by the tuning of a GPC multivariable controller. In fact, this algorithm is used to minimize simultaneously a set of objectives (overshoot and settling time for each output) for generating the multivariable GPC synthesis parameters.

4. Simulation results

As a simulation example, we choose a MIMO system characterized by the next transfer function matrix with sampling time $T_e = 1$ s:

$$H(z^{-1}) = \begin{bmatrix} \dfrac{0.09z^{-1}}{1-0.9z^{-1}} & \dfrac{0.03z^{-1}}{1-0.7z^{-1}} \\ \dfrac{0.07z^{-1}}{1-0.5z^{-1}} & \dfrac{0.06z^{-1}}{1-0.9z^{-1}} \end{bmatrix}. \tag{31}$$

The multiobjective optimization problem treated is given by

$$\begin{aligned} &\min_{X\in\Omega} J = (D_1(X), T_{s_1}(X), D_2(X), T_{s_2}(X)) \\ &\text{with} \\ &X = [H_{p_1}, H_{c_1}, \rho_1, H_{p_2}, H_{c_2}, \rho_2] \quad \text{such as} \end{aligned} \tag{32}$$

$$\begin{aligned} &1 \le H_{p_j} \le 20, && j = 1, 2, \\ &1 \le H_{cl} \le 3, && l = 1, 2, \\ &0 < \rho_l \le 10, && l = 1, 2. \end{aligned}$$

To implement the evaluation phase using both proposed strategies, we choose two different intervals for each output of the given MIMO system. For the first output, we evaluate the overshoot $D_1(\%)$ and the settling time T_{s_1} for $k \in [1500, 2000]$. For the second output, the overshoot $D_2(\%)$ and the settling time T_{s_2} are evaluated for $k \in [3000, 3500]$.

4.1. First strategy.

To apply the genetic algorithm, we choose a population of 50 individuals and with 1, 10 and 100 generations. The crossover and the mutation probabilities are fixed respectively to $c_p = 0.7$ and $m_p = \frac{1}{6}$. To use the WSM, we vary w_1 between 0 and 0.98, and $w_i, i = 2, \ldots, 4$, are computed by

$$w_i = \frac{1 - w_1}{3}. \tag{33}$$

For every set of $w_i,\ i = 1, \ldots, 4$, the genetic algorithm evaluates the criterion given by (23) and generates the best individual (H_{p_j}, H_{c_l} and ρ_l) which will be used to implement the multivariable predictive controller. In Table 1, we reported the values of the best individuals corresponding to every set of weights obtained with a maximum number of generations equal to 100.

Table 1. Results of an optimization problem using the WSM.

Weights	w_1	0	0.98
	w_2	1/3	0.006
	w_3	1/3	0.006
	w_4	1/3	0.006
Optimal solutions	H_{p1}	2	3
	H_{c1}	1	2
	ρ_1	0.8367	0.1
	H_{p2}	4	3
	H_{c2}	1	1
	ρ_2	1.1238	1.052
Objective function values	$D_{\%1}$	2.594	6.473
	T_{s1}	53	40
	$D_{\%2}$	8.2958	25.422
	T_{s2}	129	54

Figures 3 and 4 describe respectively the non-dominated solutions which constitute the Pareto front for the first and the second outputs with 1, 10 and 100 generations. From these figures, we notice that this strategy does not guarantee the diversity of solutions in the Pareto front which results in the clustering of the obtained non-dominated solutions in some areas of the Pareto surface. On the other hand, this method is not able to generate all optimal solutions of the problem, which may cause the loss in good solutions. Also, it is clear that the increase in the maximum number of generations leads to the closeness of the solutions to the ideal one.

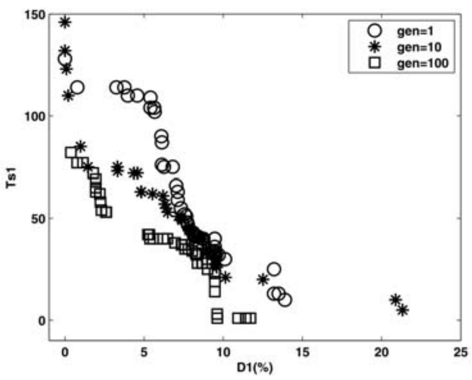

Fig. 3. Pareto front for the first output obtained using the WSM.

4.2. Second strategy. The optimization problem involves the four mentioned objective functions, which have different behaviours. The objective functions were optimized fulfilling the bounds given in Eqn. (32). As stated earlier, the NSGA-II algorithm was used for obtaining the Pareto-optimal solutions. The NSGA-II algorithm allows evolving a set of non-dominated solutions that are all equally well suited for solving the specific problem. In this work, the chosen NSGA-II parameters are given in Table 2.

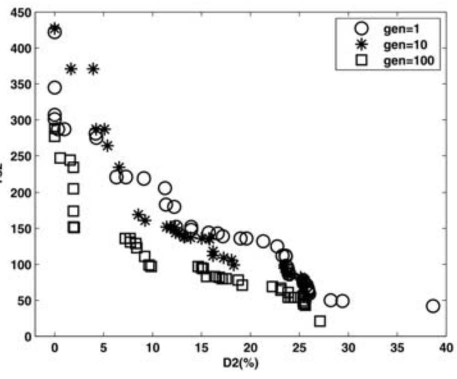

Fig. 4. Pareto front for the second output obtained using the WSM.

Table 2. NSGA-II parameters.

Parameters	Values
Number of decision variables	6
Number of objective functions	4
Population size	50
Maximum generations	1 or 10 or 100
Distribution indexes for crossover	20
Distribution indexes for mutation	20
Crossover probability	0.7
Mutation probability	1/6

Applying NSGA-II with 100 generations, we obtain the results given in Table 3 where for each population we present the obtained synthesis parameters and the corresponding closed-loop performances. Then, in Figs. 5 and 6, we have respectively the Pareto front for the first and second outputs with 1, 10 and 100 generations. These figures show good diversity and distribution of the non-dominated solutions along the Pareto surface. On the other hand, the increase in the maximum number of generations leads to the closeness of the solutions to the ideal one.

To make a better comparison between the performances of both adopted strategies, the Pareto fronts obtained using the first method and the second

Table 3. Results of an optimization problem using NSGA-II.

N of solution	1	2	49	50
H_{p1}	4	10	15	14
H_{c1}	1	2	2	2
ρ_1	0.7782	4.8291	6.456	7.4284
H_{p2}	17	5	13	12
H_{c2}	3	2	2	2
ρ_2	6.0209	2.2686	2.386	2.1736
$D_{\%1}$	3.494	0.166	1.194	0.4713
T_{s1}	1	86	29	44
$D_{\%2}$	6.9627	0	0.404	0.0094
T_{s2}	14	153	74	107

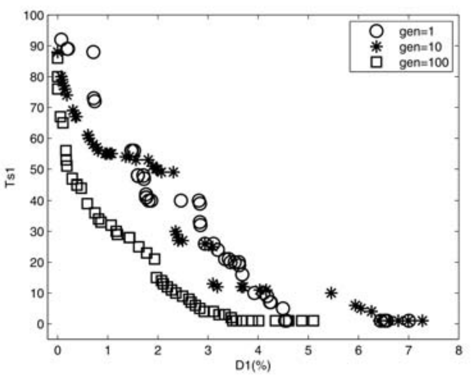

Fig. 5. Pareto front for the first output obtained using NSGA-II.

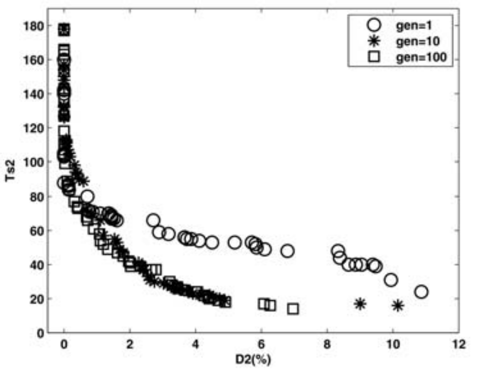

Fig. 6. Pareto front for the second output obtained using NSGA -II.

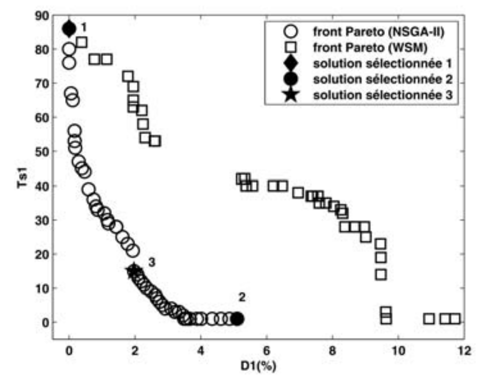

Fig. 7. Pareto fronts obtained with the WSM and NSGA-II for the first output and selected points for implementation.

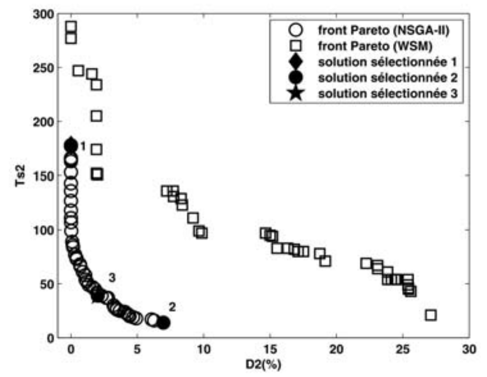

Fig. 8. Pareto fronts obtained with the WSM and NSGA-II for the second output and selected points for implementation.

method, with a maximum number of generations equal to 100, are shown together in Figs. 7 and 8.

From these figures, we notice that NSGA-II gives a better approximation of the Pareto front. In fact, the resulting solutions on the Pareto surface generated by the WSM are clustered in three areas for the first and second outputs, which results in gaps in Pareto surfaces. In contrast to the aggregative method, the uniform distribution gives an indication of a good spread of solutions along the Pareto surface obtained with NSGA-II and thanks to the concept of crowding distance; we succeed to maintain the diversity in the obtained set of solutions. It is also important to note that to obtain optimal solutions, which is an off-line procedure, the WSM needs 82554 seconds, whereas NSGA-II needs only 10513 seconds. This means that NSGA-II is approximately 8 times faster than the aggregative method.

4.3. MGPC implementation. Since all the optimal solutions are elaborated, it is necessary to choose only one solution to implement the MGPC. This choice is made by the decision-maker: if he or she decides to give the priority to the minimization of overshoot, the solution yielding the smallest value of the overshoot will be chosen. If the most important criterion to be minimized for the user is the settling time, the solution yielding the minimum settling time will be chosen. The solution making a compromise between all the closed loop performances can also be chosen. Therefore, after generating the possible optimal solutions, three of the solutions, numbered from 1 to 3 in Figs. 7 and 8, were chosen. The point number 1 is chosen to give the minimum values of overshoots D_1 and D_2. The solution number 2 is chosen to give the minimum values of T_{s_1} and T_{s_2}. And the solution 3 is chosen to make a trade-off between D_1 and T_{s_1} and a compromise between D_2 and T_{s_2}. The synthesis parameters corresponding to each selected point are presented in Table 4.

Table 4. Synthesis parameters corresponding to each selected point.

Solutions	H_{p1}	H_{c1}	ρ_1	H_{p2}	H_{c2}	ρ_2
1	10	2	5.479	5	2	2.445
2	17	2	5.474	17	2	6.020
3	14	2	7.958	17	1	3.205

Then, the equivalent simulation results which present the evolution of the system outputs and the set points and the evolution of the control signals for each chosen solution are given in Figs. 9–11.

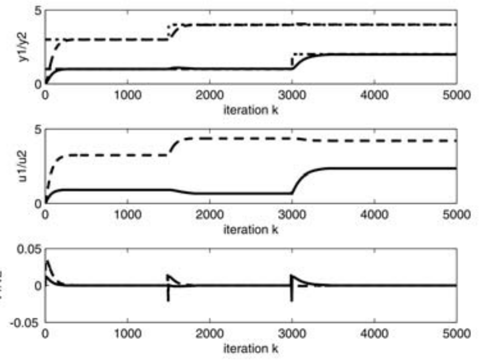

Fig. 9. Simulation results for Point 1.

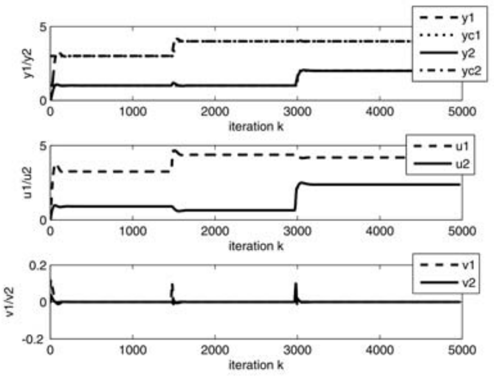

Fig. 10. Simulation results for Point 2.

In Table 5, we present the closed loop performances obtained with each chosen solution. From these results, we note that the obtained closed loop performances are equivalent to the preferences of the user and correspond to the choice of the decision-maker made previously. In addition, from the obtained results in Table 5, we conclude that, if we specify the closed-loop performances for the first output (/second output), the synthesis parameter adjustment done to obtain these desired performances has an influence on the closed loop performances of the second output (first output). This is due to interactions present in the treated system.

5. Conclusion

Two GA search techniques, aggregation into a single objective (WSM) and a Pareto-method (real-coded NSGA-II), have been used to adjust a centralized multivariable predictive controller parameters. Simulation results prove that NSGA-II yields a better approximation

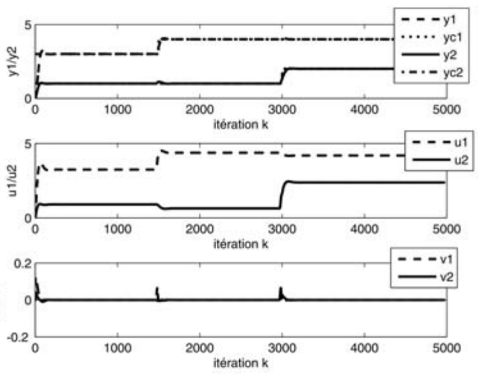

Fig. 11. Simulation results for Point 3.

Table 5. Values of the overshoots and settling times obtained with each selected solution.

Solutions	$D_{\%1}$	T_{s1}	$D_{\%2}$	T_{s2}
1	0	140	0	293
2	3.925	1	2.25	30
3	1.6201	20	1.1621	53

of the Pareto front than the WSM. The numerical results show the performance of the tuned controller with the NSGA-II method to control the treated multivariable process and to obtain closed loop performances corresponding to the decision-maker preferences. Therefore, it is concluded that NSGA-II is very effective in the tuning of MGPC synthesis parameters, is able to maintain a better spread of solutions and converges better in the obtained non-dominated front compared with the WSM. These results encourage the application of NSGA-II to more complex and real world multiobjective optimization problems. It is also important to note that the controllers considered are unconstrained ones and the proposed methodology can be applied to constrained predictive controllers.

References

Al-Gherwi, W., Budman, H. and Elkamel, A. (2010). Election of control structure for distributed model predictive control in the presence of model errors, *Journal of Process Control* **20**: 270–284.

Behroozsarand, A. and Shaffei, S. (2010). Optimal control of distillation column using non-dominated sorting genetic algorithm II, *Journal of Loss Prevention in the Process Industries* **24**(1): 25–33.

Bemporada, A. and Munoz de la Penab, D. (2009). Multiobjective model predictive control, *Automatica* **45**(12): 2823–2830.

Ben Abdennour, R., Ksouri, M. and Favier, G. (1998). Application of fuzzy logic to the on-line adjustment of the parameters of a generalized predictive controller, *Intelligent Automation and Soft Computing* **4**(3): 197–214.

Berro, A. (2001). *Optimisation multiobjectif et strat'egies d"evolution en environment dynamique*, Ph.D. thesis, Université des Sciences Sociales Toulouse I, Toulouse.

Boussaid, B., Aubrun, C., Abdelkrim, M.N. and Ben Gayed, M.K. (2011). Performance evaluation based fault tolerant control with actuator saturation avoidance, *International Journal of Applied Mathematics and Computer Science* **21**(3): 457–466, DOI: 10.2478/v10006-011-0034-x.

Camacho, E.F. and Bordons, C. (1995). *Model Predictive Control in the Process Industry*, Springer Verlag, London.

Clarke, W., Mohtadi, C. and Tuffs, P. S. (1987). Generalized predictive control, Parts 1 and 2, *Automatica* **23**(2): 137–160.

Coello Coello, C.A., Van Veldhuizen, D.A. and Lamont, G.B. (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer Academic Publishers, New York, NY.

Colette, Y. and Siarry, P. (2002). *Optimisation multiobjectif*, Éditions Eyrolles, Paris.

Deb, K. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation* **6**(2): 182–197.

Deb, K. and Agrawal, R.B. (1995). Simulated binary crossover for continuous search space, *Complex Systems* **9**: 115–148.

Gambier, A. (2007). Multi-objective optimal control: An overview, *Proceedings of the 16th IEEE International Conference on Control Applications, Singapore*, pp. 170–175.

Gambier, A. (2008). MPC and PID control based on multi-objective optimization, *Proceedings of the American Control Conference, ACC 2008, Seattle, WA, USA*, pp. 4727–4732.

Kinnaert, M. (1989). Adaptive generalized predictive controller for MIMO systems, *International Journal of Control* **50**(1): 161–172.

Królikowski, A. and Jerzy, D. (2001). Self-tuning generalized predictive control with input constraints, *International Journal of Applied Mathematics and Computer Science* **11**(2): 459–479.

Mohtadi, H., Shah, S. and Clarke, D. (1987). Generalized predictive control of multivariable systems, *Proceedings of the 5th Workshop on Applications of Adaptive Systems Theory, New Haven, CT, USA*, pp. 54–59.

Muldera, E.F., Tiwari, P.Y. and Kothare, M.V. (2009). Simultaneous linear and anti-windup controller synthesis using multiobjective convex optimization, *Automatica* **45**(3): 805–811.

Popov, A., Farag, A. and Werner, H. (2005). Tuning of a PID controller using a multi-objective optimization technique applied to a neutralization plant, *IEEE Conference on Decision and Control, Seville, Spain*, pp. 7139–7143.

Qin, S. and Badgwell, T. (2003). A survey of industrial model predictive control technology, *Control Engineering Practice* **11**: 733–764.

Richalet, J., Lavielle, G. and Mallet, J. (2005). *La commande prédictive: Mise en oeuvre et applications industrielles*, Éditions Eyrolles, Paris.

Srinivas, N. and Deb, K. (1995). Multiobjective function optimization using nondominated sorting genetic algorithms, *IEEE Transactions on Evolutionary Computation* **2**(3): 221–248.

Talbi, E.G. (2001). Learning logic, *Technical report*, Lille University of Sciences and Technologies, Lille.

Tatjewski, P. (2010). Supervisory predictive control and on-line set-point optimization, *International Journal of Applied Mathematics and Computer Science* **20**(3): 483–495, DOI: 10.2478/v10006-010-0035-1.

Veldhuizen, D. and Lamont, G. B. (2000). Multiobjective evolutionary algorithms: Analyzing the state-of-the-art, *IEEE Transactions on Evolutionary Computation* **18**(2): 125–147.

Yang, Z. and Pedersen, G. (2006). Automatic tuning of PID controller for a 1-D levitation system using a genetic algorithm: A real case study, *IEEE International Symposium on Intelligent Control, Munich, Germany*, pp. 3098–3103.

Zenghui, W., Zengqiang, C., Qinglin, S. and Zhuzhi, Y. (2006). Multivariable decoupling predictive control based on QFT theory and application in CSTR chemical process, *Chinese Journal of Chemical Engineering* **14**(6): 765–769.

Faten Ben Aicha is an electrical engineer and is presently a Ph.D. student at the National Engineering School of Tunis in Tunisia. Her current research interest includes predictive control, PID and multiobjective optimization.

Faouzi Bouani is a professor at the National Engineering School of Tunis. His current research interest includes predictive control, robust control, neuron networks and multiobjective optimization.

Mekki Ksouri is a professor at the National Engineering School of Tunis. His current research interest includes identification, adaptive control, robust control, neuron networks and nonlinear systems. He has published several monographs and textbooks.

Received: 20 August 2011
Revised: 11 January 2012
Re-revised: 18 March 2012