New optimal controller tuning method for an AVR system using a simplified Ant Colony Optimization with a new constrained Nelder-Mead algorithm

M.J. Blondin[a,1,*], J. Sanchis[b], P. Sicard[a], J.M. Herrero[b]

[a]*Research Group in Industrial Electronics, Department of Electrical and Computer Engineering, Université du Québec Trois-Rivières (UQTR), 3351, boul. des Forges, Trois-Rivières, Canada*
[b]*Instituto Universitario de Automática e Informática Industrial, Universitat Politécnica de Valencia, Camino de Vera sn 46022, Valencia, Spain*

Abstract

In this paper, an optimal gain tuning method for PID controllers is proposed using a novel combination of a simplified Ant Colony Optimization algorithm and Nelder-Mead method (ACO-NM) including a new procedure to constrain NM. To address Proportional-Integral-Derivative (PID) controller tuning for the Automatic Voltage Regulator (AVR) system, this paper presents a meta-analysis of the literature on PID parameter sets solving the AVR problem. The investigation confirms that the proposed ACO-NM obtains better or equivalent PID solutions and exhibits higher computational efficiency than previously published methods. The proposed ACO-NM application is extended to realistic conditions by considering robustness to AVR process parameters, control signal saturation and noisy measurements as well as tuning a two-degree-of-freedom PID controller (2DOF-PID). For this type of PID, a new objective function is also proposed to manage control signal constraints. Finally, real time control experiments confirm the performance of the proposed 2DOF-PIDs in quasi-real conditions. Furthermore, the efficiency of the algorithm is confirmed by comparing its results to other optimization algorithms and NM combinations using benchmark functions.

Keywords:
Automatic Voltage Regulator, PID controller, Optimization, Nelder-Mead algorithm, Ant Colony Optimization

1. Introduction

The PID controller is ubiquitous in a variety of industrial systems and processes [1]. Its design relies on determining controller parameters to achieve closed loop system requirements.

*Corresponding author
Email addresses: maudejosee.blondin@uqtr.ca (M.J. Blondin), jsanchis@isa.upv.es (J. Sanchis), pierre.sicard@uqtr.ca (P. Sicard), juaherdu@isa.upv.es (J.M. Herrero)

[1]This work was supported by the Vanier Canada Graduate Scholarship, the Michael Smith Foreign Study Supplements Program from the Natural Sciences and Engineering Research Council of Canada and by the Ministerio de Economía y Competitividad (Spain), project DPI2015-71443-R. It was also supported by the Bourse Mobilité Étudiante from Ministère de l'Éducation du Québec, the CEMF Claudette MacKay-Lassonde Graduate Engineering Ambassador Award and the SWAAC Bourse au mérite pour étudiantes de cycles supérieurs.

Several methods and tuning rules have been proposed specifically for conventional PID controllers. Although there are classical tuning methods, such as Ziegler-Nichols, pole placement and Cohen-Coon, these methods rely on low order model approximations, hardly achieve high dynamic performance and barely cope with process constraints, which are omnipresent and essential in a realistic context [2, 3]. To cope with uncertainty and repeating disturbances, iterative learning control methods are efficient. They can achieve low transient tracking error, but they can only be applied to single and repeated operations [4]. Therefore, for industrial systems subjected to rapid and unpredictable changes, other intelligent control strategies, such as neural network (NN) controllers, can be applied. For example, an auto-tuning controller method based on intelligent NN and relay feedback approach is proposed in [5]. For systems with unknown parameters or uncertainties, NN based on PID using an extended Kalman filter algorithm [6] as well as grey relational analysis based approach [7] can be effective to improve controller performances.

Each controlled system has its own particularities and operating conditions, which the controller tuning method must consider. Henceforth, modern heuristic optimization (MHO) techniques, such as Particle Swarm Optimization (PSO) [8], Genetic Algorithms (GA) [9] and Artificial Bee Colony algorithm (ABC) [10] among others, have grown during the last two decades and have been applied to various control systems [11] handling performance assessment, design specification, and other attributes as multimodality and non-linearities [12]. Moreover, multiple system requirements can also be met during PID controller tuning [3].

To enhance PID tuning performances, several modifications within MHO algorithms have been proposed. For example, a modified version of PSO yields high performance with regard to robust PID controller design of two MIMO systems, a distillation column plant and a control system of an aircraft [13]. In [14], a combination of a stochastic population based optimization technique and a pattern search based method has been proposed for PI controller tuning, where an efficient combination strategy merges the advantage of each optimization algorithm. Following this line, in [15], PID tuning for a motion system with flexible transmission is performed with a combination of Ant Colony Optimization (ACO) [16] and Nelder-Mead method (NM) [17]. This proposition exhibits higher computational performances and the tuned PID achieves a better system response than controllers tuned by ACO or NM separately.

Indeed, different ACO-NM algorithms have been proposed in the literature. The combination in [15] is the original ACO algorithm with the unconstrained NM method. In this work, the changeover criterion to pass from ACO to NM is a fixed number of ACO iterations. In [18] and [19], this combination is thoroughly analyzed; sensitivity curves with respect to a stagnation criterion are proposed to make the ACO-NM tuning easier. Since some specific applications require parameter positivity, a positivity constraint procedure for the NM algorithm is proposed in [20]. It has been demonstrated that this constraint procedure is more effective than limiting the values near its constraints [21]. Table 1 shows different propositions presenting by columns the version of ACO algorithm, the changeover criterion to pass from ACO to NM, and the type of NM algorithm used. For example, in [22], a modified ACO, ACO_R, which includes a Gaussian kernel function, is proposed in order to suit better continuous optimization than the original ACO algorithm for benchmark functions. Its changeover criterion refers to a threshold on standard deviation between ACO solutions. Similarly, in [23], an Hybrid Continuous Interactive Ant Colony algorithm (HCIAC) combined with the original NM is proposed for large set of benchmark functions optimization. To make ACO parameter tuning easier, a probability function is proposed as a threshold to pass from ACO to NM. In [24], the ACO_R strategy merged with an upper and lower constrained NM method for bankruptcy prediction in banks is proposed. In this approach, the authors introduce a reflection procedure to constrain NM; the variable is reflected

Table 1: Characteristics of various ACO-NM propositions

Application/Purpose	ACO				Changeover			NM	
	Original	ACO_R	HCICA	Simplified	Number of iterations	Stagnation criterion	Threshold	Unconstrained	Constrained
PID tuning with anti-windup [15] for grinder table positioning	✓				✓			✓	
Statistical analysis of ACO for PID tuning [18, 19]	✓					✓		✓	
Tuning of four controller structures for grinder table positioning [20]	✓				✓				✓
Benchmark function evaluations [22]		✓					✓	✓	
Benchmark function evaluations [23]			✓				✓	✓	
Feed forward neural networks for bankruptcy prediction [24]		✓			✓		✓		✓
The proposed algorithm - PID and 2DOF tuning for AVR system				✓		✓			✓

when it exceeds its constraint. The ACO switches to the constrained NM algorithm when either ACO reaches a limited number of iterations or the standard deviation between ACO solutions comes to a specific value. From this ACO-NM algorithm review, this paper proposes an optimal controller tuning method using a new constrained ACO-NM algorithm. The main novelty of the proposed ACO-NM is that, unlike the other methods in Table 1, it uses a simplified version of ACO with a new constraint procedure for NM.

In this work, the proposed ACO-NM is applied to tune controllers for the original AVR system [25]. This controller tuning problem has received much interest in recent years; due to the amount of research on this system, it has become a solid benchmark for testing optimization algorithms. A comprehensive review of the numerous optimization algorithms that have been proposed for PID tuning for the original AVR system [25]-[40] are presented in Table 2.

Table 2: Optimization algorithms and cost functions for PID controller tuning for the original AVR system

Scenario	Ref.	Optimizer	Compared to	Cost function
A	[26]	BA	[27]	$ITAE \times ((1-e^{-\beta})(V_{t_{max}}+E_{ss})+e^{-\beta}(t_s-t_r))$
B	[27]	PSO	[25]	
C	[25]	PSO	GA	$((1-e^{-\beta})(V_{t_{max}}+E_{ss})+e^{-\beta}(t_s-t_r))$
D	[28]	Continuous Action Reinforcement Learning Automata (CARLA)	[25]	$((1-e^{-\lambda})(M_p+E_{ss})+e^{-\lambda}(t_s-t_r))$
E	[29]	Third PSO	∅	$\sum_{k=1}^{1000} k\lvert y(k)-y_r(k)\rvert + t_s^2 + 10^8 M_p^2$
F	[30]	Bacterial Foraging - Genetic Algorithm (BFGA)	GA, PSO and GA-PSO	$\frac{e^{-\beta}(t_s/max(t)+\alpha M_p)}{\alpha}+E_{ss}$
G	[31]	Extended DARLA	ZN, DARLA and GA	$G_e\int_0^T e^2(t)dt + G_u\int_0^T u_c^2(t)dt + G_M M_P + G_s E_{ss} + G_d sup\lvert\frac{de(t)}{d(t)}\rvert$
H	[32]	Simplified PSO	∅	$ITAE$
I	[33]	TLBO	∅	$ITAE$
J	[34]	Bacterial Foraging Algorithm	ZN and PSO	IAE
K	[35]	Pattern Search	[38]	$ITAE+\alpha_1\times\xi+\alpha_2\times t_s+\alpha_3\times t_r+\alpha_4\times V_{t_{max}}$
L	[36]	Modified PSO	GA and PSO	$0.5(1+M_P)(t_s+t_r)$
M	[37]	PSO	∅	$\omega_{M_P}M_P+\omega_{t_r}t_r+\omega_{t_s}$
N	[38]	ABC	PSO and DE	$ITSE$
O	[39]	Many Liaison Optimization	PSO	$\alpha_1\times ITAE+\alpha_2\times t_s+\alpha_3\times V_{t_{max}}$
P	[40]	Local Unimodal Sampling Algorithm	[38]	$\alpha_1\times ISE+\alpha_2 * t_r+\alpha_3 * V_{t_{max}}$

Regardless of the optimization algorithm, the cost function should reflect the control design expectations, i.e. a lower value of the objective function should provide better system responses according to the system requirements. There are typical criteria for PID tuning, such as the integral of time-weighted absolute error (ITAE), the integral of absolute error (IAE), the integral of time-weighted squared error (ITSE) and the integral of squared error (ISE). In [32, 33], [38], the authors use one of these typical criteria. However, in some cases, a lower value of typical criteria can provide an inferior system response quality [41]. To overcome the disadvantages of the typical criteria and target specific system requirements, other cost functions have been suggested. For example, in [25], the author proposes PSO to minimize a cost function, which specifically penalizes the system response characteristics, the rising time t_r, the settling time t_s, the steady-state error E_{ss} and the maximum peak value of the system response $V_{t_{max}}$. In [26, 27], this cost function multiplied by ITAE is optimized with Bat Algorithm (BA) and PSO respectively. PSO is also used in [37] and PSO based algorithms are proposed in [29], [36] to tune the PID controller for the AVR system with a different cost function respectively. The authors in [31] and [35] propose elaborated cost functions, but it can be challenging to tune them.

Whatever the optimization algorithm and the cost function, the ultimate goal for the system under study is to obtain a fast stable and robust system response with minimal overshoot. With this objective, the proposed ACO-NM tunes the PID controller for the original AVR system and it is extended to more realistic conditions by considering control signal saturation and filtering derivative action by means of a two degree of freedom PID structure [42]. Therefore, the remainder of this paper is organized as follows: §2 details the proposed ACO-NM, §3 presents ACO-NM performance compared with other algorithms thanks to well known benchmark functions, §4 introduces the AVR system and the PID controller, §5 presents the AVR system results and discussion where a new objective function is also proposed to manage control signal limits and real control experiments in quasi-real conditions are performed and §6 draws the main conclusions and presents future work.

2. The proposed ACO-NM

ACO is a metaheuristic inspired by ants behaviour within a colony initially proposed for combinatorial optimization [16] while NM method is a local unconstrained optimization method based on simplex [17]. It is the powerful ACO global search combined with the precise NM local optimization that makes the ACO-NM effective.

2.1. ACO algorithm

An ant colony facing multiple paths between their nest and a source of food has the tendency of taking the shortest road. This is explained by the fact that ants communicate via their environment by depositing traces of pheromones. The paths with the most pheromones are more attractive to other ants. Initially, ants take different paths from each other. However, the ant that takes the shortest road will do more round trips than others. Therefore, the amount of pheromones increases more rapidly on the shortest path attracting more and more ants. Eventually, all ants will take the same path, i.e. the shortest path. Henceforth, the solution emerges from the collective interaction of the ants.

An approach to exploit the ants' behavior in optimization is by representing the problem by an optimization graph as shown in Fig. 1 [43]. The ants move probabilistically on the optimization graph trying to minimize an objective function. They deposit pheromones on the nodes

of the traveled graph, which is represented by a pheromone matrix. The quantity of deposited pheromones, which are the pheromone matrix weights, increases with the quality of the response.

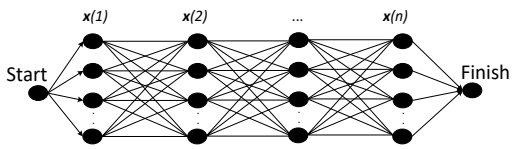

Figure 1: Optimization graph for ACO [43].

2.2. Nelder-Mead

Nelder-Mead simplex search is part of a general class of direct search methods. It is one of the most popular algorithms for solving unconstrained nonlinear optimization problems without using derivatives [17]. Generally, it is used for local optimization. NM algorithm tends to minimize a nonlinear scalar function of n variables by using only the evaluation of the objective function. Initially, NM algorithm performs its first simplex, Δ_1, of $n + 1$ vertices from a starting point x_0, where a vertex represents a variable set, $\boldsymbol{x}$, of n parameters. Thereafter, the simplex moves in the unconstrained search space in order to minimize an objective function according to different steps [17] ; *i)* Ordering, *ii)* Reflection, *iii)* Expansion, *iv)* Contraction and *v)* Shrinkage.

2.3. The proposed ACO-NM detailed

The proposed ACO-NM combines a simplified ACO algorithm with a new constrained NM algorithm. Simplified ACO means the construction of the ACO solution is only based on the pheromone matrix without weight. The advantages are: *i)* the simplified ACO is easier to tune than the original ACO *ii)* it suppresses the balancing requirement between the weights of the pheromone and the heuristic information matrices while the original ACO requires this balancing [16], and *iii)* it uses fewer mathematical operations within the algorithm. The changeover criterion to pass from ACO to NM refers to the stagnation of the cost function for a q number of consecutive ACO iterations [19]. This changeover is independent of the other parameters of the algorithm, which means no modifications or only a few trials are necessary when the proposed ACO-NM is applied to a different system. From the best ACO solution found, $\boldsymbol{x}_{ACO_{best}}$, the constrained NM algorithm makes the first simplex, Δ_1, of $n + 1$ vertices. The simplex moves in the constrained search space according to different steps [17] until it reaches one of its stop criteria. The proposed constraint procedure for NM manages any interval of search space values and considers the interaction among the variables to optimize, as opposed to [20] and [21], [24] respectively.

The simplified ACO followed by the constrained NM algorithm is performed iteratively minimizing a cost function f until the ACO-NM stop criterion is reached. The procedure is presented in Table 3.

Table 3: The proposed ACO - NM structure

```
ACONMInitialisation(UserInput)
while not (ACONMStopCriterion) do
    ACOInitialisation(τ)
    while not(ACOStopCriterion) do
        for k = 1 : m
            X(k, x_k) = BuildSolution
            fval_it(k, fval_x_k) = SolutionEval(x_k)
        end
        ACOevaluation
        UpDatePheromoneMatrix
    end while
    Δ_1 = PerformFirstSimplex(x_ACO_best)
    while not(NMStopCriteria) do
        OrderingVertices
        (x_r, f_r) = ComputeReflectionPt
        if f_1^(u) ≤ f_r^(u) < f_n^(u)
            BoundedTest(x_r)
            Introduce(x_r)
        elseif f_r^(u) < f_1^(u)
            (x_e, f_e) = ComputeExpansionPt
            if f_e^(u) < f_r^(u)
                BoundedTest(x_e)
                Introduce(x_e)
            else
                BoundedTest(x_r)
                Introduce(x_r)
            end
        elseif f_n^(u) ≤ f_r^(u) < f_n+1^(u)
            (x_oc, f_oc) = ComputeOutsideContractionPt
            if f_oc^(u) < f_r^(u)
                BoundedTest(x_oc)
                Introduce(x_oc)
            else
                PerformShrinkage
            end
        else
            (x_ic, f_ic) = ComputeInsideContractionPt
            if f_ic^(u) < f_n+1^(u)
                BoundedTest(x_ic)
                Introduce(x_ic)
            else
                PerformShrinkage
            end
        end
    end while
end while
Result: x_ACO-NM, fval_x_ACO-NM
```

The ***ACONMInitialisation*** initializes the proposed ACO-NM according to the user's specifications. These specifications includes the cost function f to minimize, the lower and upper bounds for the search range, $\boldsymbol{\varphi}_{min}$ and $\boldsymbol{\varphi}_{max}$, and the ACO-NM's parameters.

The ACO-NM's parameters are the number of ants within the colony, m, the search space discretization, ds, and the search space divisor, div. These values, $m = 15$, $ds = 20$ and $div = 100$, are effective according to the statistical results presented in [18, 19]; there are used as default values. The div value is used to create the ACO search space either when the bounds have opposite sign or when one of them is zero. For example, for the variable i, if $\boldsymbol{\varphi}_{min}$(i)=0, the search space for this variable is a vector of ds values logarithmically spaced from $\boldsymbol{\varphi}_{max}$(i)/$div$ to $\boldsymbol{\varphi}_{max}$(i). If the area near 0 is critical, div value should be higher than 100. Moreover, the user must provide at least one parameter of NMStopCriteria; the others will be inactive. The NMStopCriteria includes the maximum iteration number of NM, α, the maximum number of

function evaluations per NM iteration, Ω, and the tolerance of $fval_{\boldsymbol{x}}$, ϱ, where $fval_{\boldsymbol{x}}$ is the function value for $\boldsymbol{x}$. The ACONMStopCriterion is the maximum number of function evaluations, κ.

The ***ACOInitialisation*** initializes the pheromones matrix τ of dimension $n \times div$.

The ***BuildSolution*** refers to the ants moving probabilistically on the optimization graph shown in Fig. 1. From left to right, the ants choose one node in each column in order to build a complete solution. This graph can be represented by a matrix $\boldsymbol{N}$, $n \times ds$, where each column corresponds to one variable to optimize and each node is a quantized value within the search space. This representation allows the proposed ACO-NM to easily add or remove variables according to the optimization problem.

The first row of $\boldsymbol{N}$ is subjected to $\boldsymbol{\varphi}_{min}$ and the last row to $\boldsymbol{\varphi}_{max}$. The probability of an ant k choosing a parameter $\boldsymbol{N}(i, j)$ depends on:

$$P_{ij}^{k}(t) = \frac{\left[\tau_{ij}(t)\right]}{\sum_{j=1}^{ds}\left[\tau_{ij}(t)\right]} \tag{1}$$

where $\tau_{ij}(t)$ is the quantity of pheromones for the parameter $\boldsymbol{N}(i, j)$ stored in the pheromone matrix τ.

The ***SolutionEval*** steps saves the cost function value $fval_{\boldsymbol{x}_k}$ for an ant k in the vector $\boldsymbol{fval}_{it}$.

The ***ACOevaluation*** finds the best solution for the actual ACO iteration: $fval_{\boldsymbol{x}_{ACO_{it}}} = min\{\boldsymbol{fval}_{it}\}$. The best solution found prior to the actual iteration is defined as $fval_{\boldsymbol{x}_{ACO_{best}}}$. If $fval_{\boldsymbol{x}_{ACO_{it}}} < fval_{\boldsymbol{x}_{ACO_{best}}}$, $fval_{\boldsymbol{x}_{ACO_{best}}}$ takes the $fval_{\boldsymbol{x}_{ACO_{it}}}$ value and the stagnation parameter, *stag*, is reset to 0. If not, one is added to *stag*. When *stag* reaches the q value, the ACO algorithm stops and the constrained NM algorithm starts with $\boldsymbol{x}_{ACO_{best}}$. The q determination is based on [18]; its default value is 5.

The ***UpDatePheromoneMatrix*** is the memory process of ants' search, where a quantity of pheromone, $\triangledown\tau$, is deposited. A promising parameter receives more pheromone than a poor one. This step is computed as:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \sum_{k=1}^{m} \triangledown\tau_{ij}^{k}(t), \forall(i, j) \in \boldsymbol{x}^{k} \tag{2}$$

$$\triangledown\tau_{ij}^{k}(t) = \begin{cases} g(fval_{\boldsymbol{x}_k}, fval_{\boldsymbol{x}_{ACO_{best}}}) & = \dfrac{fval_{\boldsymbol{x}_k}}{fval_{\boldsymbol{x}_{ACO_{best}}}} \\ 0 & otherwise \end{cases} \tag{3}$$

$$\tau_{ij} = (1 - \Upsilon)\tau_{ij}, \forall(i, j) \tag{4}$$

where $\triangledown\tau_{ij}^{k}(t)$ is the quantity of pheromones deposited on the value $\boldsymbol{N}(i, j)$. The variable Υ is the evaporation rate; its default value is 0.2. The evaporation phenomenon avoids unlimited accumulation of pheromones and allows previous bad decisions to be forgotten. The evaporation phenomenon can be enforced on very poor parameters. For example, for PID tuning, if the system response is unstable according to the Routh-Hurwitz criterion, the evaporation phenomenon is applied twice and no pheromone is deposited on this solution.

The ***PerformFirstSimplex*** makes the first simplex Δ_1 from $\boldsymbol{x}_{ACO_{best}}$. The procedure is presented in Table 4.

The ***OrderVertices*** puts in ascending order, at the beginnings of uth iteration, the vertices of Δ_u as $x_1^{(u)}, \ldots, x_{n+1}^{(u)}$ such that $f_1^{(u)} \leq f_2^{(u)} \leq \cdots \leq f_{n+1}^{(u)}$, where $f_i^{(u)}$ means $f(x_i^{(u)})$ [17]. Since the

Table 4: ***PerformFirstSimplex*** procedure

```
Compute the vertex for each x ∀ j ∈ 1, ..., n
for j=1:n
    x_j = x_ACO_best
    if x_j(j) ≠ 0
        x_j(j) = (1 + 0.05)x_ACO_best(j)
        if x_j(j) < φ_min(j)
            Δ_bound = |φ_min(j) − φ_max(j)|
            x_j(j) = φ_min(j) + 0.25Δ_bound
        end
        if x_j(j) > φ_max(j)
            Δ_bound = |φ_min(j) − φ_max(j)|
            x_j(j) = φ_max(j) − 0.25Δ_bound
        end
    else
        x_j(j) = 0.00025
    end
end
Δ_1 = (x_ACO_best, x_j) where j = 1, ..., n
```

algorithm aims to minimize f, $x_1^{(u)}$ refers to the best point, $x_2^{(u)}$ to the second best point, and so on to finish with the worst point, $x_{n+1}^{(u)}$. If there is a tie between function values, the first value found dominates over the second.

The ***ComputeReflectionPt*** step computes the reflection point $\boldsymbol{x}_r$ as follows:

$$\boldsymbol{x}_r = \bar{\boldsymbol{x}} + \rho(\bar{\boldsymbol{x}} - \boldsymbol{x}_{n+1}) \tag{5}$$

where $\bar{\boldsymbol{x}} = \sum_{i=1}^{n} \boldsymbol{x}_i/n$ is the centroid of the n best points and $\rho = 1$ [17].

The ***BoundedTest*** performs the NM constraint procedure presented in Table 5. For some optimization problem, the variables to optimize are interdependent. For that reason, the constraint procedure takes into account this interdependency during the optimization tuning. If more values are out of bounds, the procedure takes the highest ratio of the excess value over the distance between the new value and the value to be replaced. Then, the shrinkage is performed by this ratio on all the variable of the vertex. Afterwards, the new point is evaluated.

Table 5: ***BoundedTest*** procedure

Compute the excess of the lower and upper bound limits of $\boldsymbol{x}$ $\forall\ i \in 1, \ldots, n$:
Initialization: $\boldsymbol{\delta}_{min} = 0$, $\boldsymbol{\delta}_{max} = 0$
for $i = 1 : n$
  if $\boldsymbol{x}(i) < \boldsymbol{\varphi}_{min}(i)$
$$\boldsymbol{\delta}_{min}(i) = |\boldsymbol{x}(i) - \boldsymbol{\varphi}_{min}(i)|$$
  elseif $\boldsymbol{x}(i) > \varphi_{max}(i)$
$$\boldsymbol{\delta}_{max}(i) = |\boldsymbol{x}(i) - \boldsymbol{\varphi}_{max}(i)|$$
  end
end
if $\boldsymbol{\delta}_{max} \neq 0 \parallel \boldsymbol{\delta}_{min} \neq 0$
  for $i = 1 : n$
    Compute the difference, $\boldsymbol{d}$, between $\boldsymbol{x}$ and the vertex to be replaced, $\boldsymbol{x}_{n+1}$,
$$\boldsymbol{d}(i) = |\boldsymbol{x}(i) - \boldsymbol{x}_{n+1}(i)|$$
    Compute the bound ratios of the excess value, $\boldsymbol{\delta}_{min}$ and $\boldsymbol{\delta}_{max}$, relative to the distance, $\boldsymbol{d}$
    if $\boldsymbol{d}(i) > 0$
$$\boldsymbol{r}_{lower}(i) = \boldsymbol{\delta}_{min}(i)/\boldsymbol{d}(i), \boldsymbol{r}_{upper}(i) = \boldsymbol{\delta}_{max}(i)/\boldsymbol{d}(i)$$
    else
$$\boldsymbol{r}_{lower}(i) = 0, \boldsymbol{r}_{upper}(i) = 0$$
    end
  end
    Compute the gain of the higher ratio:
$$r_{max} = max\{\boldsymbol{r}_{lower}, \boldsymbol{r}_{upper}\}$$
  for $i = 1 : n$
    Proceed to the overall shrinkage:
    if $\boldsymbol{x}(i) < \boldsymbol{x}_{n+1}(i)$
$$\boldsymbol{x}(i) = \boldsymbol{x}(i) + r_{max} \times \boldsymbol{d}(i)$$
    else
$$\boldsymbol{x}(i) = \boldsymbol{x}(i) - r_{max} \times \boldsymbol{d}(i)$$
    end
    Verify that the new computed vertex is within the bounds* $\boldsymbol{x}$ $\forall\ i \in 1, \ldots, n$
    if $\boldsymbol{x}(i) > \boldsymbol{\varphi}_{max}(i)$
$$\Delta_{bound} = |\boldsymbol{\varphi}_{max}(i) - \boldsymbol{\varphi}_{min}(i)|$$
$$\boldsymbol{x}(i) = \boldsymbol{\varphi}_{max}(i) - 0.75\Delta_{bound}$$
    elseif $\boldsymbol{x}(i) < \boldsymbol{\varphi}_{min}(i)$
$$\Delta_{bound} = |\boldsymbol{\varphi}_{max}(i) - \boldsymbol{\varphi}_{min}(i)|$$
$$\boldsymbol{x}(i) = \boldsymbol{\varphi}_{min}(i) + 0.75\Delta_{bound}$$
    end
  end
  Evaluate $f(\boldsymbol{x})$
end

**This procedure has also been added to the reflection constraint [24] for comparison.*
This procedure has been added to avoid bouncing between the bounds.

The ***Introduce*** step refers to replacing $\boldsymbol{x}_{n+1}$ by the new $\boldsymbol{x}$. After that, unless the NMStopCriteria or the ACONMStopCriterion is met, the constrained NM algorithm starts over with ***OrderingVertices*** and so on.

The ***ComputeExpansionPt*** computes the expansion point $\boldsymbol{x}_e$ as follows:

$$\boldsymbol{x}_e = (1 + \rho\chi)\bar{\boldsymbol{x}} - \rho\chi\boldsymbol{x}_{n+1} \tag{6}$$

where $\chi = 2$ [17].

The ***ComputeOutsideContractionPt*** performs an outside contraction point $\boldsymbol{x}_{oc}$ as follows:

$$\boldsymbol{x}_{ic} = (1 + \rho\gamma)\bar{\boldsymbol{x}} - \rho\gamma\boldsymbol{x}_{n+1} \tag{7}$$

where $\gamma = 1/2$ [17].

The ***ComputeInsideContractionPt*** computes an inside contraction point $\boldsymbol{x}_{ic}$ as follows:

$$\boldsymbol{x}_{ic} = (1-\gamma)\bar{\boldsymbol{x}} + \gamma \boldsymbol{x}_{n+1} \tag{8}$$

The ***PerformShrinkage*** shrinks the simplex from the best point $\boldsymbol{x}_1^{(u)}$, for $i = 2, \ldots, n+1$ as follows:

$$\boldsymbol{v}_i = \boldsymbol{x}_1 + \sigma(\boldsymbol{x}_i - \boldsymbol{x}_1) \tag{9}$$

where $\sigma = 1/2$ [17]. After the shrinkage, the constrained NM algorithm starts over with ***OrderingVertices*** unless the NMStropCriteria or the ACONMStopCriterion is met.

3. Benchmarking the proposed ACO-NM

The proposed ACO-NM is compared to various optimization algorithms, such as GA, Differential Evolution (DE), ABC, PSO, Teaching learning based optimization (TLBO), NM, NM-PSO and ACO_R under the same conditions as the results presented in [22], [44, 45]. The proposed ACO-NM is tested over well known benchmark functions with distinctive characteristics in order to demonstrate its efficiency for solving diverse real optimization problems. The benchmark function characteristics (C) are defined as unimodal (U), multimodal (MU), separable (S), non-separable (N) as well as by their dimension (D) [44]. As opposed to unimodal functions, multimodal functions have more than one local optimum. The challenge with these functions is to catch the global optimum among the several local minima areas. Non-separable functions' p-variables are more difficult to optimize as it is impossible to express them as a sum of p functions [46].

A successful minimization refers to a difference of less than 0.001 between the known optimum and the solution found by the algorithm. The *s-rate* represents the percentage of successful minimization [22] [44]. The *f-obj* value refers to the average difference between the best point found and the known global optimum for the successful minimization only. ME refers to the average value of the solutions while SD indicates the standard deviation. The *f-evals* value refers to the average number of function evaluations required to reach the best solution. The ACO-NM default parameter values as presented in §2 are used for all comparisons.

Table 6 presents the results compared to [44]. The simulation conditions are the same than [44], which means $\kappa = 500,000$ and the value under 10e-12 are assumed to be 0. The NM-StopCriteria are $\alpha = 10,000$, $\Omega = 10,000$ and ϱ=1e-12. The proposed ACO-NM presents similar performance for Branin, Matyas and Hartman-3 functions, but significantly dominates for PowerSum and Perm functions.

Table 6: Comparison of optimization algorithms over 30 runs

Function			GA [44]	PSO [44]	DE [44]	ABC [44]	TLBO [44]	The proposed ACO-NM
Branin	C: MU,S	ME	0.397887	0.39788736	0.3978874	0.3978874	0.3978874	0.39788736
	D-2	SD	0	0	0	0	0	0
Matyas	C : U,N	ME	0	0	0	0	0	0
	D-2	SD	0	0	0	0	0	0
PowerSum	C: MU,N	ME	0.010405	11.390448	0.0001425	0.0029468	0.0000743	0
	D-4	SD	0.009077	7.3558	0.000145	0.002289	0.0001105	0
Perm	C: MU,N	ME	0.302671	0.0360516	0.0240069	0.0411052	0.0006766	0.000166
	D-4	SD	0.193254	0.048927	0.046032	0.023056	0.0007452	0.000282
Hartman-3	C: MU,N	ME	-3.86278	-3.6333523	-3.862782	-3.8627821	-3.862782	-3.8627821
	D-3	SD	0	0.116937	0	0	0	0

The results of the proposed ACO-NM compared to [45] are in Table 7. The proposed combination of ACO with NM differs from the NM and PSO in [45]. Therefore, the $1,000 \times n$ algorithm iterations used as a stop criterion in [45] would lead to an unnecessary excessive number of function evaluations for the proposed ACO-NM. For fair comparison, the ACO-NM stop criterion κ is the biggest number of the *f-evals* among [45] according to the benchmark function. The NMStopCriteria are $\alpha = 1,000 \times n$, Ω is inactive and ϱ = 10e-12. Since ABC is a popular optimization algorithm and presents many advantages compared to other algorithms [10], the ABC [47] has been added to this benchmark test. A tuning procedure to set the number of bees, the abandonment limit parameter and the acceleration coefficient has been performed to achieve better results than those obtained with the ABC defaults values [47]. Moreover, because of its performance in solving complex non-linear problems [48], DE presented in [49] has also been added to this benchmark test. Its parameters have been chosen according to the parameters' study presented in [50]. Furthermore, the method to constrain NM in [24], which this paper refers to as *reflection constraint*, is inserted in the proposed ACO-NM instead of the proposed constraint procedure. This enables to fairly compare both NM constraint procedures. The *CPU-time* refers to the time needed for the algorithm to converge.

Table 7: Comparison of optimization algorithms over 100 runs

Function			NM [45]	PSO [45]	GCPSO [45]	NM-PSO [45]	ABC	DE	The proposed ACO-NM with the *reflection constraint* [24]	The proposed ACO-NM
Griewank*	C: MU,N	*s-rate* (%)	0	0	0	82	21	35	100	100
	D-10	*f-obj*	-	-	-	1.040e-11	6.6198e-04	0	0	0
	[-50,50]	ME	1.040	0.123	0.088	0.017	0.0064	0.009	0	0
	κ =500,000	*f-evals*	3,270	504,657	9,253	14,076	935,547	42,279	35,930	39,228
		CPU-time (s)**	-	-	-	-	37.539	0.82077	0.85285	1.2684
Booth	C: MU,S	*s-rate* (%)	100	100	100	100	100	100	100	100
	D-2	*f-obj*	2.622e-8	3.060e-8	9.722e-8	1.266e-9	5.6969e-07	7.6729e-22	2.2629e-25	2.3014e-25
	[-50,50]	ME	2.622e-8	3.060e-8	9.722e-8	1.266e-9	5.6969e-07	7.6729e-22	2.2629e-25	2.3014e-25
	κ =4,000	*f-evals*	145	3,848	2,128	1,065	7,438	3,940	994	887
		CPU-time (s)**	-	-	-	-	0.34085	0.04583	0.05951	0.04375
Beale	C: U,N	*s-rate* (%)	52	100	100	100	100	100	88	100
	D-2	*f-obj*	4.757e-9	5.986e-8	3.689e-8	1.607e-9	3.038e-6	2.7631e-21	2.1552e-26	1.8844e-26
	[-50,50]	ME	0.845	5.986e-8	3.689e-8	1.607e-9	3.038e-6	2.7631e-21	0.0434	1.8844e-26
	κ =5,000	*f-evals*	2,742	5,440	2,792	1,458	8,940	4,932	1,348	1,510
		CPU-time (s)**	-	-	-	-	0.41869	0.04319	0.05408	0.07618
Powell badly scaled function	C: U,N	*s-rate* (%)	74	94	100	100	100	100	100	100
	D-2	*f-obj*	8.591e-6	9.896e-6	2.668e-6	3.785e-6	9.3787e-6	8.8032e-8	9.3279e-29	8.0027e-29
	[-50,50]	ME	0.475	692,348,675.92	2.668e-6	3.785e-6	9.3787e-6	8.8032e-8	9.3279e-29	8.0027e-29
	κ =20,000	*f-evals*	631	20,144	12,375	2,971	15,687	6,830	2,177	2,144
		CPU-time (s)**	-	-	-	-	0.68528	0.06330	0.08654	0.07804
Rastrigin*	C: MU,S	*s-rate* (%)	0	30	0	60	0	29	94	95
	D-10	*f-obj*	-	1.080e-4	-	1.911e-11	-	0	2.0598e-15	2.4121e-15
	[-50,50]	ME	1,164.238	1.021	7.771	4.836	8.9372	1.0646	0.0597	0.0497
	κ =500,000	*f-evals*	2,887	509,193	11,146	12,376	771,212	24,980	202,303	208,175
		CPU-time (s)**	-	-	-	-	29.898	0.50458	5.10	7.30

*Since the global optimum is near 0 for these function, $div = 1,000$

*** The CPU-time is computed with Intel(R) Core(TM) i5-5200U CPU @ 2.20Ghz 2.19 GHz with 4.00Go RAM on Windows 10 Professional with Matlab R2015b.*

For the five benchmark functions, the *s-rate*, the *f-obj* and the ME are significantly better or equals with the proposed ACO-NM than [45]. Moreover, the proposed ACO-NM reaches better *f-obj*, ME, *f-evals* and *CPU-time* than ABC. Likewise, the proposed ACO-NM achieves significantly better or equals *s-rate*, *f-obj*, ME and *f-evals* than DE, except for Rastrigin function. For that function, DE reaches a lower *f-obj*, *f-evals* and *CPU-time*, but the proposed ACO-NM still gets appreciably higher *s-rate* and lower ME. For Griewank, Booth and Powell, the proposed ACO-NM achieves similar results than those obtained with the proposed ACO-NM with the *reflection constraint*. However, for Beale function, the proposed ACO-NM has a significantly higher *s-rate* and a lower ME. For Rastrigin function, the *s-rate* is slightly higher and ME is considerably lower with the proposed ACO-NM than those obtained with the proposed ACO-NM with the *reflection constraint*.

Table 8 presents the results compared to [22] under the same conditions than [22]; α and Ω are inactive and $\kappa = 1000 \times D$ function evaluations. The *f-evals* are lower than [22] ex-

cept for Griewank function. However, all the *s-rate* of the proposed ACO-NM are significantly higher. The proposed ACO-NM compared to the proposed ACO-NM with the *reflection constraint* achieves similar results for Ackley and Rastrigin functions, but reaches a higher *s-rate* with a lower *f-evals* for Griewank function.

Thus, the results over different benchmark functions with different characteristics confirm the efficiency of the proposed ACO-NM and its constraint procedure.

Table 8: Comparison ACO_R and ACO-NM algorithms over 100 runs

Function			ACO-NM [22]	ACO_R [22]	The proposed ACO-NM with the *reflection constraint* [24]	The proposed ACO-NM
Ackley	C : MU,N	*s-rate* (%)	79	81	100	100
	D-2 [-30,30]	*f-evals*	713	1,252	159	154
Griewank	C : MU,N	*s-rate* (%)	1	28	57	66
	D-10 [-50,50]	*f-evals*	2,102	2,680	3,993	4,890
Rastrigin	C : MU,S	*s-rate* (%)	56	63	100	100
	D-2 [-1,1]	*f-evals*	807	1 389	104	105

4. AVR system and PID structure

The purpose of an AVR system is to maintain the voltage terminal of a synchronous generator at a specific value [25]. It is composed of an amplifier, exciter, generator and sensor. Its constituents can be represented by the transfer functions as presented in Fig. 2 [25],[51]. The generator output voltage, V_t, is continually sensed to rectify its magnitude by the PID controller in order to reach the reference voltage, V_{ref} [32].

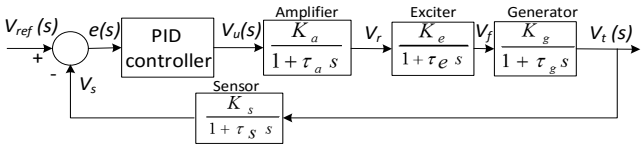

Figure 2: AVR system model with PID controller.

PID control is introduced in a system in order to improve its dynamic response. PID algorithm in time domain for parallel form is:

$$V_u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{10}$$

where V_u is the control signal; e is the error between the reference V_{ref} and the measurement of output signal V_s, and K_p, K_i and K_d are the coefficients of the proportional, integral and derivative terms respectively.

The PID tuning challenge is to find the optimal parameter set in order to minimise the rise time t_r, the settling time t_s, and the maximum overshoot M_p. PID tuning by optimization requires a cost function, which defines a compromise among the performance criteria.

5. AVR system results and discussion

5.1. Overview of previous PID controllers for AVR system

Some PID tuning algorithms are more efficient and have a more appropriate cost function for PID controller adjustments. To conduct a suitable choice of cost function and to ascertain that the parameter set of the PID controller is better than others, a survey of the PID controller tuning of the AVR system is performed according to the final closed loop response of the AVR system. This survey considers the PID adjustment works of the original AVR system [25] where $K_a = 10$, $K_e = 1$, $K_g = 1$, $K_s = 1$, $\tau_a = 0.1$, $\tau_e = 0.4$, $\tau_g = 1$ and $\tau_s = 0.01$. To maintain consistency in the comparison, the thresholds for t_s is ST = 2% with respect to the reference value and for t_r is RT = 10% to 90% of the steady-state value. For the publications where ST and RT values are unmentioned, a simulation is performed with the proposed solutions to obtain the characteristic measures. M_p value refers to the difference in percentage between the maximum peak value of the response $V_{t_{max}}$ and the reference value V_{ref}. Table 9 resumes the proposed PID solutions and their performance characteristics obtained for the AVR system; the optimization algorithms are listed in Table 2.

Table 9: Optimization results for other proposals solving PID controller tuning for the AVR system

Scenario	Ref.	Cost function $J(\boldsymbol{x})$ =	K_p	K_i	K_d	$M_p(\%)$	$t_s(s)$	$t_r(s)$
A	[26]	$ITAE \times ((1-e^{-\beta})(V_{t_{max}} + E_{ss}) + e^{-\beta}(t_s - t_r))$	0.6668	0.4825	0.2262	2.00	0.5628	0.2837
B	[27]		0.6443	0.4700	0.2423	**0.44**	1.1025	0.2800
C	[25]	$((1-e^{-\beta})(V_{t_{max}} + E_{ss}) + e^{-\beta}(t_s - t_r))$	0.6570	0.5389	0.2458	**1.16**	**0.4114**	**0.2723**
D	[28]	$((1-e^{-\lambda})(M_p + E_{ss}) + e^{-\lambda}(t_s - t_r))$	1.0184	0.2809	0.2308	12.96	4.4145	**0.2285**
E	[29]	$\sum_{k=1}^{1000} k\lvert y(k) - y_r(k)\rvert + t_s^2 + 10^8 M_p^2$	0.5069	0.3633	0.1513	**0.62**	0.6343	0.4080
F	[30]	$\frac{e^{-\beta}(t_s/max(t) + \alpha M_p)}{\alpha} + E_{ss}$	0.6728	0.4787	0.2299	1.97	0.4180	0.2795
G	[31]	$G_e \int_0^T e^2(t)dt + G_u \int_0^T u_c^2(t)dt + G_M M_P + G_s E_{ss} + G_d sup\lvert\frac{de(t)}{d(t)}\rvert$	0.948	0.256	0.232	10.07	4.8424	**0.2365**
H	[32]	$ITAE$	0.5857	0.4189	0.1772	1.95	0.5155	0.3433
I	[33]	$ITAE$	1.9522	0.4515	0.4753	28.27	2.7161	**0.1321**
J	[34]	IAE	0.9951	0.2773	0.2238	12.58	4.4394	**0.2339**
K	[35]	$ITAE + \alpha_1 \times \xi + \alpha_2 \times t_s + \alpha_3 \times t_r + \alpha_4 \times V_{t_m ax}$	1.2771	0.8471	0.4775	16.93	0.8039	**0.1438**
L	[36]	$0.5(1 + M_P)(t_s + t_r)$	0.4500	0.2900	0.1300	**0.00**	0.8169	0.4822
M	[37]	$\omega_{M_P} M_P + \omega_{t_r} t_r + \omega_{t_s}$	0.7080	0.6560	0.2820	2.60	0.8133	**0.2400**
N	[38]	$ITSE$	1.6524	0.4083	0.3654	25.01	3.0939	**0.1557**
O	[39]	$\alpha_1 \times ITAE + \alpha_2 \times t_s + \alpha_3 \times V_{t_{max}}$	0.5473	0.3556	0.1668	**0.00**	0.5864	0.3744
P	[40]	$\alpha_1 \times ISE + \alpha_2 * t_r + \alpha_3 * V_{t_{max}}$	0.6190	0.4224	0.2058	**0.59**	0.4778	0.3123

Regarding only t_s, scenario C is better. Combining t_s and M_p values as a performance measurements, scenarios C, F, H, O and P seem interesting. However, considering also the t_r value, scenario C dominates. Regardless of the optimizer used, this review highlights the performance of the cost function proposed by scenario C. Also, this cost function contains only one parameter to set. Therefore, this cost function (11) is chosen for the PID AVR tuning with the proposed ACO-NM:

$$J(\boldsymbol{x}) = (1 - e^{-\beta})(V_{t_{max}} + E_{ss}) + e^{-\beta}(t_s - t_r) \tag{11}$$

where $\boldsymbol{x}$ is $[K_p, K_i, K_d]$, β is a weighting factor, $V_{t_{max}}$ is the maximum peak value in the system response and E_{ss} is the steady-state error. Although E_{ss} is null in the final system response due

to the presence of the integrator, this term helps the optimization process to reach the desired solution faster.

5.2. PID controller of AVR system - PSO compared to the proposed ACO-NM

The proposed ACO-NM is compared to PSO (Scenario C of Table 9) under the same conditions than [25], which means the cost function J($\boldsymbol{x}$) is (11), $\kappa = 10,000$, $\boldsymbol{\varphi}_{min} = [0\ 0\ 0]$, $\boldsymbol{\varphi}_{max} = [1.5\ 1.0\ 1.0]$ and the Routh-Hurwitz criterion for closed loop stability has been added. The ACO-NM parametrization [19] is $\alpha = \Omega = 600$, $\varrho = 1e^{-5}$, $m = 15$, $\Upsilon = 0.2$, $r = 20$, $q = 5$ and $div = 100$. The evaporation rate is enforced and the $J(\boldsymbol{x})$ value is penalized on parameters leading to an unstable system response or undefined transient response characteristic values.

The proposed PID controller parameter sets are compared in Table 10 to PSO-PID [25] and to the set obtained with the SIMC method [52]. SIMC method is based on earlier ideas of Ziegler-Nichols tuning method and Internal Model Control. Some extra work for proportional gain tuning for SIMC method was necessary.

Table 10: Best PID controller over 100 runs (ST = 2%, RT = 10% to 90 %)

β	Optimizer	K_p	K_i	K_d	M_p(%)	E_{ss}	t_s (s)	t_r (s)	$J(\boldsymbol{x})$
1.0	PSO	0.6570	0.5389	0.2458	1.16	0	0.4114	0.2723	0.6906
	ACO-NM	0.67387	0.59506	0.262173	1.67	0	0.3860	0.2580	0.6898
1.5	PSO	0.6254	0.4577	0.2187	0.44	0	0.4609	0.3007	0.8160
	ACO-NM	0.63475	0.48005	0.22665	0.61	0	0.4451	0.2918	0.8159
	SIMC-PID	0.5075	0.3500	0.1575	0	0	0.6621	0.4060	-

The proposed ACO-NM achieves lower $J(\boldsymbol{x})$ values than PSO for both β values, which means a better efficiency for the proposed ACO-NM. According to the system response characteristics, ACO-NM-PID provides lower t_r and t_s values while PSO-PID and SIMC-PID present a little less overshoot M_p. Fig. 3 shows the terminal voltage step response of the AVR system and its zoom on the maximum overshoot.

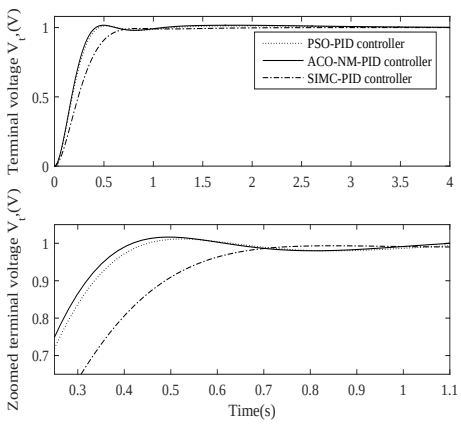

Figure 3: Terminal voltage step response of AVR system with its zoom for $\beta = 1.0$.

A second simulation scenario has to be performed to draw stronger conclusions about the proposed ACO-NM since the ST and RT values are unmentioned in [25]. To obtain exactly the

same t_s and t_r values of the AVR controlled with the PID tuned as in [25], it is necessary that ST = 2.625% and RT= 8.84% to 90% for $\beta = 1.0$ as well as ST=2.43 % and RT=8.48% to 90% for $\beta = 1.5$. As these thresholds affect the cost function value, the proposed ACO-NM is performed with these thresholds. Table 11 shows again how ACO-NM-PIDs have a lower $J(\boldsymbol{x})$ value. Moreover, the maximum overshoot M_p, the settling time t_s, and the rise time t_r of ACO-NM-PID controller system response are lower than PSO-PID controller for both β values.

Table 11: Best PID controller solutions over 100 runs

β	Optimizer	K_p	K_i	K_d	$M_p(\%)$	E_{ss}	t_s (s)	t_r (s)	$J(\boldsymbol{x})$
1.0	PSO [25]	0.6570	0.5389	0.2458	1.16	0	0.4025	0.2767	0.6857
	ACO-NM	0.66862	0.54774	0.25808	1.14	0	0.3871	0.2668	0.6836
1.5	PSO [25]	0.6254	0.4577	0.2187	0.44	0	0.4528	0.3070	0.8128
	ACO-NM	0.63041	0.4498	0.22250	0.27	0	0.4478	0.3033	0.8113

Table 12 presents the convergence of the proposed ACO-NM, the PSO [25], the proposed ACO-NM with the *reflection constraint* [24], ABC [47], DE [49] and GA [53] over 100 runs. Regarding $\beta = 1.5$, GA-PID has the lowest $J(\bar{\boldsymbol{x}})$, J_{max} and ΔE ($\Delta E = J_{max}(\boldsymbol{x}) - J_{min}(\boldsymbol{x})$) values among all algorithms. However, GA-PID overtakes the proposed ACO-NM-PID only by 0.0000051 for $J(\bar{\boldsymbol{x}})$ and the *f-evals* as well as the *CPU-time* for GA-PID are among the ones with the highest values; they are significantly higher than those obtained with the proposed ACO-NM-PID. Concerning NM constraint procedure comparison, the proposed ACO-NM-PID reaches a lower $J(\bar{\boldsymbol{x}})$, J_{max} and ΔE than the ACO-NM-PID *reflection constraint*, but the *f-evals* is a little higher.

For $\beta = 1.0$, DE-PID presents the lowest $J(\bar{\boldsymbol{x}})$, J_{max} and ΔE values, but has the highest *f-evals* and *CPU-time* among all algorithms. The proposed ACO-NM-PID reaches a slightly higher $J(\bar{\boldsymbol{x}})$ value, but *f-evals* and *CPU-time* are considerably lower. Furthermoer, the proposed ACO-NM-PID still reaches a lower $J(\bar{\boldsymbol{x}})$ value than the one obtained with the ACO-NM-PID *reflection constraint* for a lower *f-evals* and *CPU time* values. The proposed NM constraint procedure is more efficient than the *reflection constraint* because it considers the interdependency between the parameters to be adjusted. According to the AVR system and the benchmark function overall results, the proposed ACO-NM is more effective than the other compared algorithms; it reaches high quality response with a high computational performance.

Table 12: Comparison of computation efficiency over 100 runs

		J_{max}	$J_{min}(\boldsymbol{x})$	$J(\bar{\boldsymbol{x}})$	ΔE	*f-evals*	*CPU-time* (s)*
$\beta = 1.5$	proposed ACO-NM-PID	0.8116850	0.8112899	0.8113127	0.0004	4,429	341
	ACO-NM-PID *reflection constraint* [24]	0.8119710	0.8112899	0.8113139	0.0045	3,983	304
	PSO-PID [25]	0.8149300	0.8129420	0.8136034	0.0020	-	-
	ABC-PID	0.8136967	0.8115569	0.8122818	0.0021	8,096	619
	DE-PID	0.9285182	0.8112900	0.8124623	0.1172	9,842	873
	GA-PID	0.8114100	0.8112911	0.8113076	0.0001	9,726	847
$\beta = 1.0$	proposed ACO-NM-PID	0.6843690	0.6836069	0.6836410	0.0007	3,693	284
	ACO-NM-PID *reflection constraint* [24]	0.6843690	0.6836069	0.6836485	0.0007	4,566	356
	ABC-PID	0.6854456	0.6836886	0.6842643	0.0018	8,376	652
	DE-PID	0.6836070	0.6836070	0.6836070	0	9,873	766
	GA-PID	0.6836920	0.6836084	0.6836209	0.00006	9,698	712

The CPU-time is computed with Intel(R) Core(TM) i5-5200U CPU @ 2.20Ghz 2.19 GHz with 4.00Go RAM on Windows 10 Professional with Matlab R2015b.

5.3. Closed loop system analysis

The closed loop zeros and poles, the damping ratio and the natural frequencies when the AVR system is controlled with the PSO-PID [25] and the proposed ACO-NM-PID are presented in Table 13. All the closed loop poles of the system are lying to the left of the s-plane, which means the controlled AVR system is stable. The damping ratio value gives information concerning the oscillatory nature of the transient response. A larger value produces transient response with less oscillations. The damping ratios of ACO-NM-PIDs and PSO-PIDs are very similar as well as their natural frequencies. This analysis confirms the stability of ACO-NM-PIDs for the AVR system.

Table 13: Closed loop characteristics of PID controllers

	Optimizer	Closed-loop poles	Zeros	Damping ratio	Natural frequencies (rad/s)
β = 1	PSO	-1.1920 ± 0.7031i	-1.3365 ± 0.6374i	0.8611	1.3843
		-5.2186 ± 6.5303i	∅	0.6243	8.3594
		-100.6788	-100.0000	1.0000	100.6743
	ACO-NM	-1.1554 ± 0.7157i	-1.2954 ± 0.6666i	0.8501	1.3591
		-5.2382 ± 6.7949i	∅	0.6105	8.5796
		-100.7128	-100.0000	1.000	100.7128
β = 1.5	PSO	-1.3031 ± 0.3920i	-1.4298 ± 0.2201i	0.9576	1.3608
		-5.1450 ± 5.9117i	∅	0.6565	7.8371
		-100.6038	-100.0000	1.0000	100.6038
	ACO-NM	-1.2945 ± 0.3326i	-1.4167 ± 0.1211i	0.9685	1.3365
		-5.1484 ± 6.0052i	∅	0.6509	7.9100
		-100.6143	-100.0000	1.0000	100.6143

5.4. Robustness analysis

Table 14 presents the robustness analysis of the proposed ACO-NM-PIDs which is performed by varying the time constant values of the AVR components (±50% and ±25%) [35],[37]. The dynamic performances of the transient response are contained within the expected range [54] and the average of maximum deviation is comparable to [32], [37]. For the case of +50% for τ_g variation, the deviation on t_s may be perceived as high. However, ST is fixed for high performance i.e. 2%. The results in Table 15 confirm that the system is still stable and the damping ratios are acceptable considering the values in [32], [35], [38]. According to this analysis, ACO-NM-PIDs are robust.

Table 14: Deviation range and maximum deviation of the system for ACO-NM-PIDs

	Deviation range / Maximum deviation (%)			Deviation range / Maximum deviation (%)		
	β = 1.0			β = 1.5		
Parameter	$V_{t_{max}}$ (V)	t_s (s)	t_r (s)	$V_{t_{max}}$ (V)	t_s (s)	t_r (s)
τ_a	0.0666 / 6.45	1.0011 / 335.64	0.0291 / 11.57	0.0641 / 6.33	0.2640 / 120.86	0.0441 / 16.24
τ_e	0.0477 / 4.06	1.2169 / 470.45	0.1638/ 34.42	0.0527 / 4.95	0.7920 / 274.79	0.1746 / 32.74
τ_g	0.0574 / 7.27	1.8295 / 633.98	0.2288 / 45.54	0.0426 / 5.65	1.4670 / 506.89	0.2568 / 44.50
τ_s	0.0107 / 1.05	0.4643 / 122.64	0.0161 / 3.25	0.0087 / 0.86	0.4965 / 114.08	0.0175 / 3.12
Average	0.0456/4.71	1.1279 / 390.67	0.1094/23.70	0.0420 / 4.44	0.7549 / 254.18	0.1232 / 24.15

5.5. Practical PID controller

In a practical context [3], the control signal V_u should be taken into account. Therefore, a first order filter on the derivative (D) part helps to avoid problems related to real implementation of derivative on noisy signals. The PID with derivative filter expression $PID_f(s)$ is:

Table 15: ACO-NM-PIDs closed loop characteristics (+50% on τ_g value)

Closed loop poles		Damping ratio	
β = 1.0	β = 1.5	β = 1.0	β = 1.5
-1.0369 ± 0.9143i	-1.1726 ± 0.6744i	0.7501	0.8669
-5.3046 ± 4.8481i	-5.2019 ± 4.0599i	0.7382	0.7883
-100.4836	-100.4176	1.0000	1.0000

$$PID_f(s) = K_p + \frac{K_i}{s} + K_d \frac{N}{1 + N\frac{1}{s}} \tag{12}$$

where N is the filter coefficient. By incorporating the parameter N in the controller tuning ($N \in [10, \ldots, 150]$) with the proposed ACO-NM, the control signal is acceptable and the AVR system response is adequate. Table 16 shows the results compared to [55], which used a smaller search space ($0.3 < K_p < 0.6$, $0.1 < K_i < 0.5$ and $0.01 < K_d < 0.4$) and divided the filter coefficient N in two independent variables ($0.1 < N' < 2$ and $0.01 < T(s) < 0.1$) such that N=N'/T. The ACO-NM-PID_f provides a stable response faster.

Table 16: Controller solutions (ST= 2% and RT = 10% to 90 %)(β = 1)

	K_p	K_i	K_d	N	M_p(%)	E_{ss}	t_s (s)	t_r (s)	$J(\boldsymbol{x})$
ACO-NM-PID	0.67387	0.59506	0.262173	$\varnothing$	1.67	0	0.3860	0.2580	0.6898
ACO-NM-PID_f	0.6268	0.5564	0.2404	130.6039	1.76	0	0.4049	0.2662	0.6944
TLBO-PID_f [55]	0.5302	0.4001	0.1787	175.2661	1.00	0	0.5603	0.3537	-**

**The value N refers to N/T = 1.9104/0.0109 in [55].*

***The cost function in [55] is the same as this paper, but the definition of $V_{t_{max}}$ differs as a result, it is irrelevant to mention the value $J(\boldsymbol{x})$.*

Following the objective of this section, a PID controller output saturation has been included per [56]; where $V_{u_{min}} \leq V_u \leq V_{u_{max}}$ such that $-0.07 \leq V_u \leq 0.5$. As shown in Fig. 4, AVR system controlled with the previous ACO-NM-PID$_f$ has a poor performance since the control signal saturates over a long time period.

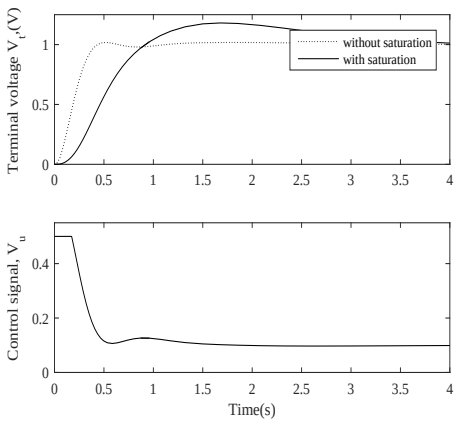

Figure 4: Terminal voltage of AVR system response for ACO-NM-PID$_f$ when control signal is saturated ($V_{u_{min}} \leq V_u \leq V_{u_{max}}$).

The incorporation of specific constraints can complicate the optimization process [3]. To manage this scenario, a 2DOF-PID controller is used [1]. This structure has two extra degrees of freedom, which enables to weight the setpoint $V_{ref}(s)$ before passing through the Proportional (P) and Derivative (D) actions [1]. The control signal is computed as follows:

$$V_u(s) = K_p\left[w_p V_{ref}(s) - V_t(s)\right] + \frac{K_i}{s}E(s) + K_d\left[w_v V_{ref}(s) - V_t(s)\right]\frac{N}{1 + N\frac{1}{s}} \tag{13}$$

where w_p and w_v are the weights on the P and D actions respectively. For this case, the decision variables refer to $\boldsymbol{x} = [K_p, K_i, K_d, N, w_p, w_v]$. Two simulation scenarios are performed to tune the 2DOF-PID structure with the proposed ACO-NM with this new fitness function (14):

$$I(\boldsymbol{x}) = (1 - e^{-\beta})(V_{t_{max}} + E_{ss}) + e^{-\beta}(t_s - t_r) + \Theta\Phi(\boldsymbol{x}) \tag{14}$$

where Θ is a weighting coefficient and $\Phi(\mathbf{x})$ is

$$\Phi(\boldsymbol{x}) = \begin{cases} \int |V_u - V_{u_{min}}| & if \quad V_u < V_{u_{min}} \\ \int V_u - V_{u_{max}} & if \quad V_u > V_{u_{max}} \\ 0 & otherwise \end{cases} \tag{15}$$

Notice that, if $\Theta = 0$, (14) equals (11).

Two parameters' value on the ACO-NM parametrization are changed, m=75 and $r = 25$ in order to achieve accurate results [19]. The search ranges for the new parameters are $w_p \in [0, \ldots, 1]$, $w_v \in [0, \ldots, 1]$ and $N \in [10, \ldots, 150]$. Table 17 and Fig. 5 show the results.

Table 17: 2DOF-PID controller solutions ($\beta = 1.0$)

Scenario	K_p	K_i	K_d	N	w_p	w_v	M_p (%)	t_s (s)	t_r (s)	Θ value	$I(\boldsymbol{x})$
a	1.2468	0.6943	0.2579	138.0926	0.8241	0.0222	0.58	0.7767	0.4624	0	0.7517
b	0.6366792	0.6219553	0.1290376	10.3772549	0.7087421	0.0100648	1.73	0.8344	0.5120	10,000*	0.7618

*$\Theta = 10,000$ for a hard penalty on violation of the control signal limit .

As (14) specifically penalizes the violation of the 2DOF-PID controller output, it is expected that the control signal remains under its saturation limits. The limitation on the control signal

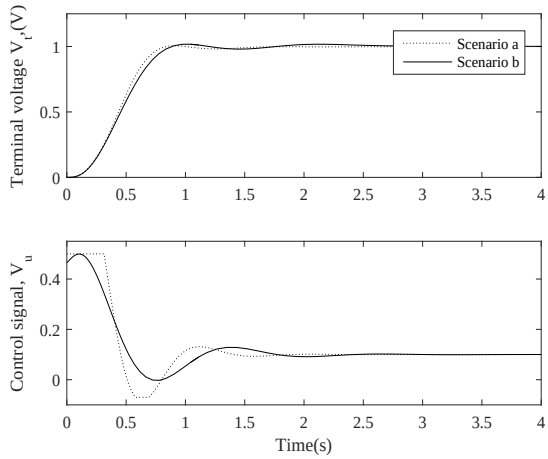

Figure 5: Terminal voltage of AVR system response with its control voltage output for 2DOF-PID for two scenarios (a: $\Theta = 0$; b: $\Theta = 10,000$).

slows down the t_s and t_r. Despite this, the tuned controller still provides satisfactory closed loop response [54].

6. Real tests with 2DOF-PIDs tuned with the proposed ACO-NM

6.1. Experimental set-up

The performance and the robustness of the controllers tuned by the proposed ACO-NM is confirmed experimentally. The experimental validation is performed with the $G26/EV$ process simulator from *Elettronica Veneta S.P.A.* [57]. This simulator is composed of different blocks. Each block develops a separate linear or non-linear function, which can be connected for the layout of the control configuration. Terminals at the input and output of each functional block enable qualitative and quantitative measurements of the variables in the control chain. The $G26/EV$ process trainer is used for educational purposes as well as for testing control algorithms under quasi-real conditions.

The process is configured using the four following blocks :

$$G(s) = \frac{V_u(s)}{V_s(s)} = \left[\frac{9.8487}{1+0.09s}\right]\left[\frac{1}{1+0.7395s}\right]\left[\frac{1}{1+1.0015s}\right]\left[\frac{1}{1+0.001s}\right] \tag{16}$$

Due to prototype physical limitations, the real process implemented (16) differs from the nominal AVR process stated in §5 but, the system is still in its parameter range [25]. Then, the experiments test the controller performance as well as its robustness. A computer with a data acquisition system is connected to the process via A/D and D/A converters as shown in Fig. 6. The computer runs NI-Labview program which implements a digital 2DOF-PID with a 5 ms sampling period.

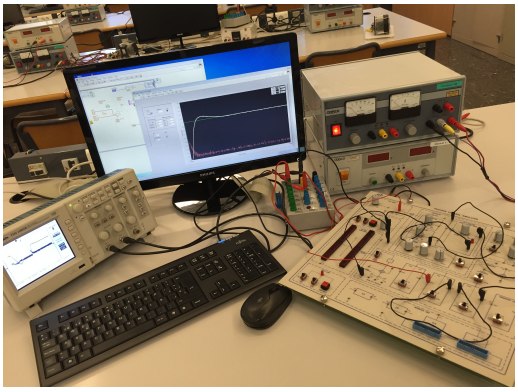

Figure 6: Experimental set-up: Closed loop implemented with a digital controller programmed in NI-LabVIEW and G26/EV process trainer simulator.

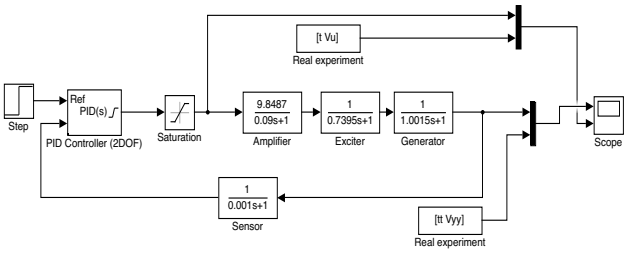

Figure 7: Simulink diagram for 2DOF-PID simulations.

6.2. *Experimental results*

The two 2DOF-PIDs of Table 17, tuned for the nominal AVR system, are implemented. As explained in previous sections, the tuning procedure was performed off-line and the resulting parameters were set in the real controller. Since the controller parameters have been tuned using the nominal AVR system values [25], a previous closed loop simulation using the Matlab/Simulink model presented in Fig. 7 has been performed to verify the closed loop stability.

After closed loop stability verification, the real control experiments are performed. The experimental results obtained are presented in Fig. 8 and 9. Even though the model used by the tuning method differs from its real process as well as the form of PID used by the method (continuous) from its implemented version (discrete), the experimental 2DOF-PIDs provide satisfactory control performance. Furthermore, the 2DOF-PIDs robustness is confirmed under real control tests and the previous closed loop performances obtained with nominal model shown in Fig. 5 are maintained. The noises and disturbances observed are inherent to real experiments.

7. Conclusion

This paper presents an optimal gain tuning method for controller parameters for the original AVR system using a new constrained ACO-NM algorithm. Its performances have been

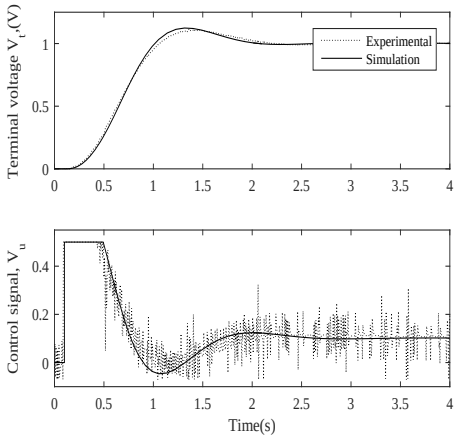

Figure 8: Terminal voltage of AVR system with its control voltage output for 2DOF-PID controller of Table 17 (Scenario-a) : Experimentation vs Simulation.

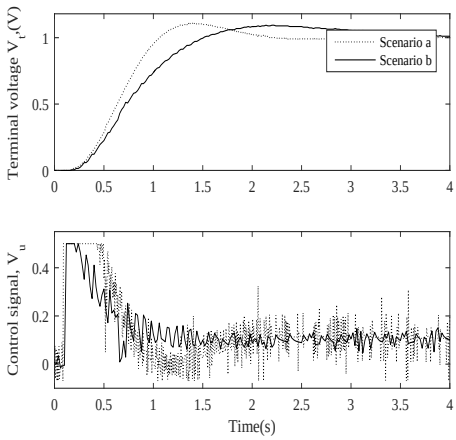

Figure 9: Real terminal voltage of AVR system with its control voltage output for 2DOF-PID controllers of Table 17 for two fitness functions.

confirmed by: *i*) achieving better or equivalent PID solutions according to the overall transient response than other AVR tuning propositions, *ii*) having higher computational efficiency, *iii*) analyzing its robustness, *iv*) validating the 2DOF-PIDs performance experimentally, and *v*) comparing simulation results with already published algorithms over benchmark functions with different characteristics. The proposed ACO-NM is promising because unlike many other algorithms used in this field, it has demonstrated that it effectively manages constraints and system requirements and works with both a realistic system and benchmark functions with different characteristics. Therefore, the proposed ACO-NM has the elements required to both be applied to other complex control structures and to account for a greater number of requirements. This paper demonstrated that taking into account the interdependency of the PID controller parameters in the NM con-

straint procedure is effective. This improvement to the number of the cost function evaluations should prove beneficial when using this method for online optimization problems. Furthermore, real control tests have shown the robustness and performance of the resulting tuned PID. For future work, robustness tests should be included during the controller tuning procedure, and the proposed ACO-NM will be applied to tune the controllers for the swing-up and stabilization for a pendulum-cart system.

8. Bibliography

[1] K. Åström, T. Hägglund, *Advanced PID Control*, ISA, Research Triangle Park, 2005.

[2] F. Padula, A. Visioli, *Optimal tuning rules for proportional-integral-derivative and fractional-order proportional-integral-derivative controllers for integral and unstable processes*, Control Theory and Applications, 6(6), 2012, pp. 776–786.

[3] G.Reynoso-Meza, J.Sanchis, X. Blasco, M. Martínez, *Evolutionary Algorithms for PID controller tuning: Current trends and perspectives*, RIAI Revista Iberoamericana de Automática e Informática Industrial, ISSN: 16977912, 10(3), 2013, pp. 251–268.

[4] D. A. Bristow, M. Tharayil, A. G. Alleyne, *A survey of iterative learning control*, IEEE Control Systems, 26(3), 2006, pp. 96–114.

[5] G. H. Nguyen, J. H. Shin, W. H. Kim, *Autotuning Controller for Motion Control System Based on Intelligent Neural Network and Relay Feedback Approach*, IEEE/ASME Transactions on Mechatronics, 20(3), 2015, pp. 1138–1148.

[6] A. Sento and Y. Kitjaidure, *Neural network controller based on PID using an extended Kalman filter algorithm for multi-variable non-linear control system*, International Conference on Advanced Computational Intelligence, 2016, pp. 302–309.

[7] P. An-hua, *GRA-based approach to PID parameter tuning for closed-loop servo systems*, Chinese Control and Decision Conference, 2013, pp. 1628–1633.

[8] R. Jensi, G.W. Jiji, *An enhanced particle swarm optimization with levy flight for global optimization*, Applied Soft Computing, 43, 2016, pp. 248–261.

[9] A. Mariajayaprakash, T. Senthilvelan, R. Gnanadass, *Optimization of process parameters through fuzzy logic and genetic algorithm - A case study in a process industry*, Applied Soft Computing, 30, 2016, pp. 94–103.

[10] X. Song, Q. Yan, M. Zhao, *An adaptive artificial bee colony algorithm based on objective function value information*, Applied Soft Computing, 55, 2017, pp. 384–401.

[11] K.Y. Lee, M.A. El-Sharkawi, *Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems*, Wiley-IEEE Press, 2008.

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

[13] S.-Z. Zhao, M.W. Iruthayarajan, S. Baskar, P. Suganthan, *Multiobjective robust PID controller tuning using two lbests multi-objective particle swarm optimization*, Information Sciences, 2011, pp. 3323–3335.

[14] P. Puri, S. Ghosh, *A hybrid optimization approach for PI controller tuning based on gain and phase margin specifications*, Swarm and Evolutionary Computation, 8, 2013, pp. 69–78.

[15] M.J.Blondin, P. Sicard, *Combined ACO algorithm - Nelder-Mead simplex search for controller and anti-windup tuning for a motion system with flexible transmission*, IEEE, IECON, 2013, pp. 3364–3369.

[16] M. Dorigo, T. Stützle, *Ant Colony Optimization*, The MIT Press, Cambridge, 2004.

[17] J. C. Lagarias J. A. Reeds M. H. Wright, P. E. Wright, *Convergence properties of the Nelder-Mead simplex method in low dimension*, SIAM J. Optimization Society for Industrial and Applied Mathematics, 9(1), 1998, pp. 112–147.

[18] M. J.Blondin, P. Sicard, *Statistical convergence analysis of ACO-NM for PID controller tuning*, International Conference on Industrial Technology, 2015, pp. 487-492.

[19] M. J.Blondin, P. Sicard, J. Sanchis, *Convergence analysis of hybrid ACO/Nelder-Mead tuning method for PID controller structures with anti-windup*, SIAM Conference on Control and its Application, 2015, pp. 251-258.

[20] M. J.Blondin, P. Sicard, *A Hybrid ACO and Nelder-Mead constrained Algorithm for Controller and Anti-Windup Tuning* , Conference Power Electronics and Applications, 2014, pp. 1–10.

[21] M.A. Luersen, R. Le Riche, F. Guyon, *A constrained, globalized, and bounded Nelder-Mead method for engineering optimization*, Structural and Multidisciplinary Optimization, 27(1), 2004, pp.43-54.

[22] N. Arun, V. Ravi, *An Ant Colony Optimization and Nelder-Mead simplex search hybrid algorithm for unconstrained optimization*, International Workshop on Machine Intelligence Research, 2009, pp. 25–31.

[23] H. Dreo,P. Siarry, *Hybrid continuous interacting Ant Colony aimed at enhanced global optimization*, Algorithmic Operations Research, 2(1), 2007, pp.52–64.

[24] S. Nigib, N. Arun, V. Ravi, *An ant colony optimisation and Nelder-Mead simplex hybrid algorithm for training neural networks: an application to bankruptcy prediction in banks*, International Journal of Information and Decision Sciences, 5(2), 2013, pp. 188–203.
[25] Z.-L. Gaing, *A particle swarm optimization approach for optimum design of PID controller in AVR system*, IEEE Trans. Energy Convers, 19, 2004, pp. 384-391.
[26] O. Bendjeghaba, I.B. Saida, *Bat Algorithm for Optimal Tuning of PID Controller in an AVR System*, International Conference on Control Engineering and Information Technology, 2014, pp. 158–170.
[27] C.-C. Wong, S.-A. Li, H.-Y. Wang, *Optimal PID controller design for AVR system*, Tamkang Journal of Science and Engineering, 12(3), 2009, pp. 259–270.
[28] M. Kashki, Y.L. Abdel-Magid, M.A. Abido, *A Reinforcement Learning Automata Optimization Approach for Optimum Tuning of PID Controller in AVR System*, D.-S. Huang et al. (Eds.), 5227, 2008, pp. 684-692.
[29] H. Kang, M.W. Kwon, H.G. Bae, *PID Coefficient Designs for the Automatic Voltage Regulator Using a New Third Order Particle Swarm Optimization*, International Conference on Electronics and Information Engineering, 2010, pp. 179–183.
[30] D.H. Kim, *Hybrid GABF based intelligent PID controller tuning for AVR system*, Applied Soft Computing, 11, 2011, pp. 11–22.
[31] S.M.A. Mohammadi, A.A. Gharaveisi, M. Mashinchi, S.M.R. Rafiei, *New Evolutionary Methods for Optimal Design of PID Controllers for AVR System*, IEEE Bucharest Power Tech Conference, 2009, pp. 1–8.
[32] S. Panda, B.K. Sahu, P.K. Mohanty, *Design and performance analysis of PID controller for an automatic voltage regulator system using simplified particle swarm optimization*, Journal of the Franklin Institute, 349, 2012, pp. 2609–2625.
[33] S. Priyambada, P.K. Mohanty, B.K. Sahu, *Automatic voltage regulator using TLBO algorithm optimized PID controller*, Industrial and Information Systems, 2014, pp. 1–6.
[34] M. Rostami, G. Heydari, A. A. Gharaveisi, S. M. R. Rafie, *Optimal Design of PID Controller Using Bacterial Foraging Algorithm for AVR System*, Congress on Fuzzy and Intelligent System, 2008.
[35] B.K. Sahu, P.K. Mohanty, S.Panda, N. Mishra, *Robust analysis and design of PID controlled AVR system using Pattern Search algorithm*, IEEE International Conference on Power Electronics, Drives and Energy Systems, 2012, pp. 1–6.
[36] G. Shabib , M. Gayed, A.M. Rashwan, *Optimal Tuning of PID Controller for AVR System using Modified Particle Swarm Optimization*, International Middle East Power Systems conference, 2010, pp. 305–310.
[37] M. A. Sahib, *A New Multi-Objective Performance Criterion used in PD Tuning Optimization Algorithms*, Journal of Advanced Research, 7(1), 2015, pp. 125-134.
[38] H. Godze, M.C. Taplamacioglu, *Comparative performance analysis of artificial bee colony algorithm for automatic voltage regulator (AVR) system*, Journal of the Franklin Institute, 348, 2011, pp. 1927–1946.
[39] B.K. Sahu, P.K. Mohanty, S. Panda, S.K. Kar, N Mishra, *Design and comparative performance analysis of PID controlled automatic voltage regulator tuned by many optimizing liaisons*, International Conference Advances in Power Conversion and Energy Technologies, 2012, pp. 1–6.
[40] K. Mohanty, K. Sahu, S. Panda, *Tuning and Assessment of Proportional Integral Derivative Controller for an Automatic Voltage Regulator System Employing Local Unimodal Sampling Algorithm*, Electric Power Components and Systems, 42(9), 2014, pp. 959–969.
[41] N. K. Yegireddy, S. Panda, *Design and performance analysis of PID controller for an AVR system using multi-objective non-dominated shorting genetic algorithm-II*, International Conference on Smart Electric Grid, 2014, pp. 1–7.
[42] M. Hanif, V. Khadkikar, X. Weidong J.L. Kirtley, *Two Degrees of Freedom Active Damping Technique for LCL Filter-Based Grid Connected PV Systems*, IEEE Trans. Ind. Electron, 61(6), 2014, pp. 2795–2803.
[43] I. Chiha N. Liouane, P. Borne, *Tuning PID controller using multiobjective Ant Colony Optimization*, Appl. Comp Intell. and Soft Computing, 2012, pp. 1–7.
[44] R. Venkata Rao, V.Patel, *Comparative performance of an elitist teaching-learning-based optimization algorithm for solving unconstrained optimization problems*, International Journal of Industrial Engineering Computations, 4, 2013, pp. 29–50.
[45] S.-K. S. Fan, E. Zahara, *A hybrid simplex search and particle swarm optimization for unconstrained optimization*, European Journal of Operations Research, 181(2), 2007, pp. 527–548.
[46] D. Karaboga, B. Akay, *A comparative study of Artificial Bee Colony algorithm*, Applied Mathematics and Computation, 214, 2009, pp. 108–132.
[47] S. M. K. Heris, *Implementation of Artificial Bee Colony in MATLAB*, Yarpiz, Project Code: YPEA114, 2015.
[48] A.R. Yildiz, *Hybrid Taguchi-differential evolution algorithm for optimization of multi-pass turning operations*, Applied Soft Computing, 13(3), 2013, pp. 1433-1439.
[49] M.E. H. Pedersen, *SwarmOps - Heuristic optimization for Matlab*, Hvass Laboratories, Copyright 2003-2010.
[50] M.E. H. Pedersen, *Good Parameters for Differential Evolution*, Technical report, Hvass Computer Science Labo-

ratories, 2010.
[51] H. Yoshida, Y. Fukuyama, S. Takayama, Y. Nakanishi, *A particle swarm optimization for reactive power and voltage control considering voltage security assessment*, IEEE Trans. Power Syst., 4, 2000, pp. 1232–1239.
[52] S. Skogestad, C. Grimholt, *The SIMC Method for Smooth PID Controller Tuning*, PID Control in the Third Millennium Advances in Industrial Control, Ed. Springer, 2012, pp. 147–175.
[53] F. X. Blasco Ferragud, *Control predictivo basado en modelos mediante tcnica de optimizacin heurstica. Aplicacin a procesos no lineales y multivariables*, PhD Tesis 1999 (in Spanish). Editorial UPV. ISBN 84-699-5429-6.
[54] W.-M. Lin, C.-L. Lee, *Discussion on parameters setting and loss of exciter time constant compensation in excitation system*, International Conference on Power System Technology, 2006, pp. 1–6.
[55] S.Chatterjee, V. Mukherjee, *PID controller for automatic voltage regulator using teaching-learning based optimization technique*, International Journal of Electrical Power & Energy Systems, 77, 2016, pp. 418–429.
[56] N. Madinehi, K. Shaloudegi, M. Abedi, H.A. Abyaneh, *Optimum design of PID controller in AVR system using intelligent methods*, IEEE Trondheim PowerTech, 2011, pp. 1–6.
[57] Elettronica Veneta S.P.A., *Process Simulator, Mod G26/EV*, Product Manual.