Jornadas de Automática

Multi-objective & multi-model PI/PID controller tunning

Callas, E.***, José, I.*, de Lima, L.*, Soriano, M.*, Ayala, H.**,****, Reynoso-Meza, G.**,****

** Escola Politécnica, Pontifícia Universidade Católica do Paraná*
***Programa de Pós Graduação em Engenharia de Produção e Sistemas, Pontifícia Universidade Católica do Paraná*
**** Facultad de Ingenieria, Universidad San Ignacio de Loyola.,*
*****Laboratório de Otimização de Sistemas de Controle, Pontifícia Universidade Católica do Paraná*

To cite this article: Callas, E., José, I., de Lima, L., Soriano, M., Ayala, H., Reynoso-Meza, G. 2024. Multi-objective & Multi-model PI/PID Controller Tunning. Jornadas de Automática, 45. https://doi.org/10.17979/ja-cea.2024.45.10855

Resumen

Este estudio aborda el desafío de modelar un controlador para una turbina eólica de 15 MW propuesta por la Universidad de Córdoba. Ante la falta de disponibilidad de un modelo que represente el funcionamiento interno de la turbina, se resolvió utilizando la herramienta de Identificación de Sistemas en MATLAB. Se empleó el modelo lineal ARMAX, y se aplicaron los criterios de Vinnicombe y Akaike para abordar múltiples modelos, objetivos y seleccionar las mejores funciones de transferencia. Inicialmente, se obtuvieron 15 funciones de transferencia. Posteriormente, el controlador PI propuesto se ajustó a un PID con un filtro utilizando las cuatro funciones de transferencia encontradas mediante el uso del algoritmo de optimización multiobjetivo spMODEx, encontrando ganancias óptimas con un Kp=-8, Ti=12, Td=0.9754, y un filtro=10, lo que resultó en un J=0.4602, superando óptimamente el valor propuesto por el desafío, J=0.5206. Las ganancias encontradas en este estudio demostraron su viabilidad para operar en una amplia gama de sistemas, respaldando un enfoque de múltiples modelos

Palabras clave: Diseño de sistemas de control, Sistemas energéticos, Control de recursos renovables, estimación y filtracón, Algoritmos evolutivos para control óptimo

Multi-objective & Multi-model PI/PID Controller Tunning

Abstract

This study addresses the challenge of modeling a controller for a 15 MW wind turbine proposed by the University of Córdoba. Faced with the unavailability of a model representing the internal workings of the turbine, it was resolved using the System Identification tool in MATLAB. The linear ARMAX model was employed, and the Vinnicombe and Akaike criteria were applied to address multiple models, objectives, and select the best transfer functions. Initially, 15 transfer functions were obtained. Subsequently, the proposed PI controller was adjusted to a PID with a filter using the four transfer functions found through the use of the multi-objective optimization algorithm spMODEx, finding optimal gains with a $Kp = -8, Ti = 12, Td = 0.9754$, and a $filter = 10$, resulting in a $J = 0.4602$, optimally surpassing the value proposed by the challenge, $J = 0.5206$. The gains found in this study demonstrated their viability to operate across a wide range of systems, supporting a multi-model approach.

Keywords: Control system design, Energy systems, Control of renewable resources, estimation and filtering, Evolutionary algorithms for optimal control

1. Introduction

*Autor para correspondencia: ernesto.leyva@usil.pe
Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)

The growing necessity for renewable energy arrived as a challenge for contemporary engineering, as it becomes necessary to seek power sources that haven't yet been completely understood. In this scenario, mechanisms were created to generate power through wind (Ackermann, 2012), which have evolved to the current wind turbines. However, as with any electromagnetic system, this has some limitations, it is necessary that certain measures aren't exceeded, so the internal functioning of the turbine isn't harmed. One of these measures is the power generated by the turbine, which should stay mostly constant during the generation. However, the power source for the turbine, wind, isn't constant, and, at the moment, there's no mechanism to predict with precision its intensity and direction, mainly by the fact it's not a linear measure. Therefore, one of the main problems that appeared is related to how to control the turbine's power generation without taking into consideration that the power source is completely unpredictable (Hau, 2013).

In this context, the University of Córdoba proposed a challenge to undergraduates and postgraduates to model a control system for wind turbines (Garrido, et al., 2024). Since the model that represents the internal working of the turbine wasn't available, first it was necessary to identify this model. For the identification of the proposed model, data provided by the University was used, which were related to the turbine inputs and outputs, and possible disturbances that may happen in the power generation process. The data provided was:

Vx: is the wind velocity. Since this disturbance isn't predictable, it can't be controlled. Its variation is defined by the University, with a range from 11m/s to 25m/s.

Pg: is the power generated by the turbine, measured in MW. It's ideal to stay constant and equal to 15(MW), being one of the controlled variables. This variable is defined as the product of the Electromagnetic Torque (Tem) by the Angular Velocity (ωg). However, the torque value will be treated as constant, making the generated power dependent only of Angular Velocity (ωg).

ωg: The turbine's angular velocity, measured in rad/s. Its maximum variation must be between -2 (rad/s) and 2 (rad/s).

M_{eje}: Torque of the generating shaft, measured in $MN * m$. According to the University, must take in consideration that there may not have a great variation in this parameter.

Servo pitch: It's the turbine's blade's real angle pitch, considering its internal servo. Measured in degrees, will be treated as constant and equal for all three blades.

Pitch control: Turbine's blade's angle pitch, considered the same for the three blades. Can vary from 0 to 90 º, however, in normal conditions, usually doesn't surpass 30 º.

Tem: Turbine's electromagnetic torque. Will be treated as a constant equal to 19786767.5 (Nm).

The University made available multiple files, containing a Matlab script to run simulations and benchmark tests, Simulink file that has both the turbine and PI controller models, and the Open Fast compiler, which is used to properly simulate the turbines. The default gains for the PI are: -6, and –0.6, for the proportional, and integral parts respectively. This work's objective is to implement another controller with a similar or better performance than the proposed controller.

2. Methods and tools

In this study it was used a multi-model and multi-objective approach to determine a better controller than that of the proposed by the challenge. Such an approach determined the methods and tools used in this challenge, since, according to Huilcapi, et al. (2019), using a multi-model approach to design a controller can provide an optimized response compared to the approach with a single model. The author explains this by stating that designing a controller for each representative function of a system and then using the Vinnicombe method to find a global controller increases the chances of finding a controller consistent with the model to be controlled, especially in black-box models where the actual plant function is unknown (Vinnicombe, 1993). Therefore, it used the Matlab System identification toolbox, specifically the ARMAX polynomial setting to identify the proposed system, followed by the Vinnicombe metric to determine which transfer functions were similar. Finally, the spMODEx algorithm was used to produce both the PI and PID gains used.

2.2. System Identification

The Matlab software's System Identification tool serves to identify black-box systems, which are those where physically depicting their operation is unfeasible, necessitating their representation through inputs and outputs (Zhang, 2010). As the University did not provide the function representing the turbine's internal operation, the decision was made to employ this tool for system identification.

Therefore, Matlab's System Identification Toolbox, basically allows to choose which method will be used to identify the transfer functions, utilizing input data previously chosen by the user. The input and output data were made available by the University, making it possible, with multiple steps generations, to build a database coherent with the one asked by the Toolbox.

In this study, the researchers used the linear ARMAX (Autoregressive Moving Average with Extra Input) model as the reference structure for the "System Identification" tool. MATLAB employs machine learning techniques to solve the estimation of its models (MathWorks, 2024). Additionally, the least squares method is used for greater accuracy, and it is important to manually define the polynomial order in [na nb nc nk] for ARMAX. The advantages of using this model include estimation covariances (parameter uncertainties) and the quality of the fit between the estimated and measured data. The ARMAX model structure can be seen in (1) (Ljung, 1999).

ARMAX MODEL STRUCTURE:

$$y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) = b_1 u(t-n_k) + \cdots + b_{n_b} u(t-n_k-n_b+1) + c_1 e(t-1) + \cdots + c_{n_c} e(t-n_c) + e(t) \quad (1)$$

Where:
$y(t)$: Output at time t.
n_a: Number of poles.
n_b: Number of zeros plus 1.
n_c: Number of C coefficients.
n_k: Number of input samples that occur before the input affects the output, also called the dead time in the system.
$y(t-1) \dots y(t-n_a)$: Previous outputs on which the current output depends.
$u(t-n_k) \dots u(t-n_k-n_b+1)$: Previous and delayed inputs on which the current output depends.
$e(t-1) \dots e(t-n_c)$: White-noise disturbance value.

On the other hand, the University of Cordoba proposed the following model, revealing only the inputs and outputs:

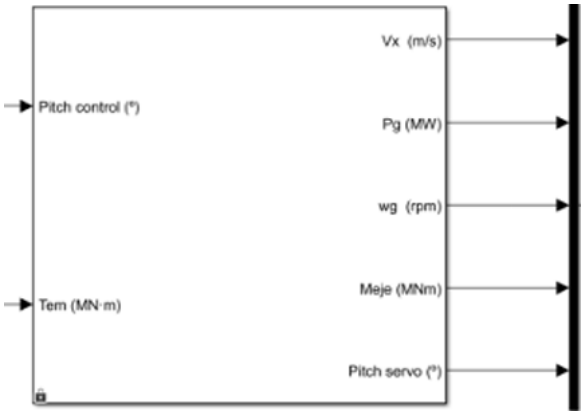

Figure 1: Model proposed by the University of Cordoba

In Image 1, it can be observed that the inputs to this function are the blade pitch angle and torque as a constant. Among all the outputs of the plant, the only one with data provided by the University is the angular velocity. The others will be treated as constants or disturbances that do not have a feasible implementation.

This way, the initial and final input steps values were also chosen, and when would happen, to get more diversified functions. These values vary from 30 to 90, while the time for the step occurrence were chosen between the options 1, 40 or 100 seconds. The generated functions were on the discrete domain "z", so, it was necessary to utilize a command to transform the functions to continuous time, on the domain "s".

With this data, a total of 15 transfer functions were found. Additionally, to optimize this entire process, the researchers used the Akaike criterion (AIC). This criterion was proposed as an approach to balance the model's complexity and fit. Its utilization facilitates the avoidance of overfitting by imposing a penalty on overly complex models, constituting a fundamental tool for more effective model selection in the advancement of system identification (Akaike, 1974). The following AIC structure was used:

$$AIC = N * \log\left(\det\left(\frac{1}{N}\sum_1^N \varepsilon(t,\tilde{\theta}_N)\left(\varepsilon(t,\tilde{\theta}_N)\right)^T\right)\right) + 2n_p + N * (n_y * (\log(2\pi) + 1)) \quad (2)$$

Where:
N : is the number of values in the estimation data set.
$\varepsilon(t)$: is a $n_y - by - 1$ vector of prediction errors.
$\varepsilon(t,\tilde{\theta}_N)$: model residuals.
$\tilde{\theta}_N$: represents the estimated parameters.
n_p: is the number of estimated parameters.
n_y: is the number of model outputs.
t: It is an index that represents each individual observation in the data. From the first $(t = 1)$ to the last $(t = N)$.
T: it is the transpose of the residual matrix.

Additionally, when comparing multiple proposed models and applying a variation of AIC, it is possible to evaluate the robustness of each model (Burnham, et al., 2024). This combination of concepts is particularly suitable when working with various models and aiming to handle a larger volume of data with the fewest possible variables (Ljung, 1999).

It's important to say that MAtlab's Toolbox already implements Akaike's method to each function generation, therefore, to each data import and method selection, only one function was generated, because the other ones had already been filtered by Akaike's criterion.

Table 1 shows the functions gathered by the methods previously cited. The sampling time used was 0.01 seconds, and the steps initial value was always 0. It must be considered that "V.F" stands for the step final values, and "T" stands for the time the step was applied. "F.T" stands for the transfer function of each configuration. The step final values refer to the angle on the turbine's blades. Attention should be paid to the fact that the final value to be controlled (angular velocity) was included on the data imported to the System Identificator.

Table 1: Identified Functions

T	V.F	F.T.
1	30	$P_1 = \frac{-0.001959s - 0.01179}{s^2 + 3.178s + 0.1249}$
40	30	$P_2 = \frac{-0.002392s - 0.002355}{s^2 + 1.178s + 0.01155}$
100	30	$P_3 = \frac{0.0001676\,s + 0.002403}{s\text{^}2 + 4.264\,s + 0.007993}$
1	45	$P_4 = \frac{-0.000363s - 0.01267}{s^2 + 5.841s + 0.2786}$
40	45	$P_5 = \frac{-9.281e-05s - 0.00196}{s^2 + 1.729s + 0.01479}$
100	45	$P_6 = \frac{0.0004289s - 0.002137}{s^2 + 2.153 + 0.008066}$
1	60	$P_7 = \frac{0.001599s - 0.01094}{s^2 + 7.887s + 0.3841}$
40	60	$P_8 = \frac{0.0004515s - 0.001585}{s^2 + 2.059s + 0.01612}$
100	60	$P_9 = \frac{0.0001848s - 0.00161}{s^2 + 2.295s + 0.007989}$

1	75	$P_{10} = \frac{-0.0005612s - 0.006725}{s^2 + 6482s + 0.3308}$
40	75	$P_{11} = \frac{0.0003496s - 0.001342}{s^2 + 2.264s + 0.01705}$
100	75	$P_{12} = \frac{0.0001169s - 0.001436}{s^2 + 2.644s + 0.008743}$
1	90	$P_{13} = \frac{-0.0002986s - 0.003043}{s^2 + 2.202s + 0.1838}$
40	90	$P_{14} = \frac{-3.048 * 10^{-5}s - 0.00116}{s^2 + 2.138s + 0.0227}$
100	90	$P_{15} = \frac{0.0001722s - 0.001009}{s^2 + 2.101s + 0.008874}$

After analyzing these functions, only the negative functions were selected through the final value theorem. This was necessary because the controller proposed by the University is unable to control positive functions (Garrido, et al., 2024). Therefore, function P3 was eliminated.

This allows researchers to analyze the logic that should be reflected in their transfer functions; firstly, they verify the stability of the systems under consideration. This means that the final value of the systems response is equal to the limit of the transfer functions as time tends to infinity, evaluated at $s = 0$. The sign of the final value indicates whether the system's response tends upwards (positive) or downwards (negative) as time progresses indefinitely, providing crucial insights into the long-term behaviour of the systems in the time domain (Chen, et al., 2007).

Therefore, a total of 14 functions remained. To filter them, the Vinnicombe method was used, which is based on the proximity of the plants found and whether they can or cannot be controlled by the same controller (Vinnicombe, 1993). To make this test, a heatmap was built based on Vinnicombe's metric values, for better visualization of the distance between the functions. Consequently, functions farther apart from each other were selected, as it is considered that the controller should be able to control the plant in as many situations as possible. Thus, the functions found totaled 4:

$$P_1 = \frac{-0.001959*s\ -0.01179}{s^2+3.178*s+0.1249} \quad (3)$$

$$P_6 = \frac{0.0004289*s\ -0.002137}{s^2+2.153*s+0.008066} \quad (4)$$

$$P_{12} = \frac{0.0001169*s\ -0.001436}{s^2+2.644*s+0.008743} \quad (5)$$

$$P_{13} = \frac{-0.0002986*s\ -0.003043}{s^2+2.202*s+0.1838} \quad (6)$$

Utilizing the controller provided by the University, it was posible to confirm that, despite being far from each other, these functions were controlled by the same controller.

2.3. Multi objective

In engineering, situations often arise where multiple objectives must be addressed simultaneously, potentially leading to conflicts as improvements in some aspects may worse others. To address these challenges, engineers employ multi-objective optimization techniques (MOP), aiming to find sets of optimal solutions rather than a single definitive solution. These optimal solutions represent trade-offs between design objectives, with some inherently better than others. This collection of optimal solutions is known as the Pareto front. In practical applications, optimization algorithms typically identify sets of solutions rather than a singular solution (Huilcapi, et al, 2019). A design concept refers to an idea aimed at solving a specific problem, with each concept corresponding to a proposed MOP and its associated Pareto front. In the realm of process control, a design concept may align with a specific control structure or a particular combination of inputs and outputs. For instance, in a multivariable process defined by a transfer function matrix with multiple inputs and outputs, it is essential to define parameters and conditions for developing an effective controller (Huilcapi, et al, 2019). By comparing the Pareto fronts linked to each design concept, engineers can utilize personal preferences to select a suitable controller for the plant. Moreover, if the chosen controller fails to fully meet expectations, engineers can iterate the process across various scenarios, adjusting parameters and conditions as needed. This adaptability underscores the advantages of employing multi-objective optimization techniques.

2.4. Selection criterion

The article provided by the University specifies the parameters to be used for defining an effective controller Garrido, et al., 2024). As seen in (7), (8), (9), (10), and (11).

$$J(CP) = \mathrm{w1} * \mathrm{R1} + \mathrm{w2} * \mathrm{R2} + \mathrm{w3} * \mathrm{R3} + \mathrm{w4} * \mathrm{R4} \quad (7)$$

Where:

$J(CP)$: Global performance index

$\mathrm{w1}, \mathrm{w2}, \mathrm{w3}, \mathrm{w4}$: The weights of each parameter (when summed, equal 1). These weights have not been provided.

$$\mathrm{R1} = \frac{RMSE(wg)}{0.5} \quad (8)$$

$$\mathrm{R2} = \frac{RMSE(Pg)}{1} \quad (9)$$

$$\mathrm{R3} = \frac{RMSE(\dot{Meje})}{1} \quad (10)$$

$$\mathrm{R4} = \frac{RMSE(\dot{\beta})}{2} \quad (11)$$

wg: Turbine angular velocity.

Pg: Power generated by turbine.

$\dot{Meje}$: Variation (derived) of generator shaft torque.

$\dot{\beta}$: Variation (derived) of the turbine pitch angle.

$RMSE(parameter)$: It represents the root mean square error for each analyzed parameter. The values of R1, R2, R3, and index value, the worse the controller performance. However, since the controller proposed by the University has a $J(CP)$ of 0.5206, the aim is to find a controller with an index approximately equal or better.

3. **Results**

3.1. SpMODEx configuration

To properly tune the PI gains, it was used a multi-objective optimization Algorithm, known as spMODEx available at the MATLAB file exchange (Reynoso-Meza, 2024). Using the four transfer functions identified that

represent the system, the algorithm was used to identify possible PI gains, specifically Kp, and Ti. The algorithm was set up using the proposed PI gains as a general base line. By evaluating the cost function of the default gains provided by the challenge it was able to determine the general idea of the optimization bounds, and the physical matrix. In the end it was possible to configure the spMODEx with the following parameters:

Strategy: Spherical Pruning

Norm: Physical with the following matrix:

0.0	50	70	90	100	200
0.0	1	1.5	2	2.5	10 (11)
0.0	1	1.7	2.0	2.5	43

Once configured the algorithm was run in a desktop computer with the following specifications: CPU: Intel(R) Core(TM) i7-4790K CPU @ 4.00GHz 4.00 GHz, GPU: NVIDIA GeForce GTX 970, RAM: 16.0GB, MATLAB version: 2024a.

3.2. PI gains

Initially spMODEx, was configured with the following optimization bounds: Kp in the range of -12 to near zero, and Ti in the range of 12 to 36. By testing each Kp, and Ti generated by the algorithm in the benchmark script provided by the challenge it was able to notice that most of the better scores were found in the range of -8 to -5. Therefore, the optimization bounds for the multi-objective algorithm were changed to reflect the better scores. The following Pareto front is the result of the multi-objective algorithm with the new optimization bounds. It can be observed that in algorithm produced a total of 18 gains in the optimization bounds.

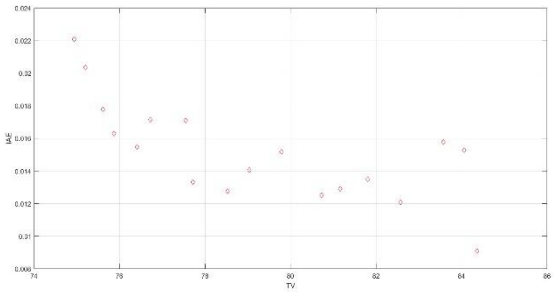

Figure 2: Resulting Pareto Front for PI gains

Each gain was tested individually in the benchmark script, the gains found in the middle of the front performed better than the gains that were in both extremes of the front. The table below contains the gains found at both extremes of the graph, and the best performing gain in the middle.

Table 1: PI gains and their respective performance.

Kp	Ti	Ki	JTotal
-8.0000	20.0000	-0.4	0.5369
-7.4169	25.2306	-0.29396	0.5214
-5.0000	24.8796	-0.20096	0.5668

Even though the best gains for the PI found by the algorithm had a slightly larger JTotal score, since it was used a multi-model approach, it is expected to be able to be used in a wider array of circumstances in comparison to the default gains.

3.3. PID gains

Based on the results, of the PI gains, the multi objective algorithm's optimizations bound was changed to search for PID gains to test their performance. Initially the optimization bounds were defined as: Kp = -8 to -5, Ti = 12 to 36, and Td = 0 to 1. The algorithm generated a total of 21 possibles gains, as it can be seen in the Pareto front below.

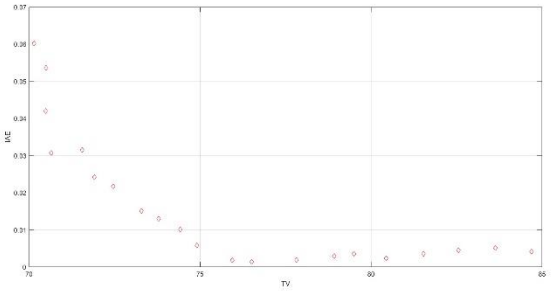

Figure 3: Resulting Pareto Front for PID gains

In this context, a PID structure outperformed the PI structure. With its best gain having a JTotal score of 0.4940, having a better performance than the default PI proposed in this challenge. The following table shows the gains, in the extremes of the pareto front, and in the middle of the front. Unlike the PI, the best gain was found in the upper extreme of the front. Therefore, it can be presumed that in the context of the PID gains, the lower the TV – Total input Variation is more critical than the IAE – Integral Absolute Error.

Table 2: PID gains and their respective performance.

Kp	Ti	Td	JTotal
-8.0000	12.0000	0.9754	0.4940
-8.0000	21.0363	0.3236	0.5136
-5.0000	29.5318	0.2519	0.5752

To achieve a better result, with the PID structure a derivative filter was applied in the Simulink PID block. With the filter with a value of 10, the JTotal went from 0.4940 to 0.4602 The graphs below show the response of the system with a PID controller in a closed loop.

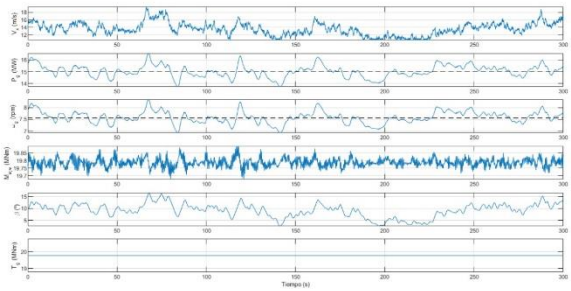

Figure 4: Closed Loop response of the system with the PID controller

Therefore, the best result found was a PID controller with a filter with the following parameters: Kp = -8.0000, Ti = 12.0000, Td = 0.9754, filter = 10. Beyond outperforming the proposed PI in the challenge, due to the fact that a multi-

model approach can be used, it can be expected that this controller can be used in a wider variety of models when compared to that of the proposed PI.

4. Conclusion

This scientific study presents a proposition about a controller for a 15MW wind generator. After a series of tests using models found in the System Identification toolbox, with a multi-model and multi-objective approach, a PID controller was developed that can effectively manage the proposed turbine model within the specifications published by the University of Córdoba. Given that the controller proposed by the University had a $J(CP)$ of 0.5206, and the controller proposed in this work has a $J(CP)$ equal to 0.4602 it can be considered an optimal controller for the process. Furthermore, graphs using Open Fast showed that the power of the wind turbine and the M_{eje} parameter had a good response, considering the overshoots of the wind velocity over time. As a future direction for this research, the evaluation and implementation of nonlinear models such as NARMAX and Hammerstein-Wiener, along with Gaussian processes, to estimate nonlinear systems and subsequently estimate a transfer function of the same dynamic process (wind turbine) or a different process, is proposed (Vance, et al., 2017). This research could involve applying Gaussian process techniques in system identification to capture uncertainty and variability in the system data, thereby enhancing estimation accuracy and robustness. Furthermore, specific application cases where uncertainty and variability are critical, such as in process control systems in variable environments (utilizing the multi-objective concept), could be investigated. This study highlights a multipurpose and integrated approach that could provide a more comprehensive and accurate representation of nonlinear system behavior, thereby enhancing process design and control across various industrial and scientific fields.

5. Acknowledgements

This work was partially supported by the grant CNPq Multiobjective Computational Intelligence PQ2/310195/2022-5, and the “projeto universal” 408164/2021-2.

6. References

Estimate parameters of ARX, ARIX, AR, or ARI model - MATLAB arx - MathWorks América Latina. (s/f). Mathworks.com. Recuperado el 25 de mayo de 2024, de https://la.mathworks.com/help/ident/ref/arx.html

Ljung, L. System Identification: Theory for the User, Second Edition. Upper Saddle River, NJ: Prentice-Hall PTR, 1999. See chapter about computing the estimate.

V. Huilcapi, X. Blasco, J. M. Herrero and G. Reynoso-Meza, "A Loop Pairing Method for Multivariable Control Systems Under a Multi-Objective Optimization Approach," in IEEE Access, vol. 7, pp. 81994-82014, 2019, doi: 10.1109/ACCESS.2019.2923654.MATH

Ackermann, Thomas, ed. *Wind power in power systems*. John Wiley & Sons, 2012.

GARRIDO, Juan, et al. Concurso en Ingeniería de Control 2024: Control de una turbina eólica de gran potencia II. Dpto. Ingeniería de Control y Automática, Universidad de Córdoba, 2024.

Zhang, P. (2010). Advanced Industrial Control Technology. Nova York.

Vinnicombe, G. (1993). Measuring Robustness of Feedback Systems. PhD thesis, University of Cambridge, Dept. of Engineering.

Akaike, H. (1974). "A new look at the statistical model identification". IEEE Transactions on Automatic Control, 19(6), 716-723.

Burnham, K. P., & Anderson, D. R. (2002). "Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach". Springer.

J. Chen, K. H. Lundberg, D. E. Davison and D. S. Bernstein, "The Final Value Theorem Revisited - Infinite Limits and Irrational Functions," in IEEE Control Systems Magazine, vol. 27, no. 3, pp. 97-99, June 2007, doi: 10.1109/MCS.2007.365008.

HAU, Erich. Wind Turbines: Fundamentals, Technologies, Application, Economics. 3. Ed. Munique: Springler, 2013.

Vance, P. J., Das, G. P., Kerr, D., Coleman, S. A., McGinnity, T. M., Gollisch, T., & Liu, J. K. (2017). Bioinspired approach to modeling retinal ganglion cells using system identification techniques. IEEE Transactions on Neural Networks and Learning Systems, 29(5), 1796-1808.

Gilberto Reynoso-Meza (2024). eXtended Multi-objective Differential Evolution with Spherical Pruning, < spMODEx > algorithm (https://www.mathworks.com/matlabcentral/fileexchange/65145-extended-multi-objective-differential-evolution-with-spherical-pruning-spmodex-algorithm), MATLAB Central File Exchange. Retrieved May 30, 2024