MODELING, OPTIMIZATION AND CONTROL OF A FCC UNIT USING NEURAL NETWORKS AND EVOLUTIONARY METHODS

Vitor Diego S. Bispo[1]
Elina Sandra R. L. Silva[2]
Luiz Augusto C. Meleiro[3,*]

Abstract
This paper presents a simulation study of the use of an artificial neural network (ANN) model for control and optimization of a Fluidized-Bed Catalytic Cracking reactor-regenerator system (FCC). This case study, whose phenomenological model was validated with industrial data, is a multivariable and nonlinear process with strong interactions among the operational variables. In order to obtain a dynamic model of the FCC system, a feed forward ANN model was identified. Genetic Algorithm (GA), and Particle Swarm Optimization (PSO) evolutionary methods were used to set optimal operating conditions for the FCC, and both algorithms presented good and consistent results for typical FCC optimization problems. The neural model was also used in the design of a Model-Based Predictive Control (MPC) for the FCC process. It was showed that the ANN-based MPC was able to reject the imposed disturbance as well as to track the proposed trajectory, while considering operational constraints of the plant.

Keywords: Fluid Catalytic Cracking; Genetic Algorithm; Particle Swarm Optimization; Model Predictive Control.

[1]Universidade Federal Rural do Rio de Janeiro (PPGEQ/UFRRJ), Brazil.
[2]Universidade Federal do Paraná (PIPE/UFPR), Brazil.
[3]Universidade Federal Rural do Rio de Janeiro, Brazil. Rodovia BR 465, km 7. Seropédica, RJ – 23890-000 – Brazil.
[*]Corresponding Author: meleiro@ufrrj.br

1. Introduction

Globalization has led to the necessity of optimizing the operation of chemical plants in such a way that the optimization and control strategies must be the more sophisticated as possible. However, the processes involved in oil refineries are generally very complex and, therefore, there is not yet a precise answer to the question of how is best way to operate them. FCC is one of the most important processes in petroleum refineries due to the high commercial values of its products, e.g. gasoline and liquefied petroleum gas (LPG). These features imply that even little improvements in the operation of this process may lead to large economic benefits. Due to its multi-variable, nonlinear features, complex dynamics, severe operating restrictions, and strong interactions among the process variables, FCC process is considered a challenging optimizing problem (Zanin et al., 2002; Vieira et al., 2005).

The literature is relatively rich in studies on modeling and simulation of FCC units (Avidan and Edwards, 1990; Avidan and Shinnar, 1990; Arbel et al., 1995; Moro and Odloak, 1995; Ellis et al., 1998). Some authors have developed models using fundamental principles, the so-called phenomenological models, where most of the equations that describe the process behavior are derived from mass, momentum, and energy balances, together with some empirical relations. Alaradi and Rohani (2002), and Jia et al. (2003) reviewed some published works on this subject, and found many FCC studies based on empirical or semi-empirical models. Jia et al. (2003) proposed an identification procedure based on state space models. Ali and Elnashaie (1997) employed a nonlinear MPC algorithm with a modified state estimation scheme to solve the problem of stabilizing a FCC unit around a high gasoline yield, but unstable, operation point. Kalra and Georgakis (1994) presented a study that can help the understanding of how nonlinearity might restrict the effectiveness of linear control strategies, and thus provide motivation for nonlinear strategies. Other contributions on this topic can be found in Yang et al. (1996), and Zanin et al. (2002).

The approach of most papers concerning the optimization of the FCC units are related to the selection of outputs and their appropriate reference values that will guarantee a probable optimal operation of the plant (Zanin et al., 2002). Thus, FCC modeling, control, and optimization have inspired the appearance of many papers in the literature during the last decades. Some of them can be found in McFarlane et al. (1993), Moro and Odloak (1985), Arbel et al. (1995), Ellis et al. (1998), Gouvêa and Odloak (1998a, 1998b), Odloak et al. (2000), Alaradi and Rohani (2002), Kasat et al. (2002), Zanin et al. (2002), Kasat and Gupta (2003), Vieira et al. (2005).

Artificial Neural Networks (ANN) have been widely applied to identification and control of nonlinear dynamic systems (Narendra and Parthasarathy, 1990; Ng, 1997; Hussain, 1999; Nørgaard et al., 2000). One of the main reasons for this success is the universal approximation capability of the ANN, i.e., such models are able to approximate to arbitrary accuracy any continuous mapping defined on a compact (closed and bounded) domain (Jones, 1987; Hornik, 1989; Cybenko, 1989).

In recent years, there has been a strong interest in the use of neural networks to describe chemical processes, due to their ability to approximate highly nonlinear systems (Ng, 1997; Zhan and Ishida, 1997; Hussain, 1999; Nørgaard et al., 2000; Piché et al., 2000). Different architectures of neural networks have been used as nonlinear models to advanced control and optimization algorithms (Zhan and Ishida, 1997; Qin and Badgwell, 1998; Kambhampati et al., 2000). Hussain (1999) presented a review of neural networks applications in process control and optimization, pointing out that these nonlinear input-output models are capable of identifying a great number of systems, and can be incorporated into various well-known nonlinear control and optimization methods. Alaradi and Rohani (2002) reported the use of neural network for the FCC dynamic identification and control. Vieira et al. (2005) implemented and evaluated the performance of an ANN-based model predictive control in a FCC unit. These authors reported that the main advantage of neural networks in comparison to the rigorous model is the CPU processing time.

Heuristic and meta-heuristic solution methods are widely used in practice, since they provide usable solutions to mathematical representations of real-world situations. According to Silver (2004), the increasing use of such optimization methods is due to they do not require the often restrictive assumptions of optimization routines, and permit the use of more representative (rigorous) models of the real-world problems. Moreover, the author highlights others reasons for utilizing such algorithms: *i)* Ease of implementation, *ii)* Show improvement over current practices, *iii)* Exhibits fast results, *iv)* Robustness, and *v)* Possibility of use within optimization routines.

A genetic algorithm (GA) is a search technique used in computer science to find approximate solutions to optimization and search problems. GA is a particular class of evolutionary algorithms (meta-heuristic solution methods) that uses techniques inspired by evolutionary biology such as inheritance, mutation, natural selection, and recombination (or crossover). In recent years, genetic algorithms have become a popular optimization tool, since it appears to provide a robust search procedure for solving difficult optimization problems. The feature of these algorithms is that they are based on ideas from the science of genetics and the process of natural selection. Kasat et al. (2002), and Kasat and Gupta (2003) obtained good results in the optimization of a FCC using GA-based methods. According to the authors, several other interesting multi-objective optimization problems can also be formulated and solved by using GA.

The term Particle Swarm Optimization (PSO) refers to a relatively new family of optimization algorithms. PSO is also a meta-heuristic solution method created by Kennedy and Eberhart (1995) on the 90's. It is easily implemented and has proven to be very effective and quick when applied to a diverse set of optimization problems. It is inspired by the swarming or collaborative behavior of biological populations. Hassan et al. (2005) proposed a comparison between PSO and GA obtaining the same quality solutions, but PSO presented superior computational efficiency. The authors, however, signalize that the difference in computational effort between PSO and GA is problem dependent.

Kasat and Gupta (2003) have discussed the suitability of using any reasonable model, even empirical, for simulation and for optimization of FCC units, provided the use of industrial data to tune the parameters usually associated with the model.

In accordance to this approach, in the present work an ANN-based model was identified using a representative data set of the operational variables obtained from a phenomenological model of an industrial FCC. This phenomenological model was tuned with industrial data by Moro and Odloak (1995), and was used to simulate a real FCC process – the Kellog Orthoflow F Converter – operating at the Petrobras' refinery of São José, State of São Paulo, Brazil. The neural model obtained was able to describe adequately the FCC dynamics for control and optimization purposes.

The proposed optimizing framework uses the identified neural model to predict the process dynamics according to each set of manipulated variables generated by the GA and/or PSO algorithms. The results are evaluated and classified according to their fitness. Evolutionary operators are then applied to the individuals (sets of manipulated variables) in order to obtain the set of manipulated variables that drives the process to some desired set-point and, simultaneously, obeys the process operational constraints.

The same neural model was also used in the design of a Non-Linear Model-Based Predictive Control for the FCC process. It was showed that the ANN-based MPC was able to reject the imposed disturbance as well as to track the proposed trajectory, while keeping the process constraints of both, the manipulated and the controlled variables, into their operational ranges.

The paper proceeds as follows: Section 2 presents the FCC process, its main features and behavior, as well as the description of the main process variables. Heuristic optimization methods are discussed in Section 3, where the GA and PSO evolutionary optimization methods are presented. Results are discussed in Section 4, separated into three parts: Process identification, process optimization, and process control. Finally, the concluding remarks are presented in Section 5.

2. The Fluid Catalytic Cracking Process

The FCC reactor/regenerator system studied in this work is the Kellog Orthoflow F Converter, operating at the Petrobras' refinery of São José, State of São Paulo, Brazil. The FCC system is represented schematically in Figure 1, and a detailed description of this unit can be found in Gouvêa (1997), Gouvêa and Odloak (1998a, 1998b), and Zanin et al. (2002).

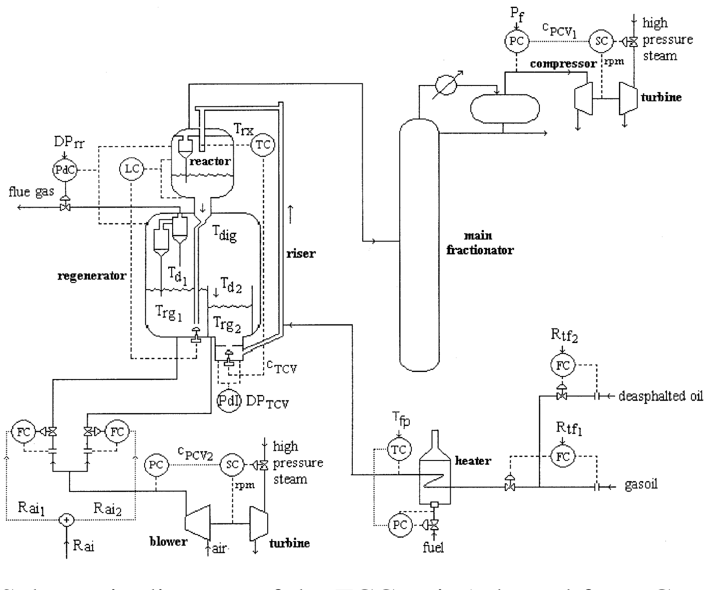

Figure 1 – Schematic diagram of the FCC unit (adapted from Gouvêa, 1997).

The FCC cracking reactions lead to lower boiling temperature products such as gasoline, liquefied petroleum gas (LPG), light cycle oil (LCO), butane, and gas. The converter is supposed to reduce the size of hydrocarbon chains of heavy petroleum fractions carried out by catalytic reactions. This process is extremely important because derived products of petroleum of high molecular weight (lower commercial value) are transformed into products of higher commercial values. The feed stock in this case study is gasoil. To guarantee adequate conversion and selectivity, the reaction temperature T_{rx} (^oC) must be kept in the range 520-550 ^oC. Secondary reactions generate coke, which is deposited on the surface of the catalyst. The spent catalyst is continuously regenerated by burning the coke under controlled conditions to prevent the catalyst deactivation (Moro and Odloak, 1995; Vieira et al., 2005).

There are two main streams that are fed into the reactor: The stream named deasphalted oil, R_{tf1} (m^3/h), which comes from the unit of treatment with propane, and the gasoil stream, R_{tf2} (m^3/h). These streams ($R_{tf1}+R_{tf2}=R_{tf}$) are pre-heated and sent to the riser, *RR*, which is a tube where the cracking reaction and coke deposition take place. In the riser, the inlet stream with temperature T_{fp} (^oC) is mixed with catalyst which comes from the regenerator, *RG*. The riser temperature has to be controlled to guarantee an adequate conversion of the cracking reaction. The opening of the catalyst valve, C_{TCV} (%), located at to the bottom of the riser, may be manipulated in such a way that the cracking reaction can be adequately controlled. The products of the reaction, in the gas phase, leave the riser and are separated from the catalyst in a separation vessel named reactor, *RX*. FCC products are sent to the main fractionator, *MF*, and the catalyst flows to the regenerator, where the catalyst is regenerated. Coke deposited on the surface of the catalyst is burned with air, whose flow rate, R_{ai} (ton/h), can be manipulated. The catalyst regenerator can be divided into several parts. Catalyst coming from the reactor is deposited on the first stage of the dense phase, rg_1, from where it falls into the second stage of the dense phase, rg_2, and is further sent to the riser. Gases from combustion fill in the so called diluted phase. Three distinct zones can be identified in the diluted phases from first, d_1, and second, d_2, stages (formed from the gases leaving the dense phases), and the general diluted phase, *dig*, formed from the gases coming from the first and second diluted phases. The flue gases from the general diluted phases are then sent to a boiler where CO is further converted into CO_2. The temperatures in the regenerator, T_{rg1}, T_{rg2}, T_{d1}, T_{d2}, and T_{dig} (^oC), must be controlled in order to avoid metallurgical damages (Zanin et al., 2002).

Due to extremely severe operating conditions, a rigorous control strategy is necessary. Several variables are chosen as controlled variables in the industry, depending on the operating point, economical objectives, as well as the priority of production of one of the products. In the present work, the main controlled variables are: the riser temperature (T_{rx}), temperatures of the dense phase in the first and second stages of the regenerator (T_{rg1} and T_{rg2}), and the cracking reaction severity, *Sev* (dimensionless). The manipulated variables are: Total feed flow-rate to riser (R_{tf}), opening of the catalyst valve (C_{TCV}), air flow-rate to regenerator (R_{ai}), and temperature of the feed stream (T_{fp}).

The reference steady state, and the operational range allowed for the main process variables are listed in Table 1.

Table 1 – Operational range and steady-state values of the main process variables.

Process Variables		**Operational Range**		**Steady State**
		max	**min**	
Controlled Variables	T_{rg1} (°C)	710	640	670.15
	T_{rg2} (°C)	710	640	700.89
	Sev	92	70	77.49
	T_{rx} (°C)	545	520	542.20
Manipulated Variables	R_{ai} (ton/h)	231	201	221
	C_{TCV} (%)	92	42	82
	R_{tf} (m^3/h)	410	208.4	404.17
	T_{fp} (°C)	245	215	235

3. Heuristic Optimization Methods

Typical chemical engineering problems involve several objective functions, often conflicting, non-commensurate, and with constraints. Moreover, these objective functions are non-trivial and one might need to carry out multi-objective optimization techniques in order to solve these real-world systems (Edgar and Himmelblau, 1989).

Unfortunately, classical optimization methods cannot provide, in many cases, solutions to these problems, and this lack encouraged the development, in the last decades, of several heuristic tools for solving such optimization problems. Among these new heuristic optimization methods, there are evolutionary heuristics, population-based search methods like Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) (Kasat and Gupta, 2003).

3.1 Genetic Algorithms

Genetic algorithms originated from the studies of cellular automata, in the 1980s, conducted by John Holland and his colleagues at the University of Michigan, but until 1989 no commercial application of GA was reported. Nowadays, however, custom computer applications began to emerge in a wide variety of fields, and these algorithms are now used by many companies to solve difficult scheduling, data fitting, trend spotting, budgeting and virtually any other type of combinatorial optimization. Since their inception decades ago, genetic algorithms, and its many versions, have evolved and becoming popular mainly because of its intuitiveness, ease implementation, and the ability to effectively solve typical complex engineering problems (Hassan et al., 2004; McCall, 2005). GA operates on a population of artificial chromosomes, each one representing a solution to the problem as a real number which measures how good this solution is to the specific problem (Michalewicz and Fogel, 2002; McCall, 2005).

Genetic algorithms are typically implemented as a computer simulation in which a population of abstract representations (called chromosomes) of candidate solutions (called individuals) to an optimization problem evolves toward better solutions. Traditionally, solutions are represented in binary format as strings of zeros and ones, [0, 1, 1, 0, 1, 0, …], but different encodings are also possible. These algorithms work with a population of solutions, and at each iteration, the fitness of the whole population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (mutated or recombined) to form a new population, which becomes current in the next iteration of the algorithm. The evaluations allow selecting a subset of the solutions to be either used directly in the next population or indirectly through some form of transformation or variation (Whitley, 1994; Michalewicz, 1999; Schmitt, 2001; Michalewicz and Fogel, 2002; Schmitt, 2004; Silver, 2004).

Two elements are required for any problem before a genetic algorithm can be used to search for a solution: **1)** There must be a method of representing a solution in such a way that it can be manipulated by the algorithm. Traditionally, a solution can be represented by a string of bits, numbers or characters, and; **2)** There must be some method of measuring the quality of any proposed solution, the fitness function. The fitness of the solution would be measured by determining the total weight of the proposed solution. The higher the weight, the greater the fitness, provided that the solution is possible. The main steps of the GA algorithm are summarized below (Whitley, 1994; Michalewicz, 1999; Schmitt, 2001; Michalewicz and Fogel, 2002; Schmitt, 2004; McCall, 2005):

Initialization**:** Initially, many individual solutions are generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Traditionally, the population is generated randomly, covering the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

***Selection*:** During each successive epoch, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based procedure, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. As this process may be very time-consuming, other methods rate only a random sample of the population. Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This procedure helps to keep the diversity of the population large, preventing premature convergence and poor solutions. Popular and well-studied selection methods include roulette wheel, and tournament.

***Reproduction*:** The next step is to generate a second generation population of solutions from those selected through genetic operators: crossover (or recombination), and mutation. To produce a new solution, a pair of "parent" solutions is selected for breeding from the pool selected previously. By producing a "child" solution, using the crossover and mutation operators, a new solution is created which typically shares many of the characteristics of its "parents"'. New parents are selected for each child, and the process continues until a new population of solutions of appropriate size is generated. These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally, the average fitness will have increased by this procedure, since only the best organisms from the first generation are selected for breeding.

***Termination*:** This process is repeated until a termination condition has been reached. Common terminating conditions are: **i)** A solution that satisfies the minimum criteria is found; **ii)** The defined number of generations is reached; **iii)** The highest ranking solution's fitness is reached.

The simplest algorithm represents each chromosome as a bit string. Typically, numeric parameters can be represented by integers, though it is possible to use floating point representations. The basic algorithm performs crossover and mutation at the bit level. Other variants treat the chromosome as a list of numbers which are indexes into an instruction table, nodes in a linked list, hashes, objects, or any other imaginable data structure. Crossover and mutation are performed so as to respect data element boundaries. For most data types, specific variation operators can be designed. Different chromosomal data types seem to work better or worse for different specific problem domains. A slight, but very successful variant of the general process of constructing a new population, is to allow some of the better organisms from the current generation to carry over to the next, unaltered. This strategy is known as elitist selection. Other variants, like genetic algorithms for online optimization problems, introduce time-dependence or noise in the fitness function (Whitley, 1994; Michalewicz, 1999; Schmitt, 2001; Michalewicz and Fogel, 2002).

Standard Genetic Algorithm:

1. Generate initial population.
2. Evaluate the fitness of every individual in the population.
3. Select pairs of best-ranking individuals (parents) to reproduce.
4. Breed new generation through crossover operator.
5. Select individuals to apply the mutation operator.

6. Use elitism operator to preserve the best solution.
7. Repeat steps 3 to 6 until terminating condition is reached.

3.1.1 The Genetic Algorithm for FCC Optimization

The specific GA implemented in this work was partially based on the paper of Hassan et al. (2004). The basic binary encoded GA with tournament selection was used. The crossover probability and mutation rate were fixed in 80% and 2%, respectively.

The genetic algorithm for FCC optimization was adapted in order to find the suitable set of manipulated variables (R_{tf}, T_{fp}, C_{TCV}, and R_{ai}) to drive the controlled variables of the process (T_{rx}, T_{rg1}, T_{rg2}, and Sev) from one operating point to another. In this study, the riser temperature, T_{rx}, was chosen as the main controlled variable. In this sense, the optimization problem is to reach a specific set-point for T_{rx}, while keeping the process constraints of both, the manipulated and the controlled variables, into their operational limits. The neural model of the process was used to predict the new operating points of the process, corresponding to each set of the manipulated variables generated by the GA. A brief description of the proposed GA is listed below:

- *Individuals*: Each individual represents a set of manipulated variables, [R_{tf}, T_{fp}, C_{TCV}, R_{ai}], generated by the genetic algorithm. The best individual, selected by the evolutionary operators, is the solution of the optimization problem. This solution corresponds to the set of manipulated variables that will drive the process to the specified set-point for T_{rx}. The initial population was created with 30 individuals, and the number of generations was set as 10.
- *Population*: Defined as the set of individuals, each one representing a set of manipulated variables, whose initial values were randomly generated by the GA.
- *Stop criteria*: In order to verify the convergence of the optimization algorithm, a stop criterion was defined as: $\left| \dfrac{T_{rx\_set} - T_{rx\_pred}}{T_{rx\_set}} \right| \leq 2\%$

where T_{rx_set} and T_{rx_pred} are the desired set-point, and the value of T_{rx} predicted by the neural model, respectively, for a given set of manipulated variables generated by the GA.

3.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a heuristic search method that was developed by Kennedy and Eberhart (1995) while attempting to simulate the choreographed, graceful, but unpredictable, motion of swarms of birds as a part of socio-cognitive study investigating the notion of collective intelligence in biological populations. PSO's mechanics are inspired by the collaborative behavior of biological populations. If one "individual" sees a desirable path to go (i.e., for food, protection, etc.), the rest of the swarm will be able to follow it quickly, even if they are on the opposite side of the swarm. PSO is similar to GA in the sense that these two evolutionary heuristics are population-based search methods. In other words, both methods move from a set of points (population) to another set of points in a single iteration, with likely improvement using a combination of deterministic and probabilistic rules. The main difference between these two methods is that the particle (individual) in the PSO is kept untouched, the individual only moves on the design space without getting any older (Hassan et al., 2004).

PSO is modeled by particles in multidimensional space that have a position and a velocity. These particles are flying through hyper-space, and remember the best position that they have seen. Members of a swarm communicate good positions to each other, and adjust their own position and velocity based on these good positions. The basic PSO algorithm consists of three steps

(Kennedy and Eberhart, 1995): **i)** Generate particles with specific positions and velocities; **ii)** Update velocity, and; **iii)** Update position.

The position and the velocity are updated through the following formulas at each iteration:

$$x = x + v \tag{1}$$

$$v = wv + c_1 r_1 \left(x^* - x\right) + c_2 r_2 \left(x_g^* - x\right) \tag{2}$$

where w is the inertial constant (standard values are usually slightly less than 1); c_1 and c_2 are constants that means how much the particle is directed towards good positions (good values are usually right around 1); r_1 and r_2 are random values in the range [0,1]; x^* is the best position the particle has seen; and x_g^* is the global best seen by the swarm, that can be replaced by x_l^*, the local best, if neighborhoods are being used. The standard PSO algorithm is the following:

Standard PSO Algorithm:

1. Initialize x and v of each particle with a random value. The range of these values may be domain specific.
2. Initialize each x^* to the current position.
3. Initialize x_g^* to the position that has the best fitness in the swarm.
4. Loop while the fitness of x_g^* is below a threshold, and the number of iterations is less than some predetermined maximum.

   For each particle, do the following:

   i. Update x according to the equation (1).
   ii. Calculate fitness for the new position.
   iii. If it is better than the fitness of x^*, replace x^*.
   iv. If it is better than the fitness of x_g^*, replace x_g^*.
   v. Update v according to the equation (2).

3.2.1 The PSO for FCC Optimization

The standard PSO algorithm was adapted for the FCC optimization problem in order to provide the suitable set of manipulated variables, (R_{tf}, T_{fp}, C_{TCV}, and R_{ai}), to reach the set-point for the riser temperature, T_{rx}. In this case, each particle in the swarm corresponds to a set of manipulated variables. It was established that the values of all particles should take into account the operational limits of the process variables before they are applied to the dynamic neural model. The particles that did not fit these limits were discharged. This procedure restricts the design space, and causes a decrease on the swarm size. However, this methodology helps to find the optimal set of manipulated variables, since the swarm is kept inside a suitable search space. In this work, the initial number of particles was set as 1000, with null initial velocity. The self-confidence, and the swarm confidence factors were set as $c_1 = 0.9$, and $c_2 = 1.1$, respectively.

4. MODEL PREDICTIVE CONTROL

The goal of a model predictive controller is to calculate a set of future control actions that minimize the following objective function:

$$J = \sum_{j=1}^{V_c} \delta_j \sum_{i=N_1}^{N_y} \left(\hat{y}^c_{j,k+i} - w_{j,k+i}\right)^2 + \sum_{j=1}^{V_m} \lambda_j \sum_{i=1}^{N_u} \left(\Delta u_{j,k+i-1}\right)^2 \tag{3}$$

where V_c and V_m are the number of controlled and manipulated variables, respectively, $\boldsymbol{\delta}$ and $\boldsymbol{\lambda}$ are weighting factors (tuning parameters). The parameters δ_j are used to add a degree of freedom in the adjustment of actions for each controlled variable, and λ_j are used to penalize the control actions. Δu_j are the increments in the manipulated variables, N_1 is the initial horizon, N_y is the prediction horizon, N_u is the control horizon, w_j are the reference trajectories (set-points) and $\hat{y}_j^c$ are the corrected model predictions.

The predictions are obtained recursively through N_y future predictions for each controlled variable, V_c, and corrected as follows:

$$\hat{y}^c_{k+i} = \hat{y}_{k+i} + d_k \quad (i = 1,\ldots,N_y) \tag{4}$$

In equation (4), $\hat{y}_{k+i}$ are the model predictions for the controlled outputs of the process at the future sampling time, $k+i$, and d_k are the correction terms given by $d_k = y_k - \hat{y}_k$, where y_k are the measured process outputs at the present sampling instant, and $\hat{y}_k$ are the respective predictions of the model (evaluated at the previous sampling instant). The optimization algorithm of the MPC structure calculates the increments of the control actions for the manipulated variables, Δu, in order to minimize the objective function given by the equation (3).

The future control actions are then derived from the optimized control increments, given by the equation (5).

$$u_{k+i} = u_{k+i-1} + \Delta u_{k+i} \quad (i = 0,\ldots,N_u - 1) \tag{5}$$

It is important to note that only the first N_u control actions are optimized, while the others are kept constant, i.e.:

$$u_{k+i} = u_{k+N_u-1} \quad (i = N_u,\ldots,N_y - 1) \tag{6}$$

The optimization problem is also subject to restrictions, and the receding horizon strategy was adopted (Clarke, 1994; Camacho and Bordons, 1999), where only the first control action for each variable manipulated (u_k) is implemented, and the optimization problem is solved again at each sampling instant.

The optimization problem (with restrictions) is solved by Successive Quadratic Programming technique (Edgar and Himmelblau, 1989).

For the specific case study of the FCC unit, the operational restrictions are given by:

- Restrictions imposed on the controlled variables:

$$\begin{aligned}
&540 \le T_{rg1(k+i)} \le 770 \quad (i = 1,\ldots,N_y)\\
&540 \le T_{rg2(k+i)} \le 770 \quad (i = 1,\ldots,N_y)\\
&50 \le Sev_{(k+i)} \le 100 \quad (i = 1,\ldots,N_y)\\
&400 \le T_{rx(k+i)} \le 700 \quad (i = 1,\ldots,N_y)
\end{aligned} \tag{7}$$

- Restrictions imposed on the manipulated variables:

$$\begin{aligned}
&201 \le R_{ai(k+i)} \le 231 \quad (i = 1,\ldots,N_u)\\
&0.45 \le C_{TCV(k+i)} \le 0.82 \quad (i = 1,\ldots,N_u)\\
&50 \le R_{tf(k+i)} \le 100 \quad (i = 1,\ldots,N_u)\\
&400 \le T_{fp(k+i)} \le 700 \quad (i = 1,\ldots,N_u)
\end{aligned} \tag{8}$$

- Restrictions imposed on the control actions:

$$\begin{aligned}
&\left|R_{ai(k+i)} - R_{ai(k+i-1)}\right| \le 1.5 \quad (i = 1,\ldots,N_u)\\
&\left|C_{TCV(k+i)} - C_{TCV(k+i-1)}\right| \le 0.015 \quad (i = 1,\ldots,N_u)\\
&\left|R_{tf(k+i)} - R_{tf(k+i-1)}\right| \le 50 \quad (i = 1,\ldots,N_u)\\
&\left|T_{fp(k+i)} - T_{fp(k+i-1)}\right| \le 2.2 \quad (i = 1,\ldots,N_u)
\end{aligned} \tag{9}$$

5. RESULTS AND DISCUSSIONS

The results related to the process identification, optimization, and control proposed to the FCC process are presented below.

5.1 Process Identification

The Artificial Neural Network (ANN) used in this work is a Multi-Input Multi-Output (MIMO) Multi-Layer Perceptron (MLP), with hyperbolic tangent and linear activation functions in the neurons of the hidden and output layers, respectively. The MLP model was trained with an optimization algorithm that uses the Modified Scaled Conjugate-Gradient (Castro and Von Zuben, 1998). The model accuracy was evaluated considering the final prediction error, and the performance of the neural model for both, one-step-ahead, and recursive predictions.

5.1.1 Data Generation and Sampling

A process simulator of the FCC process, validated by Moro and Odloak (1995) with experimental data at the Petrobras' refinery, was used to generate a representative data set that contains the input and output signals related to 600 hours of the process operation. Starting from the reference steady-state of this process, a sequence of steps was applied in the manipulated variables, and the responses of the process (the corresponding values of the controlled variables) were recorded.

5.1.2 ANN Structure Selection

The structure of the neural model was selected considering the main process variables. In this case, structure selection refers to the choice of the number of hidden neurons, the regression vectors, and the input and output variables of the neural model.

Many different network structures were tested, and the number of hidden neurons (N_{HN}), as well as the order of the regression vectors were obtained by trial-and-error procedures. Priority was given to parsimony principle, i.e., to the neural model with the small number of hidden neurons, as well as to those models with smaller regressors, especially with respect to the output variables because they are subject to predictions errors (over greater-than-one horizons). Four hidden neurons were found as the best choice for this model, and the regression vectors were set as one time delay for each controlled variable, and three time delays for each manipulated variable. Thus, the resulting neural model presents sixteen inputs:

$[R_{ai}\,(k\text{-}1), R_{ai}\,(k\text{-}2), R_{ai}\,(k\text{-}3)]$,
$[R_{tf}\,(k\text{-}1), R_{tf}\,(k\text{-}2), R_{tf}\,(k\text{-}3)]$,
$[C_{TCV}\,(k\text{-}1), C_{TCV}\,(k\text{-}2), C_{TCV}\,(k\text{-}3)]$,
$[T_{fp}\,(k\text{-}1), T_{fp}\,(k\text{-}2), T_{fp}\,(k\text{-}3)]$,
$[T_{rx}\,(k\text{-}1)]$,
$[T_{rg1}\,(k\text{-}1)]$,
$[T_{rg2}\,(k\text{-}1)]$,
$[Sev\,(k\text{-}1)]$

and four outputs:
$[T_{rx}\,(k)], [T_{rg1}\,(k)], [T_{rg2}\,(k)], [Sev\,(k)]$,

where k is the sampling instant.

The parameters of the proposed neural model were estimated using the first half of the generated data set, corresponding to 300 hours of input-output pairs. The second half of the data set was used for the validation of this model. The values presented in Table 2 correspond to the average (considering the results of three training procedures) of the mean squared error of the predictions errors provided by the selected neural model.

Table 2 – Mean squared prediction errors for the selected MLP MIMO model (N_{HN} = 4).

Prediction	T_{rg1} (°C)	T_{rg2} (°C)	**Severity**	T_{rx} (°C)
One-Step Ahead	0.0130	0.0217	0.0533	0.0450
Recursive Simulation	0.0929	0.1607	0.0965	0.0814

Figures 2 to 5 show that the resulting neural model was able to represent adequately the dynamic behavior of the controlled variables in both, One-step-ahead (top) and recursive predictions (bottom).

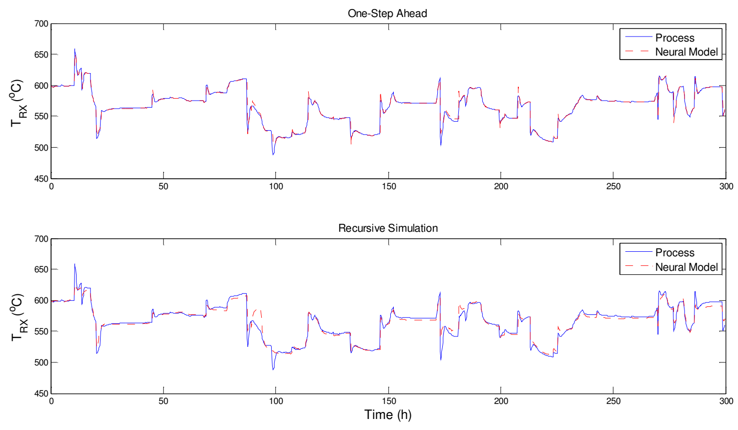

Figure 2 – Process output and neural model predictions for T_{rx}.

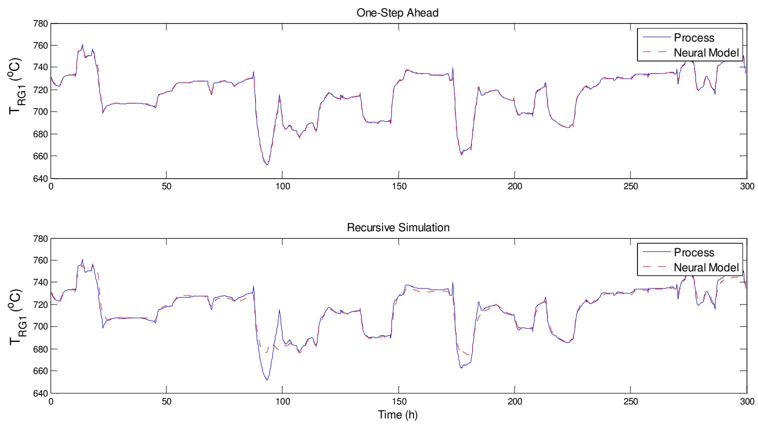

Figure 3 – Process output and neural model predictions for T_{rg1}.

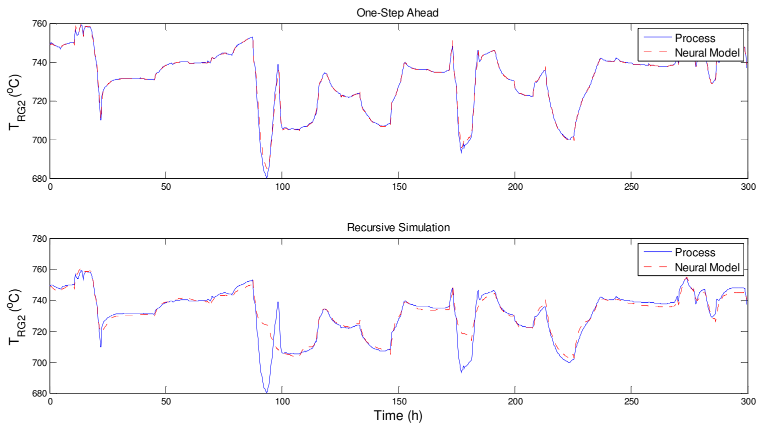

Figure 4 – Process output and neural model predictions for T_{rg2}.

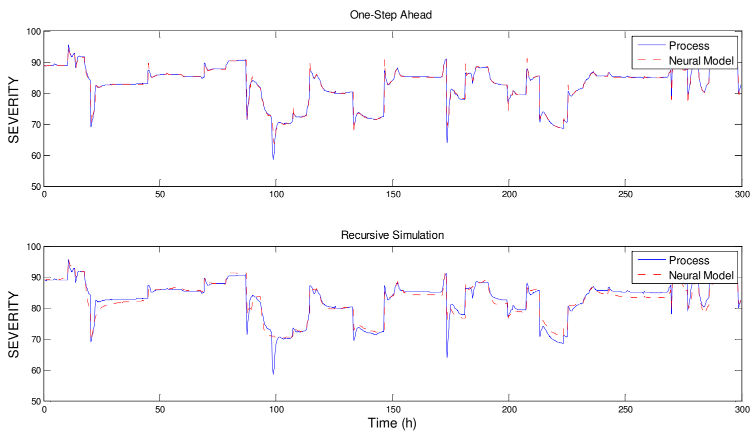

Figure 5 – Process output and neural model predictions for *Severity*.

These figures illustrate the good performance of the neural model, especially considering the long-range prediction horizons involved in recursive predictions.

5.2 Process Optimization

The results obtained with the proposed evolutionary optimization methods (PSO and GA) were compared each other in the FCC problem considering two performance tests suggested in Hassan et al. (2004). The first performance evaluation was carried out by the *effectiveness test*, which measures the quality of the solutions found by the heuristic algorithm with respect to suitable solutions for the proposed problem. The second test was a measure of *efficiency*. This test investigates the computational effort of the algorithms for a sample problem using the same convergence criteria.

In this work, the effectiveness was measured by how close the solutions provided by the evolutionary algorithms were to those provided by the phenomenological model. The effectiveness of the algorithms was evaluated with equation (10).

$$E = \left| \frac{Evol_{solution} - Real_{solution}}{Real_{solution}} \right| \% \qquad (10)$$

where $Evol_{solution}$ is the solution provided by the evolutionary algorithms, and $Real_{solution}$ is the solution provided by the phenomenological model, representing the plant outputs when subjected to the same set of manipulated variables imposed to the evolutionary algorithms.

Although this method does not guarantee that the solution obtained with the algorithm is the optimum, it allows us to verify that these solutions are not far from the real process responses, given by the phenomenological model. Moreover, despite of GA and PSO are different evolutionary methods, the solutions provided by both algorithms shown to be very similar (see Tables 3 to 5). These results suggest that the heuristic solutions should be, at least, near to the optimal solution for the proposed problem.

The first optimization problem requires that the evolutionary algorithms provide the suitable set of manipulated variables in order to operate the plant with the riser temperature set at $T_{rx} = 543.5\ ^{o}C$. The other controlled variables (T_{rg1}, T_{rg2}, and Sev) were allowed to vary inside their operational range, defined in Table 1. The stop criterion for both algorithms was defined as:

$$\left| \frac{T_{rx-Real} - T_{rx-Evol}}{T_{rx-Real}} \right| \leq 2\% \qquad (11)$$

where $T_{rx\text{-}Evol}$ is the solution provided by the evolutionary algorithm, and $T_{rx\text{-}Real}$ is the solution provided by the phenomenological model (representing the FCC outputs).

The results obtained with GA and PSO algorithms are shown in Table 3.

Table 3 – Results for the first optimization problem.

	Genetic Algorithm			Particle Swarm Optimization		
	GA	**Real**	**Error (%)**	**PSO**	**Real**	**Error (%)**
T_{rg1} (°C)	680.10	676.40	0.55	696.45	701.24	0.68
T_{rg2} (°C)	714.49	705.80	1.23	721.13	724.80	0.51
Severity	74.49	75.90	1.86	77.02	78.35	1.69
T_{rx} (°C)	543.68	543.99	0.06	543.51	547.03	0.64
R_{ai} (ton/h)	201.00	201.00		228.80	228.80	
C_{TCV} (%)	0.64	0.64		0.67	0.67	
R_{tf} (m^3/h)	353.65	353.65		361.84	361.84	
T_{fp} (°C)	226.90	226.90		232.60	232.60	

In order to evaluate the performance of the evolutionary algorithms in problems with a higher level of complexity, a second optimization problem was proposed. Now, the set-point for the riser temperature was kept the same ($T_{rx} = 543.5$ °C), but the operational range allowed for the temperature of the first stage was reduced to 673.0 °C $< T_{rg1} < 675.0$ °C.

The results obtained with GA and PSO algorithms are shown in Tables 4 and 5.

Table 4 – GA results for the second optimization problem.

	Test 1			Test 2			Test 3		
	GA	**Real**	**Error (%)**	**GA**	**Real**	**Error (%)**	**GA**	**Real**	**Error (%)**
T_{rg1} (°C)	673.32	666.68	0.99	673.23	671.62	0.24	674.00	669.01	0.74
T_{rg2} (°C)	714.51	696.71	2.55	715.18	702.96	1.74	715.43	699.03	2.35
Severity	79.56	76.64	3.81	75.57	76.53	1.26	75.17	76.75	2.06
T_{rx} (°C)	541.39	535.03	1.19	544.61	538.59	1.12	542.53	537.90	0.86
R_{ai} (ton/h)	207.22	207.22		225.92	225.92		207.45	207.45	
C_{TCV} (%)	0.73	0.73		0.79	0.79		0.74	0.74	
R_{tf} (m^3/h)	371.29	371.29		404.74	404.74		381.88	381.88	
T_{fp} (°C)	219.12	219.12		228.36	228.36		233.95	233.95	

Table 5 – PSO results for the second optimization problem.

	Test 1			Test 2			Test 3		
	PSO	**Real**	**Error (%)**	**PSO**	**Real**	**Error (%)**	**PSO**	**Real**	**Error (%)**
T_{rg1} (°C)	673.80	668.62	0.77	674.07	670.70	0.50	674.49	673.99	0.07
T_{rg2} (°C)	714.91	699.23	2.24	715.81	700.98	2.11	715.67	705.16	1.49
Severity	76.15	76.85	0.91	74.95	76.76	2.35	74.92	76.56	2.14
T_{rx} (°C)	543.50	536.37	1.33	543.46	539.53	0.73	543.50	540.21	0.61
R_{ai} (ton/h)	215.82	215.82		211.20	211.20		222.81	222.81	
C_{TCV} (%)	0.76	0.76		0.75	0.75		0.78	0.78	
R_{tf} (m^3/h)	9101.80	9101.80		9416.68	9416.68		9748.47	9748.47	
T_{fp} (°C)	215.68	215.68		240.57	240.57		238.90	238.90	

The results shown in Tables 4 and 5 reveal that the evolutionary algorithms were able to deal with the proposed optimization problems. Once more, it can be seen that the solutions found for both algorithms are very similar and coherent with the expected process behavior. However, PSO required less computational effort to find a suitable solution for the optimization problem.

The efficiency, or computational effort, was inferred by comparing the processing time of both algorithms for the same stop criteria. The average processing time (running on Intel Core2 Duo, 2.0 GHz) using GA was about 3.2 minutes, while the PSO algorithm spent about 0.9 minutes.

5.3 Process Control

The multivariate nature of the FCC unit, the strong interactions among the variables, the non-linear behavior of this process that leads to the need for a non-linear control, and the demand on operating the unit under process and materials constraints, are some of the key challenges in the design and implementation of the control of the FCC unit. Considering these features, the model-based predictive control (MPC) is a good candidate for implementing an advanced control in FCC units (Cristea et al., 2003).

In this work, the neural model identified for the FCC unit (Section 0) was used in the design of a non-linear model predictive control for this process, and the parameters of the controller were tuned as $N_1 = 1$, $N_y = 30$, $N_u = 2$, $\delta = \begin{bmatrix}10^{-5} & 10^{-5} & 10^{-3} & 10^{-1}\end{bmatrix}^T$, and $\lambda = \begin{bmatrix}0.01 & 0.01 & 0.0001 & 0.001\end{bmatrix}^T$.

The performance of the ANN-based MPC is presented below, where the controller was tested in servo and regulatory problems. In this sense, it was imposed that the main controlled variable (T_{rx}) should follow a reference trajectory, while the others controlled variables (T_{rg1}, T_{rg2}, Sev) should remain as close as possible to their steady-state values, and inside the operational range (see Table 1).

The behavior of the controlled variables under regulatory control problem is shown in Figures 6 to 9. In this test, a disturbance in the air temperature of the regenerator (from 190 °C to 200 °C) was imposed on the FCC process after 100 minutes of operation.

It can be noticed from Figure 6 that the main controlled variable, T_{rx}, was kept very close to the set-point by the multivariable ANN-based MPC.

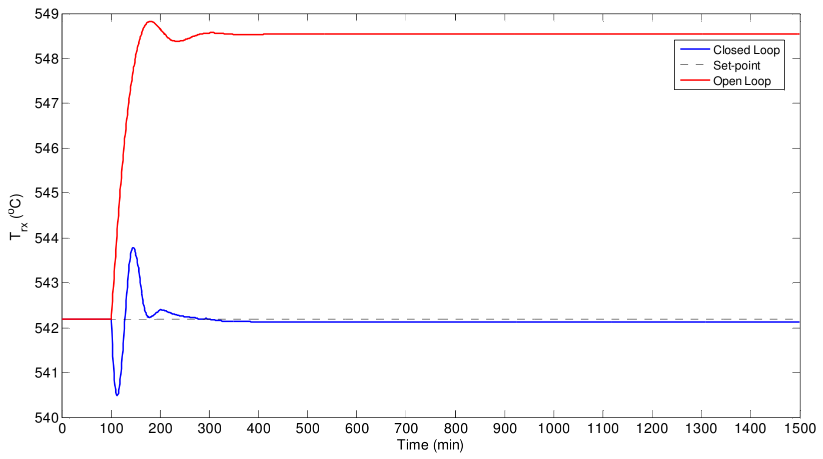

Figure 6 – Open and closed loop responses of T_{rx} in the regulatory control problem.

Figures 7 to 9 show that the others controlled variables (T_{rg1}, T_{rg2}, Sev) were kept inside the operational range, as well as that the MPC controller was able to keep them closer to their set-points when compared to the open loop response.

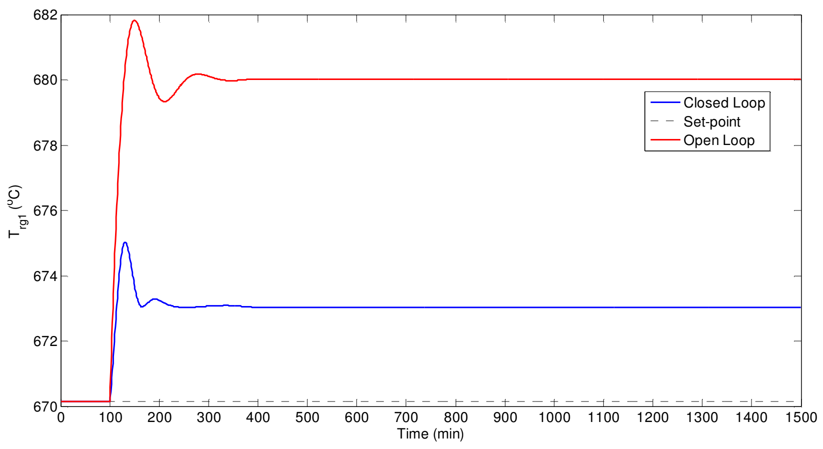

Figure 7 – Open and closed loop responses of T_{rg1} in the regulatory control problem.

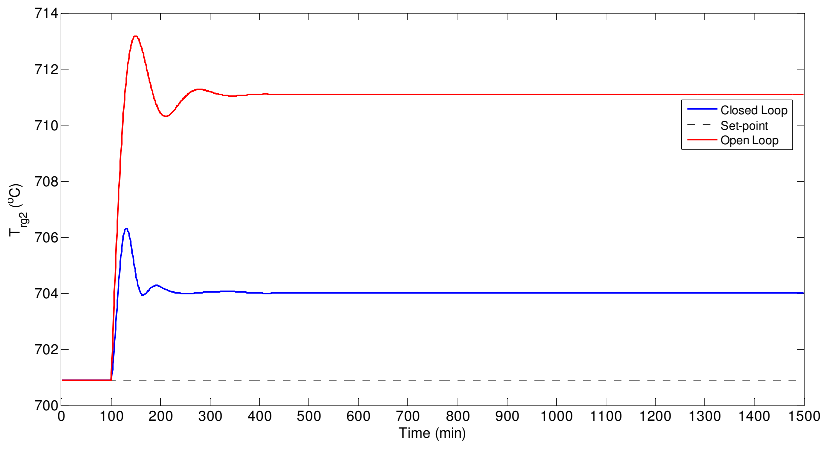

Figure 8 – Open and closed loop responses of T_{rg2} in the regulatory control problem.

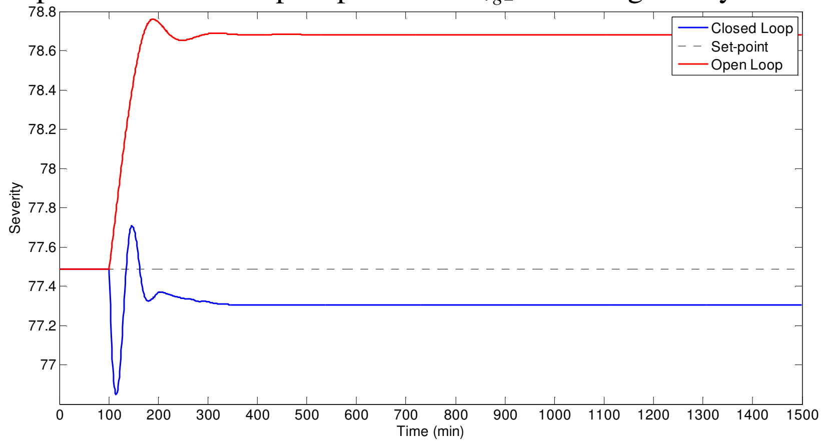

Figure 9 – Open and closed loop responses of *Severity* in the regulatory control problem.

The behavior of the main controlled variable under servo control problem is shown in Figure 10, where it is required to track a set-point change from 542.2 °C to 535.0 °C, while the others controlled variables should be kept as close as possible to their nominal values.

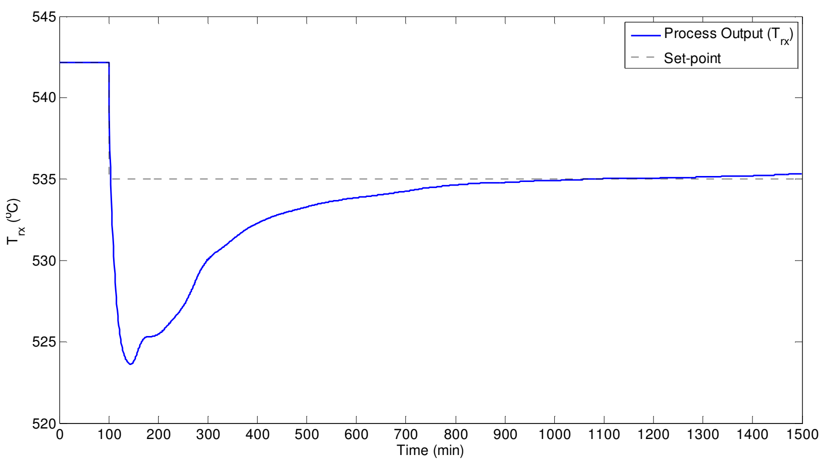

Figure 10 – Open and closed loop responses of T_{rx} in the servo control problem.

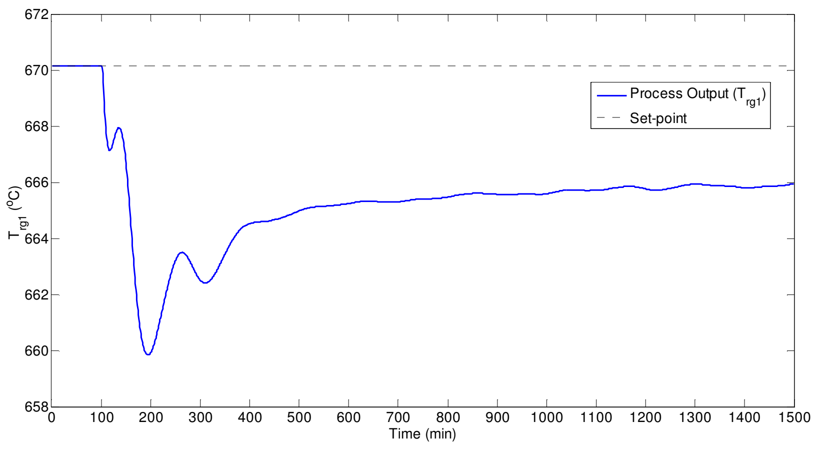

Figure 11 – Open and closed loop responses of T_{rg1} in the servo control problem.

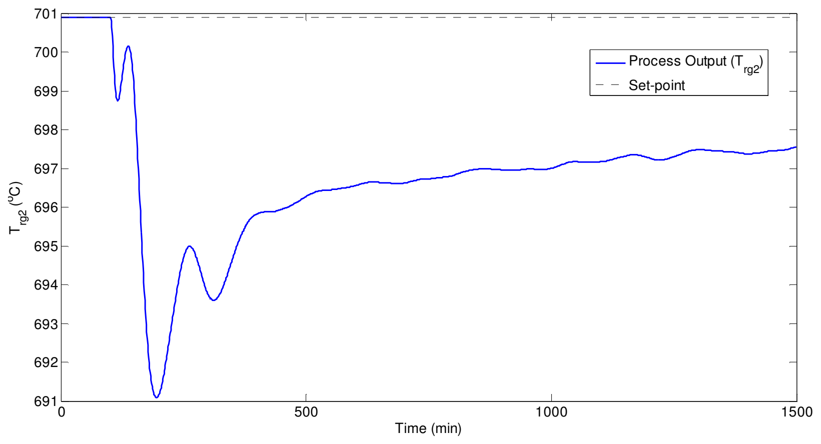

Figure 12 – Open and closed loop responses of T_{rg2} in the servo control problem.

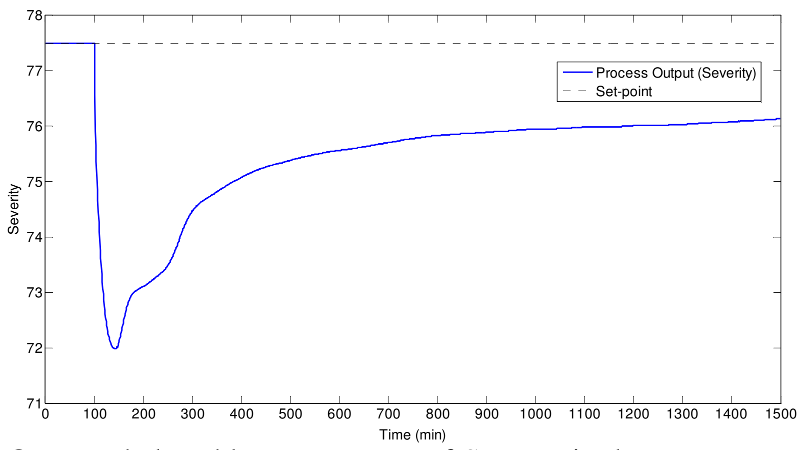

Figure 13 – Open and closed loop responses of *Severity* in the servo control problem.

The behavior of the controlled variables of the FCC unit in typical operating conditions (Figures 6 to 13) shows that the proposed ANN-based MPC was able to control the process appropriately.

6. CONCLUSIONS

In order to obtain a non-linear model of the FCC unit for further use in process optimization and control, an identification procedure for this process, based on artificial neural networks, was described. A fully connected multi-layer feed-forward network was chosen to model the FCC process from typical operational data, and the structure determination of the ANN (input and output variables, number of hidden neurons, and the regression vectors) was also discussed. The resulting neural model presents sixteen inputs, four hidden neurons, and four outputs. It was illustrated by simulations that the neural model was able to adequately describe the process dynamics, especially considering the good performance for long-range predictions.

One of the most interesting features of the ANN paradigm for modeling the FCC unit is the relative easiness for identifying a new model (set of adjustable parameters) when changes take place in the operating conditions, as typically occur in the Brazilian refineries due to changes in the crude oil to be processed, for example. In this case, the neural model can be adapted by a retraining procedure, adding a new data set with recent process conditions to the original database.

Two evolutionary optimization methods, Genetic Algorithm and Particle Swarm Optimization, were used in two optimization problems for the FCC process. The solutions provided by both algorithms showed to be similar and consistent with the expected process behavior, which indicates that the population-based optimization methods show great potential to be used in real-world process optimization problems.

Considering the operational constraints of the FCC unit, i.e., the complex dynamics of the process, the reaction that occurs in 2 to 5 seconds, besides the multivariate characteristic, nonlinear dynamics and strong coupling of variables of this process, the performance of the proposed ANN-based MPC proved to be satisfactory. The controller was able to reject the disturbance imposed on the process, as well as proved to be efficient to track a trajectory established for the riser temperature (T_{rx}), while keeping the others controlled variables within the operating range.

REFERENCES

ALARADI, A. A.; ROHANI, S. Identification and control of a riser-type FCC unit using neural networks. **Computer and Chemical Engineering**, v. 26, p. 401-421, 2002.

ALI, E. E.; ENASHAIE, S. S. E. H. Nonlinear model predictive control of industrial type IV fluid catalytic cracking (FCC) units for maximum gasoline yield. **Industrial and Engineering Chemistry Research**, v. 36, p. 389-398, 1997.

ARBEL, A.; HUANG, Z.; RINARD, I. H.; SHINAR, R.; SAPRE, A. V. Dynamics and control of fluidized catalytic crackers: Modeling and current generation FCCs. **Industrial and Engineering Chemistry Research**, v. 34, p. 1228-1243, 1995.

AVIDAN, A. A.; EDWARDS, M.; OWEN, H. Fluid Catalytic Cracking: past and future challenges. **Reviews in Chemical Engineering**, v. 6, n. 1, p. 1-71, 1990.

AVIDAN, A. A.; SHINNAR, R. Development of catalytic cracking technology. A lesson in chemical reactor design. **Industrial and Engineering Chemistry Research**, v. 29, p. 931-942, 1990.

CAMACHO, E. F.; BORDONS, C. **Model Based Predictive Control**. Springer, 1999.

CASTRO, L. N.; Von ZUBEN, F. J. Optimised Training Techniques for Feedforward Neural Networks. **Technical Report DCA-RT 03/98**, Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação, Campinas – SP, Brasil, 1998.

CLARKE, D. W. **Advances in Model Based Predictive Control**. Oxford University Press, 1994.

CRISTEA, M. V.; AGACHI, S. P.; MARINOIV, V. Simulation and model predictive control of a fluid catalytic cracking unit. **Chemical Engineering and Process**, v. 42, p. 67-91, 2003.

CYBENKO, G. Approximation by superpositions of a sigmoidal function. **Mathematics of Control, Signals, and Systems**, v. 2, n. 4, p. 303-314, 1989.

EDGAR, T. F.; HIMMELBLAU, D. M. **Optimization of Chemical Processes**. McGraw-Hill Book Co., 1989.

ELLIS, R. C.; LI, X.; RIGGS, J. B. Modeling and optimization of a model IV fluidized catalytic cracking unit. **AIChE Jounal**, v. 44, n. 9, p. 2068-2080, 1998.

GOUVÊA, M. T. Uso de um **Algoritmo SQP na Otimização de Processos Químicos Contínuos em Tempo Real**. 1997. 278 f. Tese de Doutorado. Escola Politécnica da Universidade de São Paulo, 1997. (in Portuguese).

GOUVÊA, M. T.; ODLOAK, D. On-line maximization of LPG in the FCC unit. **Latin American Applied Research**, v. 28, p. 107-114, 1998a.

GOUVÊA, M. T.; ODLOAK, D. One-layer real time optimization of LPG production in the FCC unit: Procedure, advantages, and disadvantages. **Computers and Chemical Engineering**, v. 22, p. S191-S198, 1998b.

HASSAN, R.; COHANIM, B.; WECK, O.; VENTER, G. A comparison of particle swarm optimization and the genetic algorithm, **American Institute of Aeronautics and Astronautics**, 2004.

HORNIK, K. Multilayer feedforward networks are universal approximators. **Neural Networks**, v. 2, n. 5, p. 359-366, 1989.

HUSSAIN, M. A. Review of the applications of neural networks in chemical process control – Simulation and online implementation. **Artificial Intelligence in Engineering**, v.13, p. 55-68, 1999.

JIA, C.; ROHANI, S.; JUTAN, A. FCC unit modeling, identification and model predictive control, a simulation study, **Chemical Engineering and Processing**, v. 42, p. 311-325, 2003.

JONES, L. K. On a conjecture of Huber concerning the convergence of projection pursuit regression. **The Annals of Statistics**, v. 15, p. 880-882, 1987.

KALRA, L.; GEORGASKIS, C. Effect of process nonlinearity on the performance of linear model predictive controllers for the environmentally safe operation of a fluid catalytic cracking unit. **Industrial & Engineering**

Chemistry Research, v. 33, p. 3063-3069, 1994.

KAMBHAMPATI, C.; MASON, J. D.; WARWICK, K. A Stable One-Step-Ahead Predictive Control of Non-Linear Systems. **Automatica**, v. 36, p. 485-495, 2000.

KASAT, R. B.; GUPTA, S. K. Multi-objective optimization of an industrial fluidized bed catalytic cracking unit using genetic algorithm with the jumping genes operator. **Computers and Chemical Engineering**, v. 27, p. 1785-1800, 2003.

KASAT, R. B.; KUNZRU, D.; SARAF, D. N.; GUPTA, S. K. Multi-objective Optimization of Industrial FCC Units Using Elitist Non-dominated Sorting Genetic Algorithm. **Industrial & Engineering Chemistry Research**, v. 41, p. 4765-4776, 2002.

KENNEDY, J.; EBERHART, R. Particle Swarm Optimization. **Proceedings of the IEEE International Conference on Neural Networks**, 1995. p. 1942 – 1948.

MCCALL, J. Genetic Algorithms for modelng and optimization. **Journal of Computational and Applied Mathematics**, v. 184, p. 205 – 222, 2005.

MCFARLANE, R. C.; REINEMAN, R. C.; BARTEE, J. F.; GEORGAKIS, C. Dynamic simulator for a model IV fluid catalytic cracking unit. **Computers and Chemical Engineering**, v. 17, p. 275-300, 1993.

MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs**. Springer, 1999.

MICHALEWICZ, Z.; FOGEL, D. B. **How to solve it: Modern heuristics**. Springer, 2002.

MORO, L. F. L.; ODLOAK, D. Constrained multivariable control of a fluid catalytic cracking. **Journal of Process Control**, v. 5, n. 1, p. 29-39, 1995.

NARENDRA, K. S.; PARTHASARATHY, K. Identification and control of dynamic systems using neural networks. **IEEE Transactions on Neural Networks**, v. 1, n. 1, p. 4-27, 1990.

NG, G. W. **Application of Neural Networks to Adaptive Control of Nonlinear System**. John Wiley & Sons Inc., 1997.

NORGAARD, M.; RAVN, O; POULSEN, N. K.; HANSEN, L. K. **Neural Networks for Modelling an Control of Dynamic Systems**. Springer-Verlag, 2000.

ODLOAK, D.; ZANIN, A. C.; GOUVÊA, M. T. Industrial implementation of a real-time optimization strategy for maximizing production of LPG in a FCC unit. **Computers and Chemical Engineering**, v. 24. p. 525-532, 2000.

PICHÉ, S.; SAYYAR-RODSARI, B.; JOHNSON, D.; GERULES, M. Nonlinear Model Predictive Control using Neural Networks. **IEEE Control Systems Magazine**, p. 53-62, 2000.

QIN, S. J.; BADGWELL, T. A. **An Overview of Industrial Model Predictive Control Technology**. Available at: <www.che.utexas.edu/~qin/cpcv/cpcv14.html>. Accessed on: 29 March 1996.

SCHMITT, L. M. Theory of Genetic Algorithms. **Theoretical Computer Science**, v. 259, p. 1-61, 2001.

SCHMITT, L. M. Theory of Genetic Algorithms II: Models for genetic operators over the string-tensor representation of populations and convergence to global optima for arbitrary fitness function under scaling. **Theoretical Computer Science**, v. 310, p. 181-231, 2004.

SILVER, E. A. An overview of heuristic solution methods. **Journal of the Operational Research Society**, v. 55, n. 9, p. 936 – 956, 2004.

VIEIRA, W. G.; SANTOS, V. M. L; CARVALHO, F. R.; PEREIRA, J. A. F. R.; FILETI, A. M. F. Identification and predictive control of a FCC unit using a MIMO neural model. **Chemical Engineering and Processing**, v. 44, p. 855-868, 2005.

WHITLEY, D. A genetic algorithm tutorial. **Statistics and Computing**, v. 4, p. 65-85, 1994.

YANG, H. S.; WANG, Z. X.; MCGREAVY, C. A multivariable coordinated control system based on predictive control strategy for FCC reactor-regenerator system, **Chemical Engineering and Science**, v. 51, p. 2977-2982, 1996.

ZANIN, A. C.; GOUVÊA, M. T.; ODLOAK, D. Integrating real-time optimization into the model predictive controller of the FCC system. **Control Engineering Practice**, v. 10, p. 819-831, 2002.

ZHAN, J.; ISHIDA, M. The Multi-Step Predictive Control of Nonlinear SISO Processes with a Neural Model Predictive Control (NMPC) Method. **Computers and Chemical Engineering**, v. 21, p. 201-210, 1997.