Experimental Evaluation of Horner's Method for CPU Energy Reduction in Nonlinear Modelling

Jordan Browne
Department of Electronic Engineering
Maynooth University
Maynooth, Ireland
jordan.browne.2022@mumail.ie

Thalita Nazaré
Centre for Ocean Energy Research
Department of Electronic Engineering
Maynooth University, Ireland
thalita.nazare.2023@mumail.ie

Erivelton Nepumecuno
Centre for Ocean Energy Research
Department of Electronic Engineering
Maynooth University, Ireland
erivelton.nepomuceno@mu.ie

2024 35th Irish Signals and Systems Conference (ISSC) | 979-8-3503-5298-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISSC61953.2024.10603111

***Abstract*—Computer applications have played a central role in human progress over the past few decades. Their increasing prevalence has raised concerns about high energy consumption, drawing the attention of many scientists. Numerous studies have been conducted to develop techniques aimed at enhancing the efficiency of algorithms. Many of these studies evaluate performance based on computational time, which is measured by the elapsed time between two points in the algorithm. In this work, we present an experimental evaluation of energy reduction that reveals the computational time metric may overestimate the benefits of code efficiency. We have employed Horner's method to improve the code efficiency of two identified polynomial Nonlinear AutoRegressive Moving Average with eXogenous inputs (NARMAX) models for the Lorenz Attractor and Mackey-Glass systems. To directly measure Central Processing Unit (CPU) energy consumption, we utilised a Power Measurement Device. Our findings indicate that, while commonly used indicators such as simulation time and CPU utilisation are informative, they may not fully capture the complete picture of power usage.**

***Index Terms*—Nonlinear Dynamical Systems, Green Algorithm, Horner's method, Sustainable Circuits and Systems, Computer Arithmetic, Computer Simulation.**

I. Introduction

The growing need for computational power within the Information and Communication Technology (ICT) industry is inherently linked to higher levels of energy consumption, carbon footprint, and subsequent environmental consequences [1]–[3]. Efforts to address these impacts have primarily focused on improving the efficiency of hardware and software systems, employing techniques such as dynamic voltage frequency scaling and dynamic power management [4]–[8].

In the field of sustainable computing, the primary emphasis has often been on optimising algorithms to reduce computational complexity and execution time [7], [8]. This optimisation is valuable for assessing code efficiency, with computation time widely used as a key performance indicator [9]. For example, authors in [10] introduced the Green-box System Identification approach, applying multi-objective optimisation to create sustainable computing frameworks, highlighting the balance between computational complexity and environmental impact. Gómez-Carmona et al. presented an approach that simplifies the complexity of supervised learning algorithms at the Edge, significantly reducing computational cost and promoting efficient and sustainable classification solutions [11]. In another approach, central processing unit (CPU) execution time was measured for evaluating a modelling technique using `tic` and `toc` commands in MATLAB [12].

Nonetheless, in many applications, especially those with energy constraints, what's more crucial than computation time is a direct measure of energy expended or energy reduction achieved by a novel technique [13]. Additionally, computational time measurements can be influenced by various other ongoing processes on the computer. Therefore, there is a significant gap in current research, notably the absence of a direct method for measuring CPU energy consumption to evaluate the efficiency of numerical methods in reducing simulation time and complexity.

In an effort to contribute to the academic discussion in this field, and inspired by the findings from [14] which demonstrated a significant reduction in the carbon footprint of computational processes, this paper investigates the efficacy of using Horner's approach in MATLAB for the evaluation of NARMAX models [15]–[17]. This work introduces an experimental evaluation of energy reduction on the CPU using a Power Measurement Device (PMD). It has been observed that commonly used indicators such as simulation time and CPU utilisation, while informative, may not fully capture the complete picture of power usage. The results of our research indicate that computational time may overestimate the benefits of code efficiency obtained through Horner's method. The method is applied in two case studies: the computer simulation of the Lorenz Attractor and Mackey-Glass identified models. These findings suggest that our analysis represents an important step toward gaining a more thorough understanding of the energy dynamics associated with computer simulations of nonlinear dynamical systems.

The article is structured as follows: the initial section II provide a comprehensive background on the theoretical concepts essential for understanding the study. This is followed by a detailed description of the methodology, including the experimental setup and procedures used for measuring CPU energy consumption. The section IV presents the findings on the impact of the applied methods on energy efficiency. Finally, the article concludes with a discussion of the implications of these findings and suggestions for future research in the field of energy-efficient computing.

II. Background

In this section, the basic background of this paper is presented. Polynomial NARMAX, the Horner's method and CPU power are briefly described.

A. Polynomial NARMAX

A polynomial NARMAX (Nonlinear Auto Regressive Moving Average model with eXogenous input) can be defined as [15]:

$$\begin{aligned} y_k &= F^{\ell}[y_{k\text{-}1}, \ldots, y_{k\text{-}n_y}, u_{k\text{-}1}, \ldots, u_{k\text{-}n_u}, \\ &\quad e_{k\text{-}1}, \ldots, e_{k\text{-}n_e}], \end{aligned} \tag{1}$$

in which y_k, u_k and e_k are, respectively, the model output, input and noise terms at time $k \in \mathbb{N}$. The parameters n_y, n_u and n_e are the maximum lag for output, input and noise terms. Terms of e_k are constantly used during the parameter estimation process to avoid bias. In this work, F^{ℓ} is assumed to be a polynomial with non-linearity degree $\ell \in \mathbb{Z}^+$.

Nonlinear systems are usually modelled using particular cases of the polynomial NARMAX, such as the polynomial nonlinear autoregressive exogenous (NARX) or nonlinear autoregressive (NAR). In the former case, noise terms are not used, whereas the polynomial NAR is used to represent autonomous systems in which the input is not used. In this paper, we use the acronym NARMAX to name all of them.

B. Horner's Method

Consider the following polynomial:

$$P(x) = a_k x^n + a_{k-1} x^{n-1} + \ldots + a_2 x^2 + a_1 x + a_0 \tag{2}$$

where $a_{k=1:n}$ are the coefficients to the variable $x^{k=1:n}$ in the polynomial $p(x)$. The Horner's method is an algebraic manipulation that represents (2) as:

$$P(x) = (...((a_n x + a_{n-1})x + \ldots + a_2)x + a_1)x + a_0 \tag{3}$$

When the polynomial is represented in nested form as seen in Eq. (3), the number of multiplications and additions in the equation is reduced to n, according to the polynomial degree. As a result there is a significant reduction in the number of operations, time and energy used in the calculation.

C. CPU and Computation

This section focuses on the examination of MATLAB's function in performing complex calculations and data analysis, with a particular emphasis on the central processing unit's crucial role in task execution and energy efficiency. This research focuses on the variables of CPU processing time, engagement level, and energy expenditure, with a specific emphasis on power usage. Understanding these aspects is crucial in driving advancements in energy-efficient computing systems.

1) CPU time: The emphasis is placed on CPU time rather than "wall-clock" time to provide a more precise benchmark. This measure accounts for the actual time the CPU dedicates to a specific process, offering an unobscured view of the program's time complexity, unaffected by other background processes.

2) CPU utilisation: This measure provides insight into the level of CPU utilisation by providing the proportion of time the CPU is allocated to a certain job during a defined span of CPU time. The monitoring process may be facilitated by using the Windows task manager, providing an integral insight into the efficiency and energy consumption of the process, and is given by the formula:

$$U_{CPU} = \frac{T_p}{T_{CPU}}, \tag{4}$$

where U_{CPU} is the CPU utilisation, T_p is the CPU spent on a particular process, in this case it was MATLAB.exe, and T_{CPU} is the total time the CPU has spent executing. In order to analyse peaks and sustained loads on the processor, the Area Under the Curve (AUC) of the CPU utilisation data can be used to compare the workloads of different programs, the AUC is also how the total energy used is calculated for the time-series power consumption data.

3) CPU power: The study uses two approaches to assess the power consumption of the CPU: direct and indirect. Direct measurement involves the use of a PMD or onboard sensors to quantitatively assess the magnitude of electric current being supplied to the CPU. In contrast, indirect methods include estimating energy usage by considering variables such as clock speed, CPU temperature, and voltage. Power consumption may be measured directly using a PMD or indirectly via the use of OpenHardwareMonitor, an open-source software application that estimates or directly monitors the power consumption of the CPU, depending on the presence of onboard sensors.

III. Methodology

To assess the impact of Horner's method on the computational efficiency of the NARMAX polynomial, we developed a methodology that combines theoretical analysis with empirical measurement of CPU energy consumption and utilisation. This study was conducted in a controlled environment, using MATLAB R2023a to implement and evaluate the polynomial equations, both before and after applying Horner's method. To ensure the accuracy of the results, the equations were subjected to 1×10^7 iterations.

The measurement of the CPU's energy consumption was carried out using a PMD, connected between the power supply and the CPU's 12v EPS (Extra Power Supply) power connector. This direct connection allowed for the precise collection of energy consumption data during the execution of mathematical operations in MATLAB.

Parallel to the energy consumption measurement, a dual approach was employed to monitor the CPU utilisation by the `MATLAB.exe` process. For this purpose, a specific algorithm was developed with carefully chosen parameters to operate efficiently and discreetly, avoiding unnecessary terminal openings. This algorithm utilised the `getProcessTimes()` function from the Windows library in C++. This function retrieves the total CPU time used by the `MATLAB.exe` process on the system, allowing us to calculate Processor Utilisation using Equation (4). The CPU utilisation data,

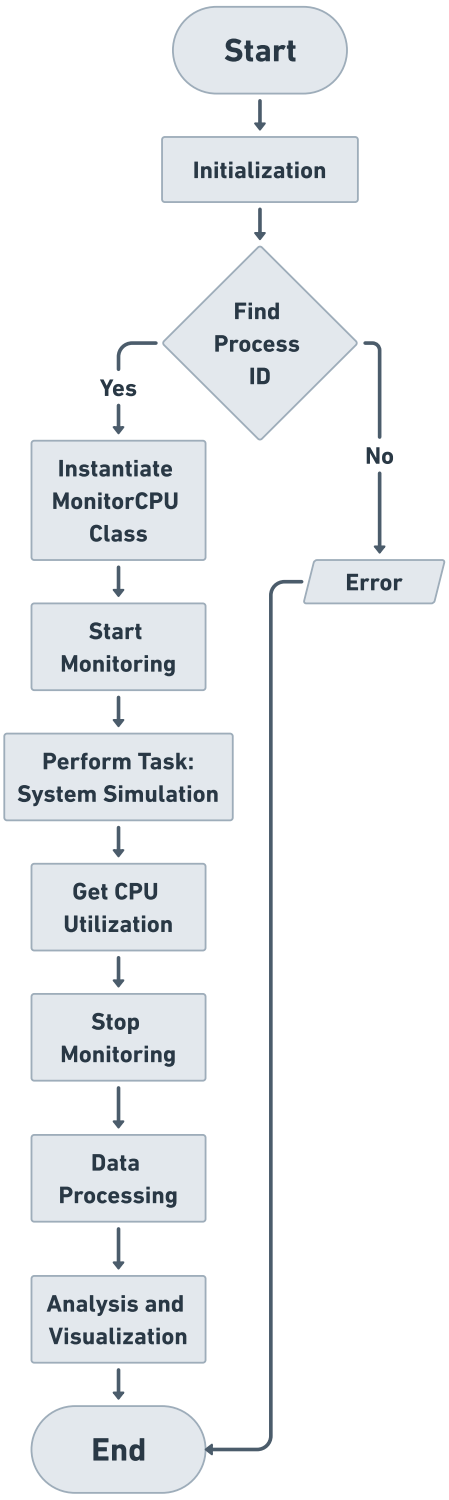


Fig. 1. Flowchart illustrating CPU monitoring for the `MATLAB.exe` process. The procedure begins by validating the existence of the `MATLAB.exe` process on the system. After verification, the CPU monitoring starts, and `MATLAB.exe` executes the 'Perform Task' function, which involves the system simulations. CPU utilisation data are carefully documented during the simulation operations at the same time. After the monitoring phase and system simulations are completed, the data is subjected to thorough processing, detailed analysis, and visualisation. If the `MATLAB.exe` process is not detected, the workflow is immediately stopped, and an error condition is triggered, thereby stopping the activity.

including processing times and utilisation percentages, were continuously collected and stored for subsequent analysis. This methodological choice ensured the accuracy of processor utilisation measurements.

Additionally, a MATLAB control interface was used to manage the start and end of data collection, as well as to process and analyse the collected data. This interface allowed for effective integration between energy consumption data collection and CPU utilisation analysis, facilitating the comparison between conditions before and after applying Horner's method.

Through this methodology, it was possible to conduct a detailed analysis of computational performance and energy consumption associated with executing complex mathematical operations in `MATLAB.exe`. This study highlighted the effectiveness of Horner's method in optimising computational performance, providing valuable insights into the interplay between algorithmic theory and computational practice in high-demand processing contexts. The flow chart for the code can be seen in Fig. 1.

A. Application of Horner's method to NARMAX model

1) Mackey-Glass: The NARMAX model parameters for the Mackey-Glass system were determined using an orthogonal regression approach. A simulated system comprising 6000 data points served as the basis for this estimation. The model was constructed to accommodate a maximum lag of 10 and a non-linearity degree of 4, criteria selected to ensure an optimal balance between model complexity and predictive accuracy.

The efficacy of the model was evaluated using the Root Mean Square Error (RMSE) metric. An RMSE of 0.267 was observed when the model's predictions were compared with a subsequent set of 6000 data points, which served as a validation dataset distinct from the training set. This low error metric highlights the model's robustness and predictive precision in replicating the complex dynamics inherent in the Mackey-Glass system.

The mathematical representation of the Mackey-Glass NARMAX model is articulated as follows:

$$\begin{aligned} y(k) = {} & 5.3569y(k-1) - 14.0952y(k-2) \\ & + 24.3744(k-3) - 31.1053y(k-4) \\ & + 30.9521y(k-5) - 24.4804y(k-6) \\ & + 15.1982y(k-7) - 7.0278y(k-8) \\ & + 2.1031y(k-9) + 0.013883 \\ & + 0.066535y(k-9)y(k-10) - 0.35279y(k-10) \\ & + 0.095508y(k-10)^2 - 0.14165y(k-10)^3 \\ & - 0.0083594y(k-8)y(k-10)^3 \\ & + 0.050857y(k-10)^4 \end{aligned} \tag{5}$$

choosing $y(k-10)$ as the variable to factorise in yields the

NARMAX model in nested form:

$$\begin{aligned} y(k) &= 5.3569y(k-1) - 14.0952y(k-2) \\ &\quad + 24.3744y(k-3) - 31.1053y(k-4) \\ &\quad + 30.9521y(k-5) - 24.4804y(k-6) \\ &\quad + 15.1982y(k-7) - 7.0278y(k-8) \\ &\quad + 2.1031y(k-9) + 0.013883 + y(k-10) \\ &\quad \times \big(0.095508 + y(k-10)(-0.14165 \\ &\quad - 0.0083594y(k-8) \\ &\quad + y(k-10) \times 0.050857))) \end{aligned} \tag{6}$$

2) Lorenz: Consider the simplified NARMAX model for the Lorenz attractor, presented by the Equations (7), (8) and (9), similar to a model proposed by Aguirre and Billings [18], note the error components were omitted for simplicity. Observing these equations it is clear the Lorenz Attractor NARMAX model has a non-linearity degree of 2, and maximum lag of 1, which may affect the energy reduction since the higher the power in the polynomial the more it can be factored.

In this model, there are three variables: $x(k-1)$, $y(k-1)$, and $z(k-1)$. Implementing Horner's method in this context deviates slightly from its standard application, given the multivariate nature of the equations, but the core principle remains intact. It's essential to strategically select one variable. Ideally, opting for the most frequently occurring variable can minimise the number of required multiplications, streamlining the computational process.

$$\begin{aligned} x(k) &= 0.85919x(k-1) + 0.22489y(k-1) \\ &\quad - 2.833 \times 10^{-4}x(k-1)z(k-1) \\ &\quad - 3.4598 \times 10^{-4}y(k-1)z(k-1) - 0.012927 \\ &\quad - 6.9159 \times 10^{-4}x(k-1)y(k-1) \\ &\quad + 7.326 \times 10^{-4}x(k-1)^2 \end{aligned} \tag{7}$$

$$\begin{aligned} y(k) &= 1.109y(k-1) - 0.006713y(k-1)z(k-1) \\ &\quad - 0.018688x(k-1)z(k-1) + 0.54947x(k-1) \\ &\quad - 3.3705 \times 10^{-5}z(k-1)^2 + 6.7054 \times 10^{-5} \\ &\quad + x(k-1)y(k-1) - 3.6965 \times 10^{-4}x(k-1)^2 \end{aligned} \tag{8}$$

$$\begin{aligned} z(k) &= 1.01z(k-1) + 9.34 \times 10^{-3}x(k-1)y(k-1) \\ &\quad - 2.1708 \times 10^{3}z(k-1)^2 + 7.2469 \times 10^{-3}x(k-1)^2 \\ &\quad + 7.6919 \times 10^{-3}y(k-1)^2 - 0.47834 \\ &\quad + 4.8760 \times 10^{-5}y(k-1)z(k-1) \end{aligned} \tag{9}$$

For instance, if designate $x(k-1)$ as the variable to be factorised in $x(k)$ and $y(k)$, while assigning $z(k-1)$ as the variable for $z(k)$, is it possible to express the equations in their nested forms. This aids in illuminating their structure and facilitating an efficient computation. The nested representation of the equations, obtained through this methodical variable selection and restructuring, are as follows:

$$\begin{aligned} x(k) &= -0.012927 - 3.4598 \times 10^{-4}\, y(k-1)z(k-1) \\ &\quad + 0.22489y(k-1) + x(k-1)\big(0.85919 \\ &\quad - 2.833 \times 10^{-4}\, z(k-1) \\ &\quad - 6.9159 \times 10^{-4}\, y(k-1) \\ &\quad + x(k-1)7.326 \times 10^{-4}\big) \end{aligned} \tag{10}$$

$$\begin{aligned} y(k) &= 1.109\, y(k-1) - 0.006713\, y(k-1)z(k-1) \\ &\quad - 3.3705 \times 10^{-5}\, z(k-1)^2 \\ &\quad + x(k-1)\big(-0.018688\, z(k-1) + 0.54947 \\ &\quad + 6.7054 \times 10^{-5}\, y(k-1) \\ &\quad + x(k-1)(-3.6965 \times 10^{-4})\big) \end{aligned} \tag{11}$$

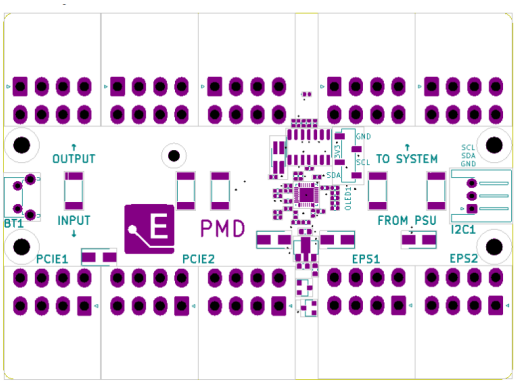
$$\begin{aligned} z(k) &= -0.47834 + 9.34 \times 10^{-3}\, x(k-1)y(k-1) \\ &\quad + 7.2469 \times 10^{-3}\, (x(k-1))^2 \\ &\quad + 7.6919 \times 10^{-3}\, (y(k-1))^2 \\ &\quad + z(k-1)\big(1.01 + 4.8760 \times 10^{-5}\, y(k-1) \\ &\quad + z(k-1)(2.1708 \times 10^{-3})\big) \end{aligned} \tag{12}$$

Here, the calculations are optimised and a clearer perspective of the interactions among variables is gained. This approach, informed by Horner's method, ensures a more concise and computationally efficient representation of the original system of difference equations

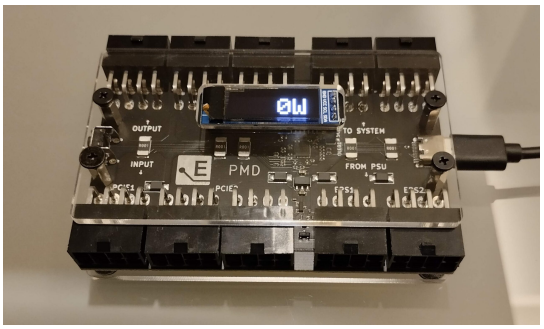
B. Bench-marking CPU

The CPU time and usage were carefully averaged throughout each segment of 1×10^7 data points to determine mean run-time and utilisation, giving insights into computational efficiency and resource engagement. As mentioned before, in this study, it examined the power consumption of the CPU utilising both direct and indirect measurement approaches. However, our preliminary results indicate that the indirect strategy exhibited much poorer accuracy and slower collection of data, operating at one-second intervals, in contrast to the real-time precision provided by the direct method using PMD. As a result, we set a higher emphasis on using the direct measurement method in order to enhance the dependability and uniformity of our data.

CPU power was evaluated in real-time using a PMD that captured continuous time-series power data, transmitted via serial connection and stored in a .csv file for analysis. The analysis was performed in MATLAB, extracting insights from raw data to understand CPU power dynamics under different operational loads. A design of the PMD from ElmoreLabs is shown in Fig. 2a [19]. This mixed-method approach provided a thorough evaluation of CPU performance and energy efficiency. Fig. 2b gives an overview of the PMD and USB connection. The PMD intercepts the EPS connector from the PSU and sends the voltage, current, and power consumption to the .csv file. The power consumption for each run was estimated by dividing the total energy used by 50. In order to analyse peaks and sustained loads on the processor the AUC was estimated using the trapezoidal rule for the mean CPU utilisation data. The AUC by means of the trapezoidal rule is



(a)



(b)

Fig. 2. (a) PMD Circuit Board Layout designed by ElmoreLabs. The PMD measures voltage, current and power consumption from the PSU. (b) ElmorLabs PMD-USB (Power Measurement Device with USB). This device presents 2x EPS 8-pin (CPU) and 3x PCIE 8-pin (GPU) power inputs from the power supply. Source: ElmorLabs.

also how the total energy used was estimated for the time-series power consumption data using Equation (13):

$$E = \int_{t_1}^{t_2} P(t)\, dt \tag{13}$$

where t_1 and t_2 specify the time interval on which the measurement took place, and $P(t)$ is the power signal.

IV. Results

This study used Horner's method to accurately evaluate energy utilisation while simulating NARMAX models in MATLAB. It directly quantified the power necessary to execute particular programmes to understand the intricate mechanics of energy consumption and the environmental effect of certain computing activities. An in-depth CPU analysis was performed before and after Horner's Method was applied to the NARMAX polynomial of two systems: the Lorenz attractor, a simple model for the chaotic system, and the Mackey-Glass system, whose model is more complex (Table 1) due to its higher maximum lag and non-linearity degree.

These results present a 0.6% and 29.3% energy reduction, 94.9% and 42.4% CPU time reduction, and a 44.48% and 95.0% reduction for AUC of CPU Utilisation for the Lorenz Attractor and Mackey-Glass system respectively, which were derived from Table 1. This data indicates a consistent reduction in CPU time post-Horner's method application, aligning with the AUC of CPU utilisation.

It is also essential to emphasise that a comprehensive examination of the findings reveals a multifaceted relation between the duration of simulation and the use of energy. The relationship between simulation duration and energy consumption does not exhibit a linear proportionality, wherein a reduction in simulation time does not result in a corresponding drop in energy expenditure. The aforementioned result highlights the delicate nature of energy dynamics in computing systems and encourages a more detailed analysis of solutions targeted at improving energy efficiency.

The relationship between carbon emissions and consumption of energy has been well recognised and proven. By using Horner's technique, a discernible drop in energy consumption was noticed, leading to a corresponding reduction in carbon emissions. This outcome serves to validate the efficacy of the strategy in mitigating the carbon footprint. The study conducted in our research examined the overall power consumption of the CPU. However, we were able to get valuable insights into individual process workloads by seeing the decrease in CPU time and utilisation. This highlights the effectiveness of our technique in many computing activities beyond the assessment of NARMAX models.

TABLE I
COMPUTER SIMULATION PERFORMANCE OF NONLINEAR DYNAMICAL SYSTEMS. TWO SYSTEMS ARE TESTED: LORENZ ATTRACTOR (LA) AND MACKEY-GLASS (MG) BOTH BEFORE AND AFTER APPLYING HORNER'S METHOD. THE PERFORMANCE MEASUREMENTS ARE: I: CPU TIME [MS]; II: AUC OF CPU UTILISATION, III: ENERGY USED PER RUN [J]; IV: NUMBER OF MULTIPLICATIONS; V: NON-LINEARITY DEGREE.

System	I	II	III	IV	V
LA	95.0	802.1	161.6	30	2
LA-H	54.7	445.3	160.3	24	2
MG	1093.7	9384.0	208.0	25	4
MG-H	56.3	465.1	147.1	14	4

V. Conclusion

This study has introduced a novel method to experimentally evaluate energy reduction on the CPU achieved through code optimisation. Horner's method has been applied to two identified polynomial NARMAX models for the Lorenz Attractor and Mackey-Glass systems. In many prior works in the literature, algorithm performance has been assessed using computational time metrics. Commands such as `tic/toc` in MATLAB have been widely used as a means to measure the efficiency of such algorithms. It is well-known that this measurement is influenced by many other processes. It has been believed that an average of hundreds of realisations could yield a reliable performance index, as demonstrated in [12]. However, when the focus is on observing the energy reduction resulting from code optimisation, our technique has revealed that computational time may overestimate the benefits.

We have conducted a comprehensive examination of energy consumption associated with the simulation of NARMAX

models in MATLAB, a subject not extensively covered previously. Horner's approach was employed, and a detailed analysis was performed to evaluate its effectiveness in reducing computing time and energy consumption. In a computer simulation of the Lorenz Attractor using the Horner's Method, we observed only a 0.6% reduction in energy consumption, while its CPU time reduction was 42.4%.

In future work, we intend to investigate the memory impact and explore techniques that may be more efficient in reducing energy consumption during computer simulations of nonlinear dynamical systems. Additionally, we aim to develop new performance indices that can simultaneously consider computation time and energy consumption. We also plan to enhance energy efficiency identification systems for highly complex systems, such as those in renewable energy systems, as demonstrated in [10].

ACKNOWLEDGMENT

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 21/FFP-P/10065. Thalita E. Nazaré, MEng, was supported by the John and Pat Hume Doctoral Award – WISH. Erivelton Nepomuceno was supported by Brazilian Research Agencies: CNPq/INERGE (Grant No. 465704/2014-0), CNPq (Grant No. 425509/2018-4 and Grant No. 311321/2020-8) and FAPEMIG (Grant No. APQ-00870-17).

REFERENCES

[1] M. W. Hsiao, J. Edmonds, and J. M. Reilly, "Global energy – assessing the future," *Southern Economic Journal*, vol. 52, no. 4, p. 1194, 1986.

[2] L. Lannelongue, J. Grealey, and M. Inouye, "Green algorithms: Quantifying the carbon footprint of computation," *Advanced Science*, vol. 8, p. 2100707, May 2021.

[3] M. Avgerinou, P. Bertoldi, and L. Castellazzi, "Trends in data centre energy consumption under the european code of conduct for data centre energy efficiency," *Energies*, vol. 10, p. 1470, Sep 2017.

[4] T. Nazaré, J. Gadelha, E. Nepomuceno, and R. Lozi, "Green computing for energy transition: A survey," *IEEE Latin America Transactions*, vol. 21, pp. 937–948, Sept 2023.

[5] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *Proceedings of the 2007 international symposium on Low power electronics and design (ISLPED '07)*, (Portland, OR, USA), pp. 38–43, 2007.

[6] L. Benini, A. Bogliolo, and G. De Micheli, "A survey of design techniques for system-level dynamic power management," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, pp. 299–316, June 2000.

[7] J. P. R and A. S. Pillai, "Enhancing energy efficiency of intensive computing applications using approximate computing," in *2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC)*, (Coimbatore, India), pp. 1231–1236, 2022.

[8] S. Rangan and K. Poolla, "Time-domain validation for sample-data uncertainty models," *IEEE Transactions on Automatic Control*, vol. 41, pp. 980–991, July 1996.

[9] E. A. Efremov, A. A. Chufistov, R. R. Mukaev, and V. Skvortsov, "Methods for optimizing software performance," in *2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus)*, pp. 2072–2075, 2021.

[10] T. Nazaré, K. Mahmoodi, and E. Nepomuceno, "Green-box system identification and carbon footprint analysis for sustainable computing," in *OCEANS 2023 - Limerick*, (Limerick, Ireland), pp. 1–6, 2023.

[11] O. Gómez-Carmona, D. Casado-Mansilla, D. López-de Ipiña, and J. García-Zubia, "Optimizing computational resources for edge intelligence through model cascade strategies," *IEEE Internet of Things Journal*, vol. 9, no. 10, pp. 7404–7417, 2022.

[12] S. Surya, S. Krishna, S. Madhu, and A. Chhetri, "Analysis of processor execution time in modeling dc-dc converters for mild hybrid electric vehicle application," in *2023 5th International Conference on Energy, Power and Environment: Towards Flexible Green Energy Technologies (ICEPE)*, (Shillong, India), pp. 1–6, IEEE, Jun 2023.

[13] L. Lannelongue, J. Grealey, A. Bateman, and M. Inouye, "Ten simple rules to make your computing more environmentally sustainable," *PLOS Computational Biology*, vol. 17, p. e1009324, Sep 2021.

[14] Y. Zhao, T. Nazaré, and E. Nepomuceno, "Reduction of carbon footprint of dynamical system simulation," in *2023 15th IEEE International Conference on Industry Applications (INDUSCON)*, pp. 1322–1329, 2023.

[15] S. A. Billings, *Nonlinear System Identification*. Chichester, UK: John Wiley & Sons, Ltd, 2013.

[16] W. G. Horner, "A new method of solving numerical equations of all orders, by continuous approximation," *Philosophical Transactions of the Royal Society of London*, vol. 109, p. 308–335, 1819. Accessed: Oct. 02, 2023.

[17] M. B. R. Cardoso, S. S. da Silva, L. G. Nardo, R. M. Passos, E. G. Nepomuceno, and J. Arias-Garcia, "A new prng hardware architecture based on an exponential chaotic map," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, (Daegu, Korea), pp. 1–5, 2021.

[18] L. A. Aguirre and S. A. Billings, "Retrieving dynamical invariants from chaotic data using narmax models," *International Journal of Bifurcation and Chaos*, vol. 05, p. 449–474, Apr 1995.

[19] Elmore Labs, "Pmd – power measurement device." https://elmorlabs.com/product/elmorlabs-pmd-usb-power-measurement-device-with-usb/, 2023.