Adaptive filters for DC motor identification: a case study

Simona-Daiana Sim
Department of Automation
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
simona.sim@aut.utcluj.ro

Attila Medgyesi
Department of Automation
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
medgyesi.is.attila@student.utcluj.ro

Zsófia Lendek
Department of Automation
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
zsofia.lendek@aut.utcluj.ro

Mircea Susca
Department of Automation
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
mircea.susca@aut.utcluj.ro

Vlad Mihaly
Department of Automation
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
vlad.mihaly@aut.utcluj.ro

Doina Pisla
CESTER
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
doina.pisla@mep.utcluj.ro

Calin Liviu Vaida
CESTER
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
calin.vaida@mep.utcluj.ro

Petru Dobra
Department of Automation
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
petru.dobra@aut.utcluj.ro

Anca Elena Pica
Department of Automation
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
anca.pica@campus.utcluj.ro

***Abstract*— This paper investigates the practical application and evaluation of four adaptive algorithms: Least Mean Square (LMS), Normalized Least Mean Square (NLMS), Recursive Least Square (RLS) and Affine Projection (AP) for the identification of a DC motor. The work focuses on dynamic system identification, where the adaptability and efficiency of numerical filters play an important role. To identify the parameters of a DC motor, these adaptive algorithms are used on measured data. With the aim of having a shorter processing time, the data was downsampled. Satisfactory results were obtained with all algorithms. Finally, the difficulty of implementing the algorithms, the errors obtained and the processing time were analyzed.**

Keywords— adaptive filters, motor identification, testing

I. Introduction

In the rapidly evolving landscape of technology and automation, the demand for efficient and adaptable systems has become significant. One area where this need is particularly pronounced is signal processing [1], where the ability to extract meaningful information from data is crucial. In this context, the application of numerical filters [2], especially adaptive filters [3], [4] has gained importance.

Numerical filters [5], [6] are used in various applications ranging from audio processing [7], and telecommunications [8] to biomedical signal analysis [9]. These filters enhance signal quality [10], reduce noise [11], and extract relevant features from complex datasets [12]. One specific application is the representation of the impulse response of a process [13, 14, 15, 16] by FIR filter coefficients. This can be considered as an identification method [17].

Adaptive filters [18] self-adjust and optimize their parameters based on the characteristics of the input data. This makes them well-suited for dynamic and changing environments [19], where traditional fixed filters may fail. The application of adaptive filters extends across fields [20], [21] such as communication systems [8], control systems [10], and, notably, identification of dynamic systems.

Understanding and modeling the behavior of complex systems [22], [23] is essential for optimizing their performance. Adaptive algorithms such as [17] Least Mean Squares (LMS), Normalized LMS (NLMS), Affine Projection (AP), and Recursive Least Squares (RLS), offer powerful tools for identifying dynamic systems.

The main idea in adaptive filters is to adjust the filter coefficients iteratively to minimize the error between the estimated output and the actual output. This ensures [4] that the filters respond to the evolving characteristics of the system under observation.

This article applies the above-mentioned adaptive algorithms to identify the model of a motor. Besides the identification of the FIR filter model, the paper highlights the choice of the appropriate adaptive algorithm, its impact on processing time and the accuracy of the result. It also discusses the impact of downsampling of the input signal in order to reduce processing time. The identification process starts with the collection by a dedicated development board of real-time input-output data from a direct current motor. The acquired data undergoes preprocessing, where downsampling is applied. Downsampling [5], [24], or the selective reduction of data is a critical step to realize the computational efficiency required for adaptive filtering processes.

We test how well each algorithm captures DC motor behavior and the effect of downsampling. By comparing predicted and actual outputs, we evaluate the algorithm's adaptability and robustness. This serves as a practical benchmark for assessing performance. We also evaluate the processing time and identification error for both original and downsampled data for practical implementation.

The central focus is on determining the algorithm that provides the most accurate and reliable results. We consider both theoretical robustness and real-world feasibility. Thus,

This research was supported by the project New frontiers in adaptive modular robotics for patient-centered medical rehabilitation–ASKLEPIOS, funded by European Union – NextGenerationEU and Romanian Government, under National Recovery and Resilience Plan for Romania, contract no. 760071/23.05.2023, code CF 121/15.11.2022, with Romanian Ministry of Research, Innovation and Digitalization, within Component 9, investment I8.

we contribute to the field of adaptive filtering, providing guidelines in selecting algorithms that show high performance and easy implementation.

The paper is structured as follows: in Section II we describe aspects related of the DC motor, the development board and the acquisition of input-output data. In Section III, four adaptive algorithms (LMS, NLMS, AP, RLS) are described. The results are presented in Section IV. Section V presents some conclusions and starting points for future developments.

II. Setup

Our case study involves identifying the direct current motor in Fig.1. We use the Nucleo64-P development board for data acquisition.

A. The Motor

The motor to be identified is a direct current motor (see Fig. 1). The schematic diagram is presented in Fig. 2.

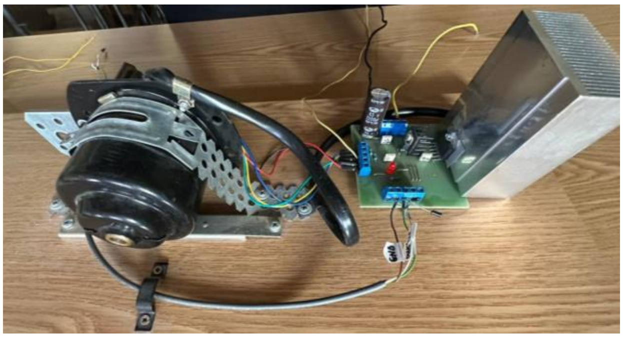

Fig. 1. The motor used for identification

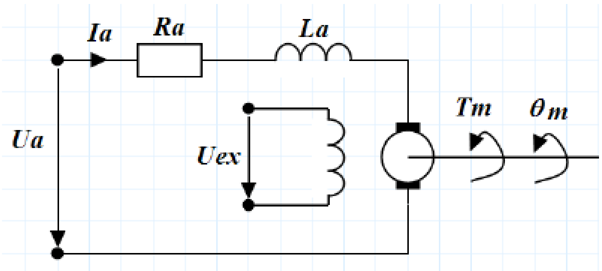

Fig. 2. The structure of the motor

Based on [25], the state-space model of the motor in continuous time is:

$$\begin{pmatrix} \frac{di_a(t)}{dt} \\ \frac{d\omega_m(t)}{dt} \\ \frac{d\theta_m(t)}{dt} \end{pmatrix} = \begin{pmatrix} \frac{-R_a}{L_a} & \frac{-K_e}{L_a} & 0 \\ \frac{K_t}{J_m} & \frac{-D_m}{J_m} & 0 \\ 0 & \omega_m & 0 \end{pmatrix} \begin{pmatrix} i_a \\ \omega_m \\ \theta_m \end{pmatrix} + \begin{pmatrix} \frac{L_a}{L_a} \\ 0 \\ 0 \end{pmatrix} u_a(t) \qquad (1)$$

where the variables are: $i_a(t)$ is the rotor current, $\omega_m(t)$ is the angular velocity and $\theta_m(t)$ is the angle; R_a is armature resistance, L_a is armature inductance, K_e proportionality constant, K_t is the motor torque constant, J_m is the equivalent moment of inertia of the motor, D_m is the damping of the engine (including both its own viscous damping and the viscous damping of the load).

The motor parameters are not known and they have to be identified.

B. The Development Board

The STM32 Nucleo 64-P development board [26] was used to generate the SPAB input signal and to acquire data from the motor. The board (see Fig. 3) is a compact option microcontroller, offering a 64-pin form factor for easy prototyping and development of embedded systems. It has a maximum frequency of 80kHz and contains both an ADC (Analog to Digital Converter) and a DAC (Digital to Analog Converter). Also, it has a versatile set of peripherals, an Arduino Uno R3 connector for expansion, and supports a wide range of IDEs, making it a robust choice for developing embedded applications.

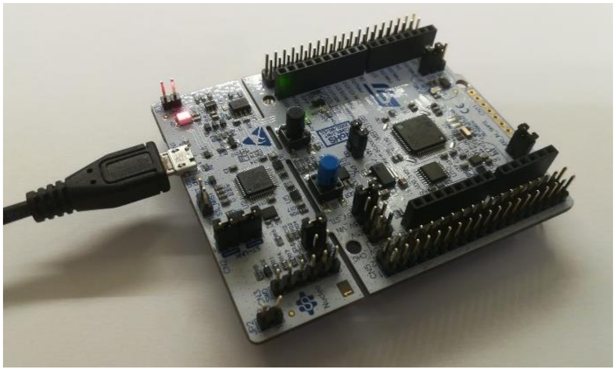

Fig. 3. The development board used for data acquisition

C. Data Acquisition

Ensuring that the signal has a wide frequency spectrum is a crucial point when choosing the input signal. This means that if a system contains several elements, some of them will operate at a certain frequency, while others will operate at a different one. We must select a signal that will activate all the elements in the system. Another important issue is to prevent saturation. Therefore, the input signal has been chosen with a period so that the motor can reach its response time in the largest period, but also respond in the smallest period of the input signal.

We apply several inputs to our system and record the output using the STM32 Nucleo 64P microcontroller. The input signal is generated by the development board, which is connected with Matlab. Both input and output are saved in a file using Matlab. The setup is presented in Fig. 4. Both the SPAB type input and the motor output were acquired in real time and can be seen in Fig. 5. The sampling period is $T_s = 1ms$.

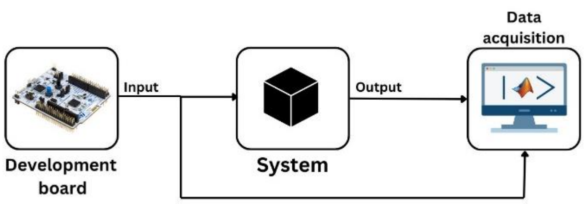

Fig. 4. Hardware setup for data acquisition

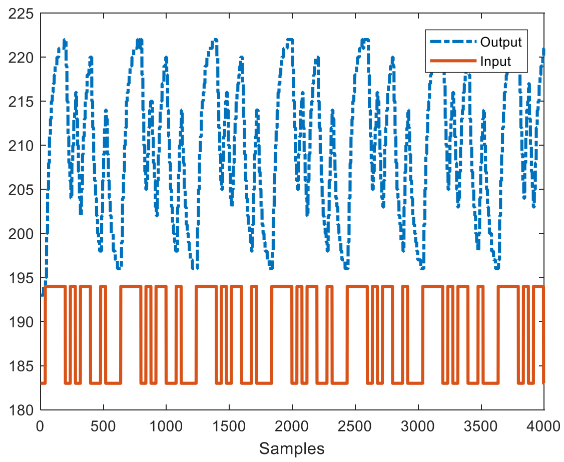

Fig. 5. Acquired signals

Downsampling is achieved by discarding certain samples. With the goal of implementation on an embedded system, downsampling is used to reduce the computation time. Step 7 was chosen for downsampling, obtained through several trials in order to obtain a satisfactory result (with an approximation of over 80%), and the comparison can be seen in Fig. 6.

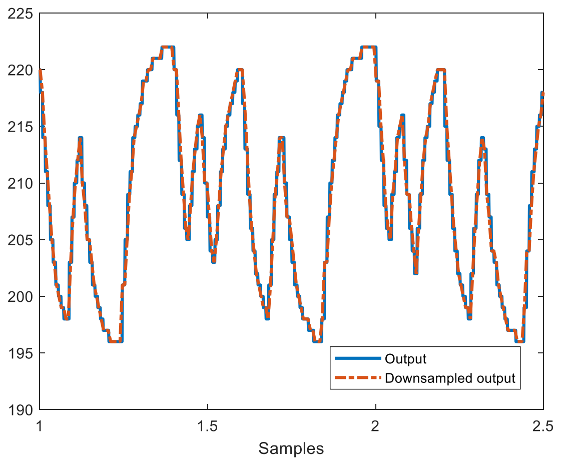

Fig. 6. Original and dowsampled output signal

III. Adaptive Filters

Our goal is to accurately identify the process in the form of an FIR (Finite Impulse Response) filter. In this section, four adaptive filtering methods are explored: LMS, NLMS, AP, and RLS, which return an imposed number of coefficients of the FIR filter.

The identification structure is shown in Fig. 7, where the adaptive filter is placed in parallel with the unknown system to be identified.

In our case, the motor output to an impulse input is represented by the Markov coefficients or weight vector $w(n)$ which are also the FIR filter coefficients. The FIR filter computes the output $y(n)$ as (2), where $w(k)$, $k = 0, \dots, M$ are the Markov coefficients, M denotes the order of the filter and $u(k)$ represents the input at sample k.

$$y(n) = \sum_{k=0}^{M} w(k)u(n-k) \tag{2}$$

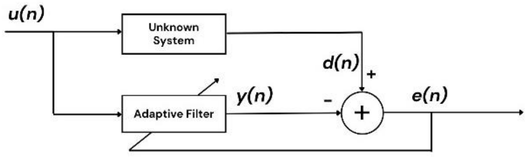

Fig. 7. Block diagram of adaptive system identification

An adaptive algorithm [17] is a set of recursive equations used to automatically adjust the weight vector $W = [w(0)\ w(1) \dots w(M)]^T$ at time n to minimize the error signal $e(n)$ so that the weight vector iteratively converges to the optimal solution.

The FIR filter is, in fact, a linear model that approximates the unknown system. The excitation signal *u(n)* at time-step n serves as input to both the unknown system and the adaptive filter, while *η(n)* represents the disturbance affecting the unknown system.

The objective is to model the unknown system so that the filter's output *y(n)* closely matches the unknown system output's *d(n)*. This can be achieved by, e.g., minimizing the square sum of the difference between the output of the model and of the filter.

$$J = \sum_{k=0}^{\infty} (d(k) - y(k))^2 \tag{3}$$

The error $e(n)$ is defined as the difference between the physical response $d(n)$ and the model response $y(n)$.

$$e(n) = d(n) - y(n) \tag{4}$$

The Least Mean Squares (LMS) algorithm [17] is a widely employed adaptive filtering method known for its simplicity and computational efficiency, as it operates on the principle of gradient descent. LMS is particularly suited for applications where real-time adaptation to changing conditions is crucial.

The LMS algorithm adjust the vector of the weights $W = [w(0)\ \ w(1)\ \dots\ w(M)]^T$ at each time-step n based on the last error $e(n)$ and the vector $U(n) = [u(n)\ u(n-1) \dots u(n-M+1)]$ containing the inputs that have been used to obtain the corresponding output of the plant and of the model, respectively. The weights are updated as:

$$W(n+1) = W(n) + \mu U(n)e(n) \tag{5}$$

where $\mu \in [0,2]$ is the step size (or convergence factor) that determines the stability and the convergence rate of the algorithm.

The normalized Least Mean Squares (NLMS) [17] normalizes the update with the power of the input signal. The weight update equation for NLMS is expressed as (6):

$$W(n+1) = W(n) + \frac{\mu}{P_u(n) + \alpha} U(n)e(n) \tag{6}$$

where μ is the convergence rate, P_u is the power of the input vector and α is a small value between 0 and 1 to avoid division by zero.

The power $P_u(n)$ of the input sequence can be iteratively estimated [17] for each time-step n as:

$$P_u(n) = (1-\beta)P_u(n-1) + \beta u^2(n) \quad (7)$$

where $0 < \beta < 1$ is a forgetting factor.

This normalization makes it able to handle signals with varying amplitudes.

Recursive Least Squares (RLS) [17] uses an estimate of the inverse covariance matrix of the input, to adapt to changes in the input signal statistics and is therefore well-suited for applications requiring high accuracy and efficiency. The weights are updated as (8) and the updating gain $g(n)$ is calculated using (9-11).

$$W(n+1) = W(n) + g(n)e(n) \quad (8)$$

$$g(n) = \frac{r(n)}{1 + u^T(n)r(n)} \quad (9)$$

$$r(n) = \lambda^{-1}P(n-1)u(n) \quad (10)$$

$$P(n) = \lambda^{-1}P(n-1) - g(n)r^T(n) \quad (11)$$

In (10-14) $P(n)$is the cross-correlation matrix and λ is the forgetting factor.

Affine Projection (AP) [17] employs a more intricate mechanism, taking advantage of past input signals to account for colored input noise. The AP algorithm uses a set of v constraints of the form $d(n-k) = W^T(n+1)U(n-k)$ for $k = 0, 1, \ldots, v-1$.

The error vector $E(n) = [e(n), e(n-1), \ldots, e(n-M+1)]$ is calculated as:

$$E(n) = D(n) - A(n)W(n) \quad (12)$$

where input signal matrix $A^T(n) = [U(n), U(n-1), \ldots, U(n-v+1)]$ consists of v columns of input vectors of length M, $U(n) = [u(n), u(n-1), \ldots, u(n-M+1)]^T$ and $D(n) = [d(n), d(n-1), \ldots, d(n-v+1)]^T$ is the desired response vector.

The weights are updated as:

$$W(n+1) = W(n) + \mu A^T(n)[A(n)A^T(n)]^{-1}e(n) \quad (13)$$

Next, we apply these filters to identify the motor.

IV. Results

The algorithms from Section III have been implemented and tested in Matlab using data collected from the motor. Several tests were made to choose a number of coefficients that is reasonable in terms of computational cost and for which all four algorithms respond favorably. A number of 55 coefficients was chosen, but depending on the algorithm and its characteristics, more satisfactory responses can be found. The acquired data were divided into two parts: one for obtaining the filter coefficients, and the second part of the data was used for testing. In what follows, the results obtained on the test data will be presented.

To evaluate the filters, the average squared error (ASE) was used:

$$ASE = \frac{\sum_{k=0}^{L}(d(k) - y(k))^2}{L} \quad (14)$$

where $d(k)$ is motor output (desired signal), $y(k)$ is the filter output and L is the length of the data.

For the LMS algorithm, with $M = 55$ terms, $\mu = 10^{-6}$, the result presented in Fig. 8 was obtained and an $ASE = 5.4279$. The total processing time obtained with downsampling is $0.0017\ sec$, $5.5033\ \mu s/iteration$. Without downsampling, a total time of $0.00397\ sec$, $1.9419\ \mu s/iteration$ was obtained.

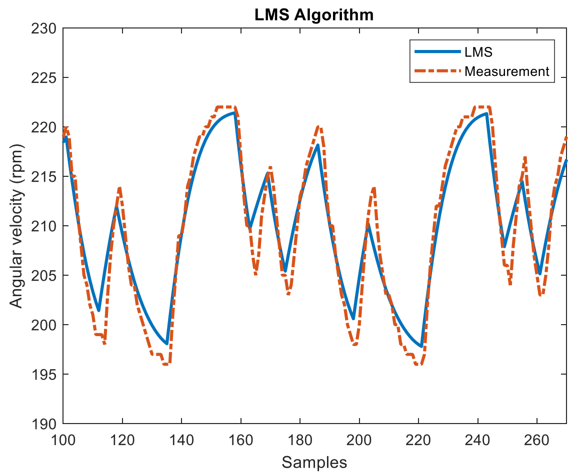

Fig. 8. Result obtained with LMS algorithm

For the NLMS algorithm, with $M = 55$ terms, $\mu = 1$ and $\alpha = 10^{-5}$, the result presented in Fig. 9 was obtained and an $ASE = 12.5618$. The total processing time obtained with downsampling is $0.002\ sec$, $6.7240\ \mu s/iteration$ respectively. Without downsampling, a total time of $0.005\ sec$ and $2.4912\ \mu s/iteration$ was obtained.

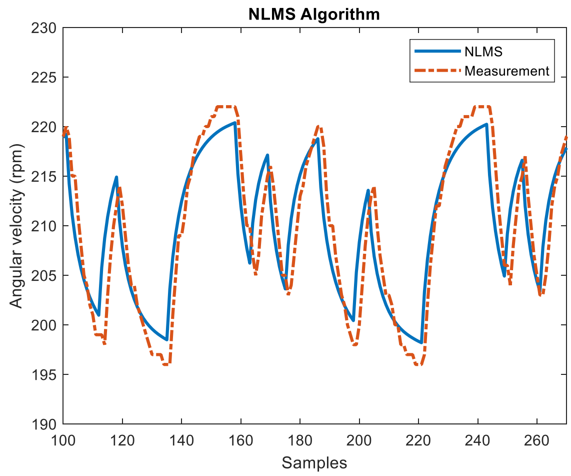

Fig. 9. Result obtained with NLMS algorithm

For the RLS algorithm, with $M = 55$ terms, $\lambda^{-1} = 1.03$, the result presented in Fig. 10 was obtained and an $ASE = 1.6331$. The total processing time obtained with downsampling is $0.0079\ sec$, $26.429\ \mu s/iteration$. Without downsampling, a total time of $0.0466\ sec$, $23.307\ \mu s/iteration$ was obtained.

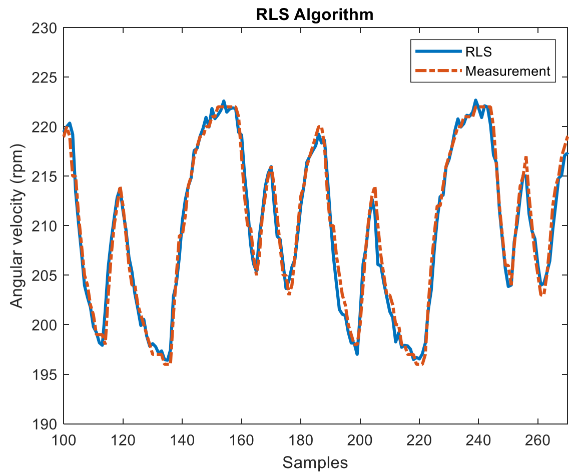

Fig. 10. Result obtained with RLS algorithm

For the AP algorithm, with $M = 55$ terms, the result presented in Fig. 11 was obtained and a ASE = 16.5701. The total processing time obtained with downsampling is 0.0057 sec, 19.008 $\mu s/iteration$ and for the version. Without downsampling, a total time of 0.0309 sec, 15.451 $\mu s/iteration$ was obtained.

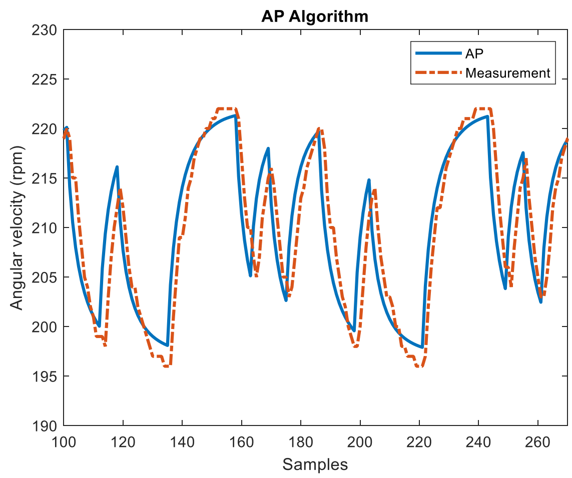

Fig. 11. Result obtained with AP algorithm

To summarize the processing times and the errors obtained, the results are compared in Table 1.

TABLE I. ALGORITHM COMPARISONS

	ASE	**Time with downsampling**	**Time without downsampling**
LMS	5.4279	0.0017 sec	0.00397 sec
NLMS	12.5618	0.002 sec	0.005 sec
AP	16.5701	0.0057 sec	0.0309 sec
RLS	1.6331	0.0079 sec	0.0466 sec

The fact that the evaluation time on original data is lower is also due to issues such as: computationally the first part of a code generally takes a long time, for downsampled data there are fewer iterations, which is reflected in the total processing time, in parallel other programs are run, etc.

For testing, the algorithms were also implemented using the same parameters on the original data, without downsampling. The result was subsequently downsampled and compared with those obtained above. The results of the ASE errors between the original dowmsampled filter and downsampled filter are: 3.4599 for LMS, 4.3289 for NLMS, 5.0927 for AP and 3.6549 for RLS.

We have a small mismatch between the experimental data and the model. Nonlinearity in the system behavior is the first possible cause. The disturbance, which is a significant source of error in the system output or in the measurements, also impacts the model's accuracy.

We can consider all the results satisfactory but depending on the requirements of the application and available resources different algorithms may be chosen.

The algorithm with the best accuracy, which can be vital in certain circumstances, is RLS, but it is also the one with the longest processing time, due to its complexity. The fastest algorithm was LMS, which performed surprisingly well. The last ones in terms of error were the NLMS and AP algorithms. They had an average processing time. A compromise choice between processing time and accuracy is the LMS algorithm, and, if processing time is not a problem, then the RLS algorithm. Downsampling is useful when we have a processing time limitation, or we have a signal on which periodic noise is overlaid. In our case downsampling improved the processing time and led to satisfactory results with a small loss of the accuracy of the results.

V. CONCLUSIONS

Within this article, we tested several algorithms to identify a DC motor. After acquiring input and output data using the STM32 development board and downsampling the data for processing efficiency, four adaptive algorithms—LMS, NLMS, AP, and RLS—were applied, each using 55 Markov coefficients. The LMS algorithm, although simple to implement and process, produced reasonably good results. On the other hand, its normalized variant, NLMS, showed more modest performance. RLS, being the most complex, provided the best approximation, whereas AP exhibited weaker outcomes. In the end, the best algorithm is chosen depending on the project requirements, application specifics, and computational resources. There is also a difference of at least 0.005 seconds between the algorithms run on downsampled and original data. This difference can be significant when these algorithms run in real time on the development board. It was also noted that the LMS algorithm is the fastest and the RLS the slowest, both having very good results in terms of error. The choice between them depends on the requirement with a higher priority: accuracy versus processing speed.

A promising future research direction involves advanced methodologies to extract system models through the Markov coefficients inherent in Finite Impulse Response filters.

Another direction for further development could involve exploring the identification of an adaptive model for control in rehabilitation medicine. This approach could open up new perspectives for the development of healthcare systems, with the potential to optimize and tailor rehabilitation treatments to the specific needs of individual patients.

References

[1] L. R. Rabiner and B. Gold, Theory and application of digital signal processing, Prentice-Hall, 1975.

[2] B. Widrow and S. D. Stearns, Adaptive Signal Processing, Prentice Hall, 1985.

[3] A. H. Sayed, Fundamentals of Adaptive Filtering, New York: John Wiley & Sons, Inc, 2003.

[4] S. Haykin, Adaptive Filter Theory, Inc. Prentice-Hall, 1996.

[5] J. G. Proakis and D. G. Manolakis, Digital Signal Processing, Third Edition ed., Prentice-Hall International INC, 1996.

[6] M. Cherniakov, An Introduction to Parametric Digital Filters and Oscillators, Wiley, 2003.

[7] S. M. Kuo and D. R. Morgan, Active Noise Control Systems: Algorithms and DSP Implementations, New York: John Wiley & Sons, Inc., 1996.

[8] F. Balestro, A. Chianale, G. Privat, M. S. Tawfik, I. Vandeweerd and A. D. Wittmann, "Design of digital filters for advanced telecommunications ASIC's using a special-purpose silicon compiler," *IEEE Journal of Solid-State Circuits,* vol. 26(7), pp. 1047-1055, 1991.

[9] E. Ozpolat, B. Karakaya, T. Kaya and A. Gulten, "FPGA-based digital Filter Design for Biomedical Signal," in *2016 XII International Conference on Perspective Technologies and Methods in MEMS Design (MEMSTECH)*, Lviv, Ukraine, 2016.

[10] E. B. Solovyeva, "Nonlinear models of digital filters and their application fields," in *2016 XV International Symposium Problems of Redundancy in Information and Control Systems (REDUNDANCY)*, St. Petersburg, Russia, 2016.

[11] E. Hansler and G. Schmidt, Acoustic Echo and Noise Control: A Practical Approach, New York: John Wiley & Sons, Inc, 2004.

[12] P. L. de Leon and D. M. Etter, "Experimental results with increased bandwidth analysis ´," *IEEE Signal Processing Lett,* vol. 2(1), pp. 1-3, January 1995.

[13] A. K. Tangirala, Principles of System Identification: Theory and Practice, CRC Press, 2014.

[14] M. Zidane, S. Said, M. Sabri and A. Boumezzough, "Impulse Response Identification of Minimum and Non Minimum Phase Channels," *International Journal of Advanced Science and Technology,* vol. 64, pp. 59-72, 2014.

[15] T. Münker, J. Belz and O. Nelles, "Improved Incorporation of Prior Knowledge for Regularized FIR Model Identification," in *Annual American Control Conference (ACC),* Milwaukee, WI, USA, 2018.

[16] D. Westwick and E. Perreault, "Unbiased Identification of Finite Impulse Response Linear Systems Operating in Closed-Loop," in *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, New York, NY, USA, 2006.

[17] K. A. Lee, W. S. Gan and S. M. Kuo, Subband adaptive filtering, John Wiley & Sons, 2009.

[18] B. Farhang-Boroujeny, Adaptive Filters: Theory and Applications, New York: John Wiley & Sons, Inc., 1998.

[19] D. Kutluyıl, Partial-Update Adaptive Signal Processing. Design, Analysis and Implementation, Academic Press, 2009.

[20] J. R. Treichler, C. R. Johnson and M. G. Larimore, Theory and Design of Adaptive Filters, Prentice Hall, 2001.

[21] S.-D. Sim, Zs. Lendek and P. Dobra, "Implementation and testing of digital filters on STM32 Nucleo-64P," in *IEEE International Conference on Automation, Quality and Testing, Robotics*, Cluj-Napoca, Romania, 2022.

[22] K. J. Keesman, System Identification: An Introduction, Springer, 2011.

[23] B. Wise and N. Ricker, "Identification of finite impulse response models by principal components regression: Frequency-response properties," *Process Control and Quality,* vol. 4, pp. 77-86, 1992.

[24] D. Williamson, Discrete-time Signal Processing, London: Springer London Ltd, 1999.

[25] P. Dobra, "Curs Modelarea Proceselor," UTCN, Cluj-Napoca, 2018.

[26] st.com, "STM32 Nucleo-64 development board with STM32L45RC MCU, SMPS, supports Arduino, ST Morpho connectivity," [Online]. Available: https://www.st.com/en/evaluation-tools/nucleo-l433rc-p.html.

[27] J. Ford, "Online identification of non-minimum phase finite impulse responselinear systems with discrete inputs via hidden Markov models," in *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No.00CH37187)*, Sydney, NSW, Australia, 2000.

[28] I.-D. Fîciu, J. Benesty, L. Dogariu, C. Paleologu and S. Ciochina, "Efficient Algorithms for Linear System Identification with Particular Symmetric Filters," *Applied Sciences*, vol. 12, no. 9, pp. 42-63, 2022.