IDENTIFICATION AND CONTROL USING GEA

Adrian Saldanha, Milan Hofreiter

Czech Technical University in Prague, Faculty of Mechanical Engineering

Abstract: This article presents the Guiding Evolutionary Algorithm (GEA) for optimization and its application in parametric identification and control of several processes. The key advantage of using this optimization approach is that it can be used for identification of linear as well as non-linear systems. The algorithm has previously been applied successfully to optimizing a number of benchmark problems and the results were shown to be quite promising. This article applies the GEA optimization approach to the problem of identification and control of a dynamic system. The approach is verified experimentally on a physical setup and the feasibility of this approach is studied and evaluated for practical implementation.

Keywords: optimization, identification, control

1 Introduction

The Guiding Evolutionary Algorithm (GEA) was proposed by Cao et al in [1]. The algorithm was introduced to exploit some of the advantages of already existing optimization such as the Bat Algorithm (BA) [2], the Particle Swarm Optimization (PSO) [3] and the Genetic Algorithm (GA) [4]. Like the BA and GA, the GEA is also a meta-heuristic algorithm which means that the optimization is problem independent and thus can be used in a wide range of applications. The GEA bears strong resemblance with the BA and the PSO in its particle-based movement, but unlike the two, it encompasses an added mutation operator which, in essence helps avoiding the solution getting stuck in a local optimum.

The main purpose of this article is to introduce the optimization method called the 'Guiding Evolutionary Algorithm' and secondly to apply the concept to a control problem involving system identification and control of dynamic systems. The results of the identification and control are finally applied to an experimental setup and the results are drawn.

The following sections are organized as follows: Section (2) introduces the concept of optimization and explains the Guiding Evolutionary Algorithm (GEA). Section (3) explains the procedure of system identification of an unknown system, following which, in Section (4) the procedure for tuning a PID controller to control the previously identified model is described. Section (5) consists of an experimental setup and the concepts described in the previous sections are applied to a real system. Finally, section (6) presents the conclusion.

2 Optimization

As part of this article, the GEA algorithm is investigated as a potential method for solving non-linear optimization algorithms. Like most nature-based optimization algorithms, the GEA consists of a number of 'particles' or 'solutions' randomized over a higher-dimensional domain, each solution with an associated cost-function. Added to this, the algorithm, consists of three main operators namely, '*Crossover'*, *'Mutation'*, and *'Selection'*. The *crossover* operator facilitates adequate mixing within the solution space. With each generation or iteration, every solution moves slowly towards the best solutions, i.e. the particle with the optimal cost function. The *mutation* operator is necessary in order to avoid the particles settling at a local optimum solution. It provides the essential exploratory framework for ***diversification*** of the solutions. Finally, the *selection* operator ensures that the

particle with the optimum cost function is selected as the new best. With the three operations in place, with every new generation, the solution slowly converges towards the best solution, which, may or may not be the global optimum. Note that heuristic algorithms in general, do not guarantee arrival at the global optimum but are highly efficient in finding the optimum result relatively quickly. Thus, these serve as an efficient way of searching through a given space to arrive at an acceptable solution in least time.

2.1 Guiding Evolutionary Algorithm (GEA)

The pseudocode for the GEA algorithm is shown in figure (1). The Guiding Evolutionary Algorithm, while similar to the BA and PSO, is essentially simpler to use due to the fewer number of parameters required for tuning the optimization, and unlike the two, it consists of an additional mutation operator which helps to avoid the solution settling at a local optimum.

GEA Algorithm

1. Objective function ,
2. Initialize locations , define parameters c, M, L

   Calculate the cost functions at the initialized positions
3. Select the best individual
4. While (t<max number of iterations)

   Update p;

   For each individual:

   Make crossover to generate a new individual
   If (rand <p)
   Make mutation for
   End if
   If (rand <p)
   Make local search for
   End if
   If (f($\mathbf{x}_i$) < f($\mathbf{x}^*$))
   Accept new solutions
   End if
   Find current best $\mathbf{x}^*$
   End while
5. Best Solution:

Figure 1 - Pseudocode - GEA Algorithm

1. Crossover: The crossover for GEA is given by:

   $$x_i^t = x_i^{t-1} + x_*^{t-1} - x_i^{t-1} * \beta \quad (1)$$

   Where,

   β: Step length of position increment, uniformly distributed r.v

   The step length β defines the rate of convergence and is generally between 1 and 2. A higher value represents faster convergence.

2. Mutation: The mutation provides the required exploratory mechanics for optimization. It can be given by the equation:

$$x_i^t = x_i^t + \epsilon M \quad (2)$$

Where,
ϵ: Uniform r.v [-1, 1]
M: Mutation vector,
$M_j = (\max(x_{ij}^t - \text{a}, b - x_{ij}^t)$; [a, b] = range of j[th] dimension

According to GEA, the probability of mutation is given by the following:

$$p = c * \ln \frac{T_{max}}{T_{max} - t} \quad (3)$$

Where,
T_{max}: max number of generations
t: current generation
c: 0.2 (constant)

The above equation shows that the probability of mutation increases as the generations pass, thus it helps ensuring that the solution does not settle around the local maxima / minima.

3. Local Search: As before, the local search provides the necessary exploitative dynamics and can be given as follows:

$$x_i^t = x_*^{t-1} + \epsilon L \quad (4)$$

Where,
L: Local search vector
$L_j = 0.1 \ b - a$; [a, b] = range of j[th] dimension

Similar to the mutation vector, the local search functions similar to the mutation except that it serves to find a solution around the current best unlike mutation, which helps to find a new solution around the unsearched territory.

Again, the probability of local search is given by p defined by equation (3).

2.2 Parameterization

Before simulating the above functions, the appropriate parameters must be set correctly to ensure that the algorithm works in an efficient manner. One of the advantages of the GEA lies in its inherent simplicity in that it consists of only a few parameters which can be easily tuned to ensure quick convergence. In this case, the main parameters for tuning are β and c which are taken directly from literature [1] as:

Parameter	Value
c	0.97
β	[0, 2]

The value of β determines how quickly or slowly the solution moves towards the best solution. A higher value of this parameter indicates quicker convergence, but it can also result in the final solution moving about the optimum value, whereas a lower value indicates a slower convergence to the optimum. For the following simulations, the value is taken as 2.

2.3 Results

The above algorithm was evaluated in the article [5] and it was shown that the GEA performs considerable better than some of the other existing algorithms namely, BA and PSO in terms of convergence rate and its tendency to reach the global optimum solution. Furthermore, it was shown that to achieve reaching the global optimum, it was necessary to apply the modified mutation and local search operations to equations (2) and (4) as:

$$x_i^t = x_i^t + \epsilon M_j . * (rand \ 1, j \ < p) \quad (5)$$

And,

$$x_i^t = x_*^{t-1} + \epsilon L.*\,(rand\ \ 1, j\ \ < p) \tag{6}$$

The above modifications improve the exploratory tendencies of multimodal functions, by performing a probabilistic mutation and local search operation to the individual dimensions of each solution rather than to the solution as a whole. This helps avoiding the solution jumping over the optimum value, thereby enhancing the convergence characteristics.

3 Relay-based Feedback Identification

Following the above-described optimization approach, the next step is to apply this same algorithm in parametric identification of a dynamic system. The goal of the system identification task is to perform *black box identification* of an unknown process. The advantage of black box identification is the fact that this approach necessitates no knowledge about the physical system, but instead, uses the experimental data which includes inputs and outputs and a certain defining factor in terms of the cost function to identify the system to a reasonable accuracy. In this case, the ITAE (*Integral Time Absolute Error)* Method is used as a criterion for the cost function. With system identification, the end goal is to find an accurate model of the process so as to control it correctly.

3.1 System Schematic

The identification schematic can be seen in the below figure (2).

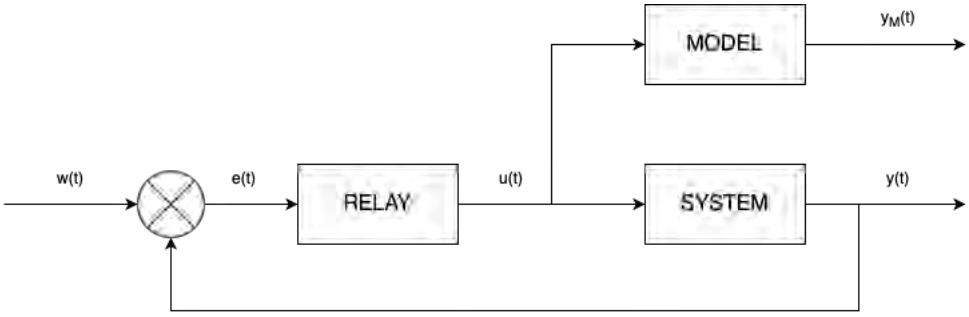

Figure 2 - Experimental Block Diagram Using Simulink

The above figure shows the process *or system* that is to be identified and then controlled. The input *u* is given by the output of the on-off relay which is used to generate sustained oscillations within the closed-loop system as proposed by Astrom and Hagglund [6]. Once the closed-loop system is running automatically, the output from the controller 'u' is fed as input to the model $G_M(s)$ which, is considered to be a Second Order Plus Time Delay (SOPTD) model. The reason for using an SOPTD model is that this model can represent almost any linear system. As explained by Ramakrishnan and Chidambaram, the SOPTD model can incorporate various processes such as under-damped and higher order processes in which case, an FOPTD model is not sufficient [7]. Furthermore, SOPTD models can also be used for unstable processes in which case, an FOPTD model is not sufficient.

Thus, by knowing the input 'u', the outputs 'y' and 'y_M', we can proceed with identification of the process.

3.2 Problem Statement

First, the terms related to the problem are defined as follows:

1. Model: The SOPTD model describe above can be represented as follows:

$$G_M\ s\ = \frac{K}{a_2 s^2 + a_1 s + 1} e^{-sT_d} \tag{7}$$

Where, 'K' = Process gain, 'θ' = Time Delay, 'a_1' and 'a_2' are dynamic constants of the transfer function.

2. Parameters: Using the SOPTD model from equation (7), the goal is to identify the parameters as follows:

$$\boldsymbol{x} = \left[K, a_1, a_2, T_d\right]^T \tag{8}$$

3. Cost function: The criterion for optimization can be given by the cost function J which can be given by:

$$J = \int_0^{T_k} \left[y_m(t) - y(t)\right]^2 dt \tag{9}$$

Where,

$y_m =$ Model output

$y_t =$ Process output

$T_k =$ Simulation time

The above cost function uses the ITAE criterion as our optimization constraint for minimizing the error over a pre-set simulation time.

4. Constraints: The constraints restrict the upper and lower limits of each parameter

$$K, a_1, a_2, T_d > 0 \tag{10}$$

Thus, the problem statement can be reduced to the following:
Estimate parameters $\boldsymbol{x}$ *of process model* G_M *by minimizing the cost function* $\boldsymbol{J}$ *subject to the constraints* $K, a_1, a_2, T_d > 0$.

4 PID Control

With the identification procedure described in the previous section, the next step is controlling the actual process using the identified model of the system.

4.1 Closed-Loop System schematic

In this article, the controller in use is a PID controller. The block diagram of the controller can be shown in the below figure.

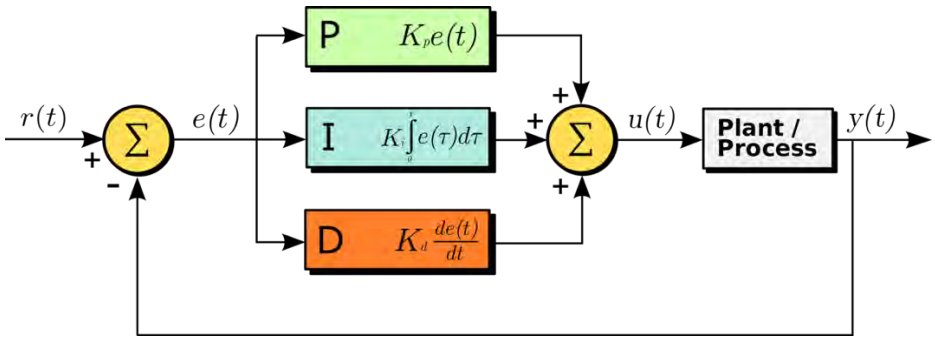

Figure 3 - PID Controller Block Diagram [8]

The equation for the controller in Laplace domain can be given by:

$$R(s) = r_P + \frac{r_I}{s} + r_D s \tag{11}$$

Where,

$r_P =$ Proportional Gain – P-term (K_P)

$r_I =$ Integral Gain – I-term (K_I)

$r_D =$ Derivative Gain – D-term (K_D)

It must be noted that in the above figure (3), the real unknown process is to be controlled using only the

identified model with no more information about the system.

4.2 Tuning Methods

To tune the controller is to find suitable values of the proportional, integral and derivative gain so as to achieve the desired control objectives. In this article, the PMC method [9] for tuning the controller is used.

Before applying the tuning procedure, the SOPTD Model $G_M\ s\ = K.\frac{e^{-sT_d}}{a_2s^2+a_1s+1}$ obtained in section (3) is reduced to one of the below forms:

$$G\ s\ = \frac{(K\omega_0^2)e^{-sT_d}}{s^2+2\zeta\omega_0 s+\omega_0^2} \text{ for oscillatory processes} \quad (12)$$

Where,
$\omega_0 =$ oscillation frequency
$\zeta =$ damping ratio

or

$$G\ s\ = \frac{Ke^{-sT_d}}{T_1s+1\ \ T_2s+1} \text{ for non-oscillatory processes} \quad (13)$$

Where,
$T_1, T_2 =$ Time constants of modelled system

The relations for tuning the controller are given by the below tables

a) Non-oscillatory Processes ($\zeta \geq 1$)

Table 1 - Tuning Equations (Non-Oscillatory Process)

No.	Parameter	PMC	
		$T_d > 0$	$T_d = 0$
1	r_P	$r_p = a_1.r_i$	$r_p = a_1 r_i$
2	r_I	$r_i = \frac{\pi - 2\gamma}{2.\ K\ T_d}$ or $r_I = \frac{\pi}{2T_d.m_A.K}$	$r_i = \frac{1}{K\tau_c}$
3	r_D	$r_d = a_2.r_i$	$r_d = a_2 r_i$

In the above equations, the parameter τ_C is the closed loop time constant and can be estimated using a general rule:

$$\tau_{dom} > \tau_c > T_d \quad (14)$$

Where,
$\tau_{dom} =$ dominant time constant of the process

As a thumb rule, we shall estimate the value of τ_c as: $\tau_C = \frac{\tau_{dom}}{3}$.

For PMC Tuning, the parameters γ and m_A refer to the phase margin and gain margin respectively. These values are generally in the range:

$$\frac{\pi}{6} < \gamma < \frac{\pi}{3}; 2 < m_a < 5 \quad (15)$$

It must be noted that the PMC tuning method is suitable for an SOPTD Model only.

b) Oscillatory Processes ($\zeta < 1$)

Table 2 - Tuning Equations (Oscillatory Process)

No.	Parameter	PMC	
		$T_d > 0$	$T_d = 0$
1	r_P	$r_p = a_1.r_i$	$r_p = a_1 r_i$
2	r_I	$r_i = \frac{\pi - 2\gamma}{2.\ K\ T_d}$ or $r_I = \frac{\pi}{2T_d.m_A.K}$	$r_i = \frac{1}{K\tau_c}$
3	r_D	$r_d = a_2.r_i$	$r_d = a_2 r_i$

Using the identified model of the system and the formulas listed in the above table, the tuning of the actual process is carried out.

5 Identification and Control on Physical System

To verify the theory and algorithms presented in the previous sections, we apply the same concepts of identification and PID control on a real system from the Automatic Control Laboratory.

5.1 Experimental Setup

The system which we will be using is a combination of two chambers arranged vertically in a tube with a system of interconnecting valves. The schematic of the setup can be seen in the below figures.

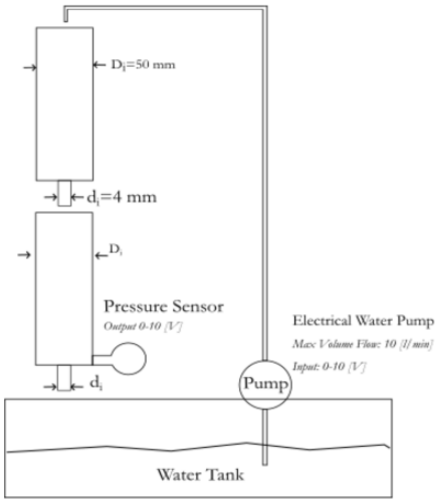

Figure 4 - Two tanks - Functional Diagram

The experiment is a combination of two hollow chambers arranged vertically in a tube with a system of interconnecting valves. Pump C1 supplies water to the upper chamber, thus, delivering a pressure head which causes the accumulated water in the upper chamber to trickle down into the lower chamber and eventually back into the tank. A pressure differential sensor is used to map the height of the water in the lower column. Since the upper and lower tanks are connected to each other in series, the system is, by default a second-order system.
To perform the experiment, we need to supply an input signal 0-10V to the pump via the MATLAB / SIMULINK software from the PC which is then transmitted to the pump which in turn pumps up water to the upper chamber A1. From the bottom hole of the upper chamber, water trickles down to the lower tank B1. The system output is measured in terms of the height of water column in the lower chamber B1. This output is measured by a pressure sensor which converts the pressure head to an equivalent height of water column. Before the experiment is performed, it is necessary to test the system for its static characteristics so that we supply the inputs and obtain the readings at around the operating point.

5.2 Simulink Schematic Setup

Similar to the Simulink setup in section (3), the experimental Simulink Schema of the physical system is arranged likewise, except that in this case, the input is not fed directly to the theoretical model (see figure (5)), but instead is done separately once the readings have been collected. Therefore, there are two separate schematics, one for the physical system and the other, the *Identification Schematic* similar to the one shown in figure (2)..

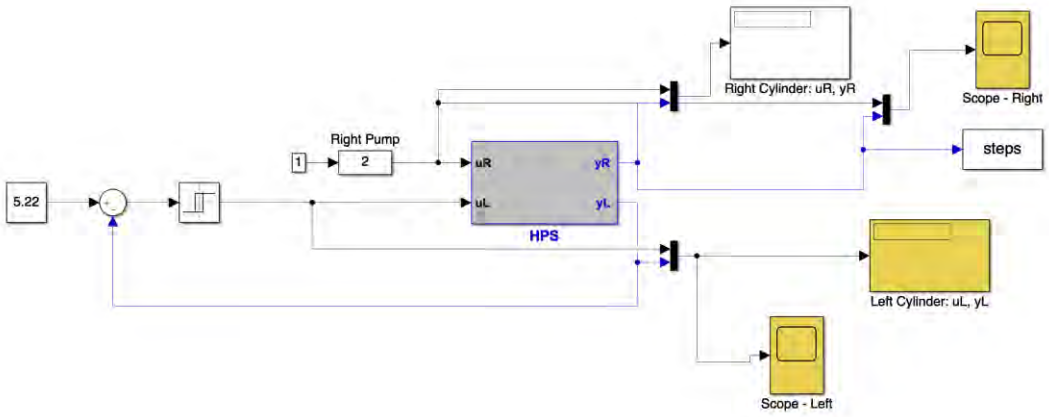

Figure 5 - Simulink Schematic - Physical system

As can be seen from the schematic in Figure (5), the Simulnk model is supplied with an input from the relay controller which in turn causes the system to auto-oscillate. From this setup, the input to the system uL and the output yL are recorded from the practical setup.

Once the values are collected, these are now fed to the simulation setup (*Identification Schematic)* which operates within the MATLAB / Simulink environment only. The advantage of this is that using just one reading from the physical setup, we can simulate the theoretical model a number of times until we obtain the model which nearly describes the unknown process. For optimization, the same input $u1$ which was supplied to the real process is fed to the theoretical SOPTD model to obtain its output y_m. By comparing the model output y_m with the real output y_1, the previously described method using the ITAE criterion can be applied.

5.3 Identification Process Model

Using the above-described procedure and the one described in section (3), the identified parameters **x** is given by:

$$\boldsymbol{x} = \ 1.1179, 1.295, 35.17, 4.43 \ ^{T} \tag{16}$$

or

$$G_M \ s \ = \frac{1.1179}{1.295s^2 + 35.17s + 1} e^{-4.43s}$$

5.4 PID Control

Using the model obtained in the previous section, the experimental process is controlled using the PID control method detailed in section (4.2). The tuning parameters for control of the identified process from equation (16) are computed as follows:

Table 3 - PID Tuning Parameters - Physical System

No.	Parameter	PMC
1	Proportional Gain, rp	0.961
2	Integral Gain, ri	0.027
3	Derivative Gain, rd	0.035

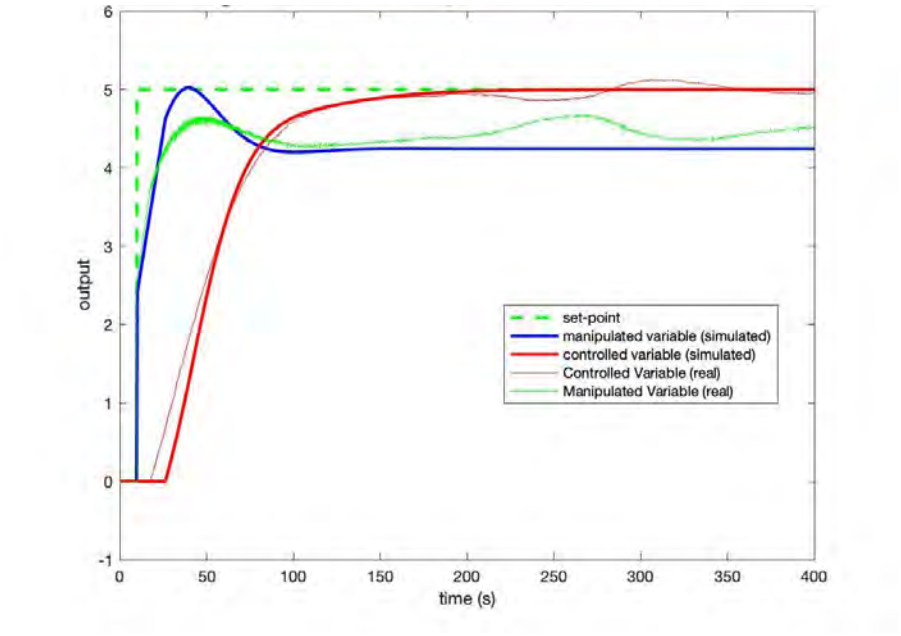

Figure 6 - PID tuning results

The figure (6) shows the PID tuning results of the physical setup. As can be seen from the figure, the controlled variable from the real setup closely matches the controlled variable from the simulated model and thus, is a close reflection of the actual process. Additionally, the PID parameters selected using the PMC tuning procedure are extremely precise and with minimum overshoot.

6 Conclusion

The GEA algorithm is evaluated as a potential method for optimization of a wide variety of functions. In this article, the algorithm is used perform system identification of an unknown process from which the controller parameters are calculated. One of the main advantages is that the method is suitable for different types of processes, be it linear or non-linear processes. Additionally, for higher order controllers, optimization of the controller can be performed using the GEA, thus making it a universal approach in solving problems of greater complexity.

Acknowledgement

This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS19/158/OHK2/3T/12

REFERENCES

[1] L. a. X. L. a. G. E. Cao, "A Guiding Evolutionary Algorithm with Greedy Strategy for Global Optimization Problems," *Computational Intelligence and Neuroscience,* vol. 2016, pp. 1-10, 2016.

[2] X.-S. Yang, "A New Mataheuristic Bat-Inspired Algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, Berlin, Springer, 2010, pp. 65-74.

[3] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, Perth, 1995.

[4] J. H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, Cambridge, MA: MIT Press, 1992.

[5] A. Saldanha and M. Hofreiter, "Relay Feedback Identification using Guiding Evolutionary Algorithm," in *New Methods and Practices in the Intrumentation, Automatic Control and Informatics 2020*, Lobec, 2020.

[6] K. Astrom and T. Hagglund, "Automatic Tuning f Simple Regulators with Specifications on Phase and Amplitude Margins," *Automatica, Volume 20, Issue 5,* pp. 645 - 651, 1984.

[7] V. Ramakrishnan and M. Chidambaram, "Estimation of a SOPTD transfer function model using a single asymmetrical relay feedback test," *Computers & Chemical Engineering, Volume 27, Issue 12,* pp. 1779-1784, 2003.

[8] "PID Controller," 09 08 2020. [Online]. Available: https://en.wikipedia.org/wiki/PID_controller.

[9] M. Hofreiter, Zaklady Automatickeho Rizeni, Prague: CVUT, 2016.

[10] A. Gupta, *Bat Optimization Algorithm,* MATLAB Central File Exchange, 2020.

Selected article from
Tento dokument byl publikován ve sborníku

Nové metody a postupy v oblasti přístrojové techniky, automatického řízení a informatiky 2021

New Methods and Practices in the Instrumentation, Automatic Control and Informatics 2021

15. 9. – 17. 9. 2021, Žatec

ISBN 978-80-01-06889-2

Web page of the original document:
http://iat.fs.cvut.cz/nmp/2021.pdf

Obsah čísla/individual articles:
http://iat.fs.cvut.cz/nmp/2021/

Ústav přístrojové a řídicí techniky, FS ČVUT v Praze, Technická 4, Praha 6