COMPARISON OF SEVERAL GENETIC ALGORITHM STRATEGIES ON A NONLINEAR GAPID CONTROLLER OPTIMIZATION APPLIED TO A BUCK CONVERTER

Fábio Galvão Borges*, Marco A. Itaborahy Filho*, Hugo V. Siqueira*, Fernanda C. Corrêa*, Maurício S. Kaster*

*Departamento de Eletrônica
Universidade Tecnológica Federal do Paraná
Ponta Grossa, PR, Brazil.

Emails: binhogborges@gmail.com, marco.itaborahy@hotmail.com, hugosiqueira@utfpr.edu.br, fernandacorrea@utfpr.edu.br, mkaster@utfpr.edu.br

Abstract— This work presents a comparison of six versions of the Genetic Algorithm to optimize the parameters of the nonlinear GAPID controller (Gaussian Adaptive Proportional, Integral and Derivative), elaborated to control a step-down DC-DC converter. This task comprises 8 free parameters and there is no analytic solution to solve it. Also, the design of the controller is hard to determine because there can exist several near-optimal solutions with different values for the parameters, which defines this problem as multimodal. In this sense, different optimization strategies can lead to different solutions. This paper analyzes the behavior of six distinct Genetic Algorithm strategies and compares the obtained results.

Keywords— Genetic Algorithm, Optimization, Gaussian Adaptive PID Control

1 Introduction

Historically humanity, in the seek to evolve technologically, has faced with increasingly more complex problems, many of them cannot be solved directly by employing traditional methods. Optimization tools are helpful on finding solutions to these problems (Jones et al., 2002).

In the field of control systems, more advanced controllers applied to highly nonlinear plants frequently need complex design procedures (Dimeo and Lee, 1995), some only achievable by employing optimization tools (Puchta et al., 2016).

Traditionally, the linear PID controller designed for one nominal operating point based on the local-linearization small-signal model is employed in many industrial applications. However, many plants does not follow a predicted behavior, exhibiting time-varying characteristics and sometimes a large operating point variation, which impacts directly the plant's model and, consequently, pushing the controller out of the designed operational point. So, the controller design is a challenging task to be coped with through conventional linear PID control strategy (Krohling and Rey, 2001).

Trying to overcome this limitations, several nonlinear alternatives are being explored: Adaptive, Sliding mode (Mattavelli et al., 1993), optimal controllers (LQR, SDRE) (Fujimoto et al., 2010), among others (Gahinet et al., 1994), that can achieve good performance enhancements. But some of them are very complex and demand high performance hardware with higher cost (Fujimoto et al., 2010).

Some adaptive controllers seek to keep the PID structure. This enables taking into account the design requirements of the original PID and then improve its performance while using the same design specifications of the PID (Puchta et al., 2016).

GAPID is a a kind of adaptive control based on the linear PID with the adaptive rule defined by a gaussian function (see section 3) (Puchta et al., 2016). The gaussian function presents some desirable characteristics, like being smooth with smooth derivatives and being top and bottom bounded. Some other approaches also employ smooth bounded functions: In (Pedroso et al., 2013) a hyperbolic tangent function is used as the adaptive function. In (Hawwa and Masoud, 2006) a double-gaussian-like function is used to adapt the derivative gain. These approaches need more computing effort to calculate the employed functions when compared to GAPID.

In this work, GAPID is used to control the output voltage of a step-down DC-DC converter widely used as power supply of several electrical and electronic equipment. The main objective of the converter is to maintain a constant output voltage under variable load current and unregulated input voltage. The transient overshoot and recovery time of the output voltage must be minimized for stable operation in many electronic applications (Yuan et al., 2015).

As there is no algebraic design method for GAPID, then optimization tools can be employed (Puchta et al., 2016).

As part of the Bio-inspired Metaheuristics, Genetic algorithm (GA) is a classical optimization method with several variations. GA counts on three fundamental operators on every generation namely the selection, crossover, and mutation. For each generation, solutions are selected from the population based on the fitness value. A

number of chromosomes is fixed. Then, they undergo the crossover and/or are passed through the mutation to generate a new population. Finally, each member of the current population is tested to evaluate the ability to solve the optimization problem. (Badis et al., 2016). Based in this operators, which may change, this paper will examine the application of six GA strategies to GAPID design.

This paper is organized as follows: In Section 2 a brief description of the application circuit, the step-down DC-DC converter, is given, with its mathematical model. In Section 3 the GAPID controller is described, defining the adaptive function of gains and its parameters. In Section 4 a short description of the Genetic Algorithm is given and the six proposed variations to be tested in the optimization of GAPID parameters. In Section 5 the chosen GA parameters and figures summarizing the evolution of each GA strategy are shown. Then the output voltage waveforms of the Buck converter are shown in order to compare the performances of traditional PID and GAPID. Finally, Section 6 brings up the conclusions.

2 Application circuit

The step-down converter with its control circuit is presented in Figure 1.

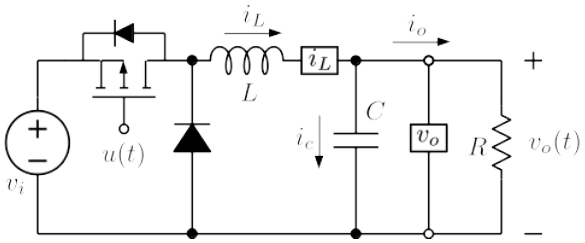


Figure 1: Schematics of the converter and the control system.

The converter can be described by the state-space equations

$$\begin{aligned} L\frac{di_L}{dt} &= -v_c + uV_i \\ C\frac{dv_c}{dt} &= i_L - \frac{1}{R}v_c \end{aligned} \tag{1}$$

where L is the converter inductor, C is the capacitor, R is the load, V_i is the supply voltage, i_L and v_c are the state variables for inductor current and capacitor (output) voltage, and u is the control input. This system presents the following control input-to-voltage output transfer function

$$\frac{v_o(s)}{u(s)} = \frac{V_i}{LCs^2 + \frac{L}{R}s + 1}. \tag{2}$$

This system is controlled by a GAPID controller presented in the next section.

3 Gaussian Adaptive PID

It is known that nonlinear controllers are more efficient than linear controllers; the problem is they are harder to design. In this work, a nonlinear adaptive PID controller is used, where the adaptive rule is based on a Gaussian function defined by

$$f(\delta) = k_1 - (k_1 - k_0)\mathrm{e}^{-q\delta^2} \tag{3}$$

where k_0 and k_1 are the upper and lower bounds of the function and q defines the degree of concavity. This function is shown in figure 2.

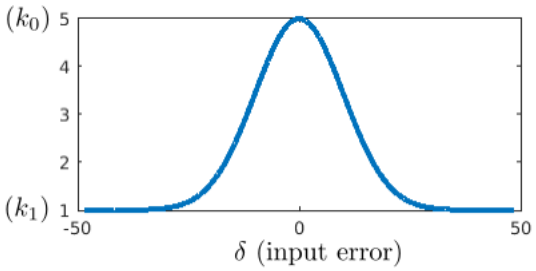


Figure 2: Gaussian function.

The Gaussian function represents a smooth function with smooth derivatives and is upper and lower bounded. Being smooth, it avoids abrupt changes in the gains that can lead to problems in some systems, like the occurrence of chattering due to fast repetitive changes in the gains. Being bounded, the designer can establish clear limits for the gains in the whole error range.

This gaussian function has three parameters: k_0, k_1 and q. Each of the PID gains employ gaussian functions each with its own set of parameters. This initially comprises a nine parameters problem. However, an important assumption is to define the value of k_{0d} as zero. This constant represents the derivative gain when the set-point is achieved, i.e., when the error is null. In this situation, the controller presents only Proportional-Integral actions helping avoiding derivative noise issues.

At the moment, there is no methodology to solve the problem algebraically, so an optimization algorithm based on metaheuristics is employed in order to find an optimized solution to the problem, which allows to achieve the faster response with low overshoot. Genetic Algorithm was chosen, and will be presented in the next section.

4 Genetic Algorithm

Genetic Algorithm (GA) is a metaheuristic inspired by the biological process of evolution by natural selection, developed to deal with optimization problems. GA is a probabilistic method, acting on individuals (or gene) and influencing the population indirectly (Holland, 1992).

The optimization process using a Genetic Algorithm is initiated creating multiple random candidate solutions (individuals) to the task. Each individual is characterized by its coordinates on the problem-solving space, being these the equivalent of a genotype of a living creature. In other words, an individual is a vector containing the values of the coefficients of the problem addressed (Castro, 2006).

The individuals present a phenotype, named the fitness. The fitness behave like a score: solutions with higher fitness values are more likely to survive to the process of selection and be able to maintain their genotype in the next generation.

The next step is the application of the selection procedure. This allows the assortment of some individuals to participate to the next steps of the GA. There are many ways to perform that, being the most used the roulette wheel and tournament (Goldberg, 2006).

In the first case, a roulette is created considering that each part is proportional to the fitness of the individuals. It means that the "slice" correspondent to better individuals are greater than those of the worse ones. Then, the probability to draw the genes with best fitness is more likely. Also, the same individual may be selected more than once.

The second proposal, the tournament, is initiated selecting randomly some individuals, usually two. The competition is the comparison of their fitness values. The winner is the one which present the higher fitness. As the participants can be some of the best, the selective pressure is lower than in the roulette wheel method(Michalewicz, 1996).

The subsequent generation is created by the application of the genetic operators. The first is the crossover, in which two individuals (parents) are randomly chosen among those that survived after the selection process. Then, two new individuals are created using the genotype of both parents. To explain that, we use an example: suppose that the individuals (vectors) have 10 dimensions. We choose the fifty dimension as the cut point. Therefore, the first new individual will have the first five elements of the parent 1 and the last five elements of the parent 2. Obviously, the second new one will have the other dimensions of both parents. The proposal is known as "one-point crossover". This process allows the local search, or the exploitation (Holland, 1992). A scheme of the one-point crossover is in figure 3.

The second operator is the mutation. This method changes randomly some of the genes of the entire population, with the view of become possible a global search, or the exploration. Therefore, a percentage of the genes must be selected to be mutated. Figure 4 shows an example of this process (Castro, 2006).

The new generations are evaluated and the process is repeated until a fitting solution be found. A summary of a Genetic Algorithm is shown in the flowchart on Figure 5.

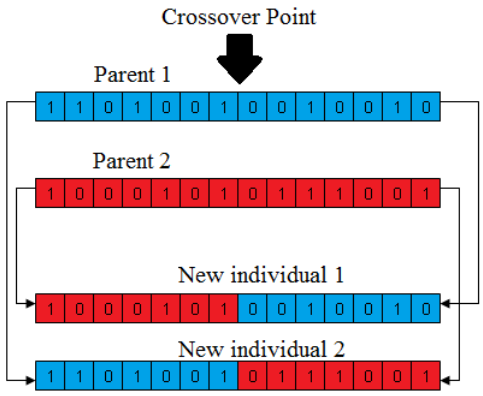


Figure 3: Example of the Crossover.

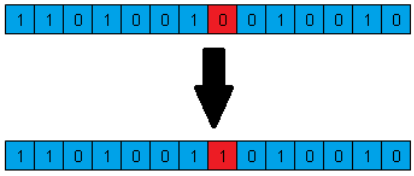


Figure 4: Example of the Mutation.

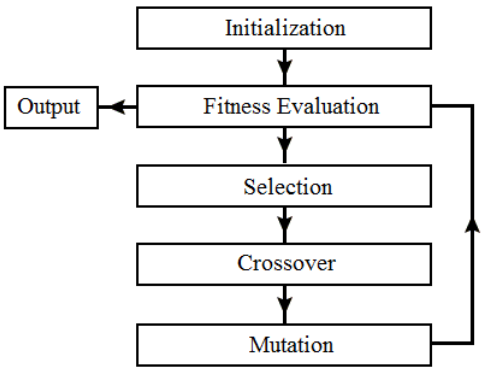


Figure 5: Flowchart of the GA.

4.1 Genetic Algorithm Variations

The proposed method presented in the last Section showed the general steps to be followed in the implementation of the Genetic Algorithm. However, it is possible to change same parts of the canonical proposal, creating some variations of the GA. It can be done, for example, changing the selection process or applying rates to determine if the genetic operators will be performed (Michalewicz, 1996). We highlight that the mutation rate adopted in this work to all propositions was 10%.

GA 1

The first GA implemented was the original method described by John Holland (Holland, 1992). In this case, the selection is performed using the roulette wheel, as described in Section 4.

The following step is the application of the genetic operators. Here, we have to determine a probability of occurrence of the crossover or the mutation. In the first case, we consider a rate of 70%. It means that after the selection of the parents, they have 70% of chance to perform the crossover. Otherwise, they stay in the population as the offspring.

GA 2

The second GA proposal is quite similar to the last. The only difference is that the crossover rate is set as 100%. In this approach, all the parents selected must perform the crossover. It is obvious that they never participate of the new population.

GA 3

The next approach differs from the GA 2 in the selection of the individuals. Here, the binary tournament is addressed. However, even if some individual loose the competition, it is able to participate again.

GA 4

In the fourth approach, we apply the "death tournament" to perform the selection. Therefore, the individuals that loose the tournament are suppressed of the population. In this way, each one just participate only once of the selection.

GA 5

The following GA is quite different from the previous. In this case, we change the order of the operations described in figure 5, so that the selection is applied after the mutation. The consequence is the creation of a intermediate subpopulation, having twice times the number of the individuals. Then, the selection decrease the population size to the original quantity.

In this proposal the roulette wheel is used to select the next population.

GA 6

Finally, in the last GA, we follow the same steps of the GA 5. The only difference is the application of the tournament to select the individuals to the next generation.

4.2 Fitness function

In optimization tasks the assessment of the individuals plays a fundamental role, which is accomplished by the fitness function, similar to the cost function, where higher scores are given to best solutions. It is expected that in the evolution process, the mean score of the population increases and at least one individual approaches the maximum point.

It is important to observe that a wrong choice of such function compromises the results because the highest score may not match the best solution of the problem.

The objective is to select a solution that presents the best possible solution with a maximum admissible overshoot of 5%.

In this sense, a good alternative is the Integral Absolute Error (IAE) that has been used by several research groups worldwide.

The IAE function is given by

$$IAE = \int_0^\infty |\delta(t)|dt \tag{4}$$

where $\delta(t)$ is input error, and the fitness function is given by

$$fit = 1/IAE. \tag{5}$$

Therefore, the cost function defined in eq. (4) must be minimized, it means that we have to maximize the fitness defined in eq. (5).

5 Results

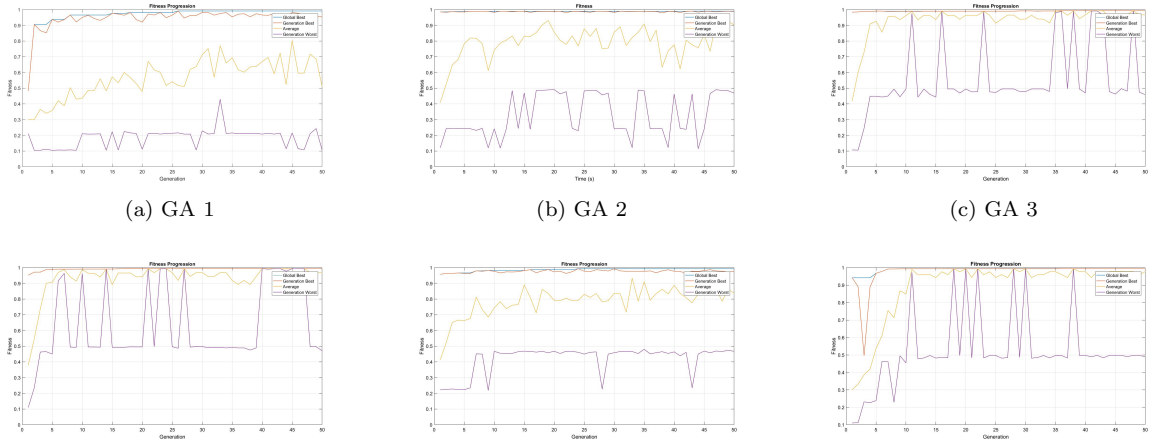
In this section we present the computational results achieved by the six proposals of Genetic Algorithm applied to the GAPID controller. To all proposals we use 50 iterations and a population with 40 individuals. It is important to highlight that they were through empirical preliminary tests:

The final results of each proposal are summarized in table 1. The boxplot of the results is in figure 7.

GA model	Fitness
GA1	0.993574
GA2	0.993661
GA3	0.993898
GA4	0.993680
GA5	0.993432
GA6	0.993999

Table 1: Physical simulation parameters.

As one can be note, the best results are related to the proposals GA3, GA4 and GA6. The main similarity of these schemes is that all of them use the binary tournament as the selection proposal. It seems to be clear that to utilize a methodology with less selective pressure is an advantage to



(a) GA 1 (b) GA 2 (c) GA 3

(d) GA 4 (e) GA 5 (f) GA 6

Figure 6: Flowchart of the GA.

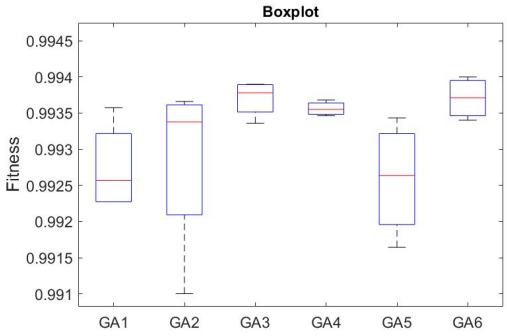


Figure 7: Boxplot of the fitness

this application. Therefore, the use of the roulette wheel should be avoided.

Comparing the performances of the best proposals, the GA3 achieved the best general results. In the same way, the GA4 presented the the lowest dispersion.

6 Conclusions

This paper presented the application of six Genetic Algorithm strategies to find the best parameters configuration of a Gaussian Adaptive PID controller. The results of each strategy demonstrated some little differences, as shown in figure 7(boxplot) where strategies 3 and 6 achieved a higher fitness of the best individual, with relatively low dispersion. Strategy 4 presented the lowest dispersion and a relatively high fitness, similar to strategy 2. In general, strategies 3, 4 and 6 performed better. Taking the results obtained in strategy 6, the optimized GAPID performed as presented in figure 6 demonstrating its effectiveness for enhancing the traditional PID performance using the same design requisites.

References

Badis, A., Mansouri, M. N. and Sakly, A. (2016). Pso and ga-based maximum power point tracking for partially shaded photovoltaic systems, *Renewable Energy Congress (IREC), 2016 7th International*, IEEE, pp. 1–6.

Castro, L. N. (2006). *Fundamentals of Natural Computing: Basic Concepts, Algorithms and Applications*, Chapman & Hall/CRC.

Dimeo, R. and Lee, K. Y. (1995). Boiler-turbine control system design using a genetic algorithm, *IEEE Transactions on Energy Conversion* **10**(4): 752–759.

Fujimoto, T., Tabuchi, F. and Yokoyama, T. (2010). A design of fpga based hardware controller for dc-dc converter using sdre approach, *The 2010 International Power Electronics Conference - ECCE ASIA -*, pp. 1001–1005.

Gahinet, P., Nemirovskii, A., Laub, A. J. and Chilali, M. (1994). The lmi control toolbox, *Proceedings of 1994 33rd IEEE Conference on Decision and Control*, Vol. 3, pp. 2038–2041 vol.3.

Goldberg, D. E. (2006). *Genetic algorithms*, Pearson Education India.

Hawwa, M. and Masoud, A. (2006). A nonlinear pid servo controller for computer hard disk drives, *Advanced Motion Control, 2006. 9th IEEE International Workshop on*, pp. 672–676.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, MIT press.

Jones, D., Mirrazavi, S. and Tamiz, M. (2002). Multi-objective meta-heuristics: An overview of the current state-of-the-art, *European Journal of Operational Research* **137**(1): 1 – 9.
URL: *1*

Krohling, R. A. and Rey, J. P. (2001). Design of optimal disturbance rejection pid controllers using genetic algorithms, *IEEE Transactions on Evolutionary Computation* **5**(1): 78–82.

Mattavelli, P., Rossetto, L., Spiazzi, G. and Tenti, P. (1993). General-purpose sliding-mode controller for dc/dc converter applications, *Power Electronics Specialists Conference, 1993. PESC '93 Record., 24th Annual IEEE*, pp. 609–615.

Michalewicz, Z. (1996). Evolution strategies and other methods, *Genetic algorithms+ data structures= evolution programs*, Springer, pp. 159–177.

Pedroso, M. D., Nascimento, C. B., Kaster, M. S. and Tusset, A. (2013). A hyperbolic tangent adaptive pid + lqr control applied to a step-down converter using poles placement design implemented in fpga, *Mathematical Problems in Engineering* **2013**: 1–8.

Puchta, E. D., Lucas, R., Ferreira, F. R., Siqueira, H. V. and Kaster, M. S. (2016). Gaussian adaptive pid control optimized via genetic algorithm applied to a step-down dc-dc converter, *Industry Applications (INDUSCON), 2016 12th IEEE International Conference on*, IEEE, pp. 1–6.

Yuan, Y., Chang, C., Zhou, Z., Huang, X. and Xu, Y. (2015). Design of a single-input fuzzy pid controller based on genetic optimization scheme for dc-dc buck converter, *Next-Generation Electronics (ISNE), 2015 International Symposium on*, IEEE, pp. 1–4.