energies

MDPI

Article

Optimal Selection of Switch Model Parameters for ADC-Based Power Converters

Saif Alsarayreh * and Zoltán Sütő

Department of Automation and Applied Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary; zoltan.suto@aut.bme.hu
* Correspondence: saif.alsarayreh@aut.bme.hu

Abstract: Real-time hardware-in-the-loop-(HIL) simulation integration is now a fundamental component of the power electronics control design cycle. This integration is required to test the efficacy of controller implementations. Even though hardware-in-the-loop-(HIL) tools use FPGA devices with computing power that is rapidly evolving, developers constantly need to balance the ease of deploying models with acceptable accuracy. This study introduces a methodology for implementing a full-bridge inverter and buck converter utilising the associate-discrete-circuit-(ADC) model, which is optimised for real-time simulator applications. Additionally, this work introduces a new approach for choosing ADC parameter values by using the artificial-bee-colony-(ABC) algorithm, the firefly algorithm (FFA), and the genetic algorithm (GA). The implementation of the ADC-based model enables the development of a consistent architecture in simulation, regardless of the states of the switches. The simulation results demonstrate the efficacy of the proposed methodology in selecting optimal parameters for an ADC-switch-based full-bridge inverter and buck converter. These results indicate a reduction in overshoot and settling time observed in both the output voltage and current of the chosen topologies.

Keywords: associate discrete circuit; real-time simulation; field-programmable gate array; optimisation algorithm; power converters

Citation: Alsarayreh, S.; Sütő, Z. Optimal Selection of Switch Model Parameters for ADC-Based Power Converters. *Energies* **2024**, *17*, 56. https://doi.org/10.3390/en17010056

Academic Editors: Krzysztof Górecki and Kalina Detka

Received: 12 November 2023
Revised: 12 December 2023
Accepted: 18 December 2023
Published: 21 December 2023

1. Introduction

The progress in power-electronics applications and the expanding range of power-electronics converters have significantly improved their operational capabilities in the field. Therefore, there is a substantial need to simulate and test these converters. Nevertheless, the necessity to build the converter and conduct physical testing on both the converter and its controller is reduced, due to many practical limitations. These limitations contain possible hazards of harm, doubts about the robustness of the converter, and the associated financial costs. Real-time simulation (RTS) is a common way to test how well a converter controller works in a hardware-in-the-loop-(HIL) environment, which involves connecting actual physical components to simulated models. Incorporating real-world hardware effects into this integration improves the dependability of the simulation, rendering it a viable instrument for validating control techniques and evaluating the efficiency of power-electronics converters, which gives RTS an advantage over other simulation methods, in regard to accuracy and reliability [1,2].

Prior to installation and commissioning, it is necessary to conduct testing and verification procedures on the controller, to ensure its functionality. Performing testing in a real-time-simulation-(RTS)-environment setting is the most feasible approach to guaranteeing the system's resiliency and to effectively evaluating its capabilities and functionality. Moreover, utilising RTS is vital for executing random switching investigations, as it substantially reduces the total length of the study. RTS is renowned for its capacity to deliver instantaneous reactions, enabling the dynamic simulation of power-electronics systems

with precise timing data. RTS provides a more accurate depiction of the system's dynamic reaction compared to conventional time-domain simulations, which may not accurately replicate real-time behaviour [3,4].

In order to examine the dynamics of any given system, the simulation speed must fall within the range of tens of seconds. The process that is being referred to is commonly known as real-time simulation. The real-time simulator effectively solves the mathematical model of a given system by discretising time into short and defined steps. It then correctly reproduces the desired waveforms, such as the voltage and currents of power converters. Enhanced mathematical models yield heightened precision in voltage and current measurements. The slowness of the simulation time observed in power converters can be attributed to the inefficient switching technique employed by these devices. Furthermore, the model needs to include the capability of observing switching characteristics and detecting and analysing unusual events, such as switch short circuits. Additionally, the model must account for the latency in data transmission from the converter switches and external networks, resulting in errors. Hence, developing a stable model for switching converters featuring many topologies is a great challenge.

A flexible numerical integration method is one approach to accurately and efficiently modelling converters. This method allows for the appropriate level of speed and precision in the modelling process. Additionally, interpolation techniques can be employed to correct errors arising from the switching phenomena. It is important to note that this approach does not fix the simulation time step and nodal matrix. Furthermore, the solution method is characterised by its non-iterative nature. If many switching-device models require changes in status within a single time step, the network model is accordingly adjusted to accommodate all of these demands. The last ones are only considered in the resolution of the subsequent time interval. Hence, the solution is deemed inaccurate for a fraction of the time-step size after the first request. This inaccuracy can lead to the development of switching errors, inaccurate power flows, and non-characteristic harmonics. The measures implemented to mitigate the associated inaccuracy often entail modifications to the time-step size and the utilisation of interpolation algorithms. Therefore, a notable drawback to these approaches is the substantially higher levels of computing effort at specific time intervals. This conflicts with the real-time simulation, as it demands that the solution process be efficient in each time step [5].

Another approach to RTS modelling is the resistive model. While this method enhances the system's dynamic behaviour by eliminating undesirable oscillations, it also requires changing the system's topology for each switch state. This presents a significant challenge when dealing with complex and large-scale systems. However, in the case of basic converter types, it is possible to utilise a resistive model to obtain a speed advantage at the expense of accuracy. By contrast, it is observed that nonlinear switch models exhibit significant inefficiency when employed for CPU simulation. Moreover, their implementation on FPGA platforms generally requires a significant allocation of hardware resources, primarily due to the iterative nature of the solution. The time-averaging method and the state-space method (SSM) are alternative approaches that can also be employed. The accuracy of the SSM approach to solving equations is limited, particularly when verifying the controller linked to the switching power converter. The asynchronous interface between the controller and the simulator poses a significant challenge in this regard [6,7].

The utilisation of the ADC technique has various advantages. Firstly, the stability of the network matrix is defined, making it constant and stable, hence avoiding the need for matrix inversion calculations, leading to less computational burden and higher computational efficiency [2]. Furthermore, the utilisation of field-programmable gate arrays (FPGAs) can lead to enhanced efficiency, hence reducing the amount of memory space consumed within the FPGA. However, the ADC model generates simulation losses and transients not present in real-world scenarios. Numerical errors are considered to be the inaccuracies that arise from the ADC-based model [4,8].

The next parts of this paper are structured in the following manner. Section 2 provides a comprehensive overview of the modelling process using the ADC method. Section 3 gives the eigenvalues analysis used to select the optimal ADC parameters. Section 4 outlines the proposed approach for the optimal calculation using the ABC, GA, and FF algorithms. The simulation results obtained through the utilisation of the proposed method are presented in Section 5. Subsequently, the conclusion of the study is outlined in Section 6.

2. Associated-Discrete-Circuit-(ADC) Modelling

The associated-discrete-simulation method integrates a model representing an equivalent conductance and a current source. The mentioned parts are connected in a parallel configuration, to effectively replicate the operational characteristics of the various switching elements within the circuit, including but not limited to IGBTs, diodes, thyristors, and related devices, as shown in Figure 1. The inductive characteristics of the power switch are observed while it is in the ON state, and these characteristics can be measured using the parameter L_{sw}. On the other hand, during the OFF state, the power switch demonstrates capacitive characteristics, which can be symbolised by the variable C_{sw}. A small parasitic component can be integrated into the model, to mitigate the presence of overshoots and oscillations in switching transients. One such element is a resistor denoted as R_{sw}, which can be connected in series with the capacitor C_{sw}. The optimal operation of these models requires the usage of low capacitance and inductance values. As a result, this enables a reduction in the required time interval, denoted as Δt, to develop the discrete-time system. Hence, this specific model is sometimes referred to as the small-time-step model in academic literature [9,10].

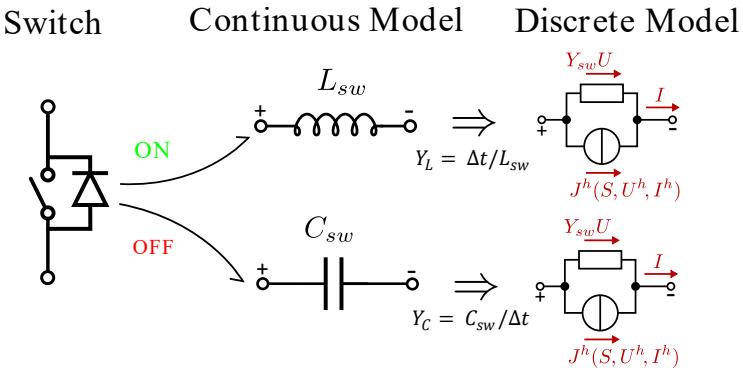

Figure 1. Conceptualization of the ADC-switch-based method.

The discrete-time model of the ADC switch in the ON state can be represented as

$$\frac{dI(t)}{dt} = U(t)$$

$$d(I_{(t)} - I_{(t-1)}) = \int_{t_0}^{t} \frac{U(t)}{L}\, dt \tag{1}$$

$$I_{(t)} = I^h + \frac{\Delta t}{L_{sw}} \cdot \int_{t_0}^{t} U(t)\, dt$$

$$I = J_L^h + Y_L U. \tag{2}$$

The earlier calculated inductor current, labelled I^h, can be considered a continuous current source, represented as $J_L^h = I^h$. Moreover, the parallel conductance, represented as $Y_L = \frac{\Delta t}{L_{sw}}$, determines the increase in current caused by the voltage U during the given time step.

Equations (3)–(5) represent the discrete-time model of the ADC switch in the OFF state, assuming we neglect the extra damping resistance. The variable U^h represents the

voltage magnitude at the preceding time step. The variable Y_C denotes the conductance of the switch in its inactive state:

$$\frac{dU_{(t)}}{dt} = I_{(t)}$$

$$U_{(t)} - U_{(t-1)} = \int_{t_0}^{t} \frac{I(t)}{C}\, dt \tag{3}$$

$$U_{(t)} = U_h + \frac{\Delta t}{C_{sw}} \cdot \int_{t_0}^{t} I(t)\, dt.$$

Rearranging Equation (3) by expressing the current I, Equation (4) is formulated:

$$I = J_C^h + Y_C U, \tag{4}$$

where

$$\begin{aligned} Y_c &= C_{sw}/\Delta t \\ J_c^h &= -Y_C U^h. \end{aligned} \tag{5}$$

The position of the switch, represented as S, solely impacts the current source, while the conductance stays unaltered by the switch position. Therefore, the equation governing the behaviour of the switch can be formulated as follows:

$$I = Y_{\mathrm{sw}} U + I^h S - Y_{\mathrm{sw}} U^h S. \tag{6}$$

The logical role of the switch state S in the context of an IGBT switch with a parallel diode is as follows:

$$S = G + S^h \cdot \left\{ \begin{array}{ll} 1 & \text{if } I^h < 0 \\ 0 & \text{otherwise} \end{array} \right\} + \overline{S}^h \cdot \left\{ \begin{array}{ll} 1 & \text{if } U^h < 0 \\ 0 & \text{otherwise} \end{array} \right\}. \tag{7}$$

The variable G is utilised to indicate the turning-on signal for the power switch, such as an IGBT. The other parts in Equation (7) describe the condition of the diode. Different varieties of power switches necessitate an alternative form of logical operation. It is essential to acknowledge that, in contrast to other behavioural models of power converters, the logic functions in ADC models are specifically designed to address the switching devices on a local level without considering the overall circuit logic.

However, it is important to acknowledge that the capacitors and inductors acquired via the ADC method may not precisely reflect the parasitic components in real switches. The presence of this gap presents the ability to introduce inaccuracies inside the simulation. In the domain of transient functions, there have been notable numerical simulation mistakes when applying the ADC modelling method. Switching power losses arising from numerical integration are acknowledged despite their absence in the physical domain. During modelling, it was noted in a prior work [11] that switching losses occurred due to inaccuracies in circuit parameters. Nevertheless, a thorough examination of this occurrence still needs to be undertaken.

In Equation (8), it is assumed that during the n step, the switch is in the ON state. In that instance, the switch can be considered equivalent to the inductor (Lsw), and it is possible to determine the amount of energy stored on the inductor. As a parallel current source represents the energy stored in the inductor:

$$\begin{aligned} I_{Lsw}(n) &= I(n) \\ V_{Lsw}(n) &= 0 \\ E_L &= 0.5 \cdot Lsw \cdot I^2(n) \\ J^h(n) &= -I_{Lsw}(n-1) = -I_{load}(n-1) \end{aligned} \tag{8}$$

In the $n+1$ time step, the switch is in the OFF state, Lsw is replaced by Csw, and the initial energy stored in the capacitor is calculated using Equation (9). The energy stored in the inductance goes to zero in the ON-OFF state transition of the switch:

$$J^h(n+1) = Y_{sw}U_{Lsw}(n) = 0 \tag{9}$$

In Equation (10), it is also assumed that during the n step, the switch is in the OFF state. In that instance, the switch can be considered equivalent to the capacitor (Csw), and it is possible to determine the amount of energy stored in the capacitor:

$$\begin{aligned} I_{Csw}(n) &= 0 \\ V_{Csw}(n) &= U(n) \\ E_C &= 0.5 \cdot Csw \cdot U^2(n) \\ J^h(n) &= Y_{sw}U_{Lsw}(n-1) = Y_{sw}U_{Lsw}(n-1). \end{aligned} \tag{10}$$

Also, in the $n+1$ time step, the switch is in the ON state, Csw is replaced by Lsw, and the initial energy stored in the inductor is calculated using Equation (11). The energy stored in the capacitor goes to zero in the OFF-ON state transition of the switch:

$$J^h(n+1) = -I_{Lsw}(n) = 0. \tag{11}$$

The capacitor will change to an inductor for a switch modelled using the ADC method to be switched to ON. The initial current flowing through the inductor must equal that of the capacitor, which is 0 amperes. There is no abrupt alteration in the flow of electric currents within the inductor. The charging process of the ADC switch commences, leading to a gradual rise in the current of the equivalent inductor until it achieves a steady state. Meanwhile, the voltage of the switch will gradually decline to zero. The multiplication of the voltage and current values of the ADC model yields a non-zero result. Similarly, the energy expended during this operation is also non-zero, which can be interpreted as the energy stored in the inductor.

The same can be noticed during the switch turn-off, as the model will change from an inductor to an equivalent capacitor. As the capacitor's initial voltage and current are both zero, energy will be stored in the capacitor due to the charging process, as the capacitor voltage will reach a new non-zero steady-state value.

It is necessary to reset the current value of the inductor, in order to facilitate the subsequent switching-on operation, which aims to replace the capacitor with a state of zero current in a stable condition. In order to ensure proper functioning, it is necessary to reset the voltage value of the capacitor prior to replacing the inductor during the switching-off process. The phenomena above result in virtual switching losses inside the ADC model of a power switch.

Finally, the calculation of virtual power loss can be determined using the following equation:

$$E_{Lost} = E_L + E_C = \frac{1}{2}LswI^2 + \frac{1}{2}CswU^2. \tag{12}$$

To summarise, there are several drawbacks associated with this particular model. Firstly, the parameter settings are not universally applicable, and selecting an appropriate parameter set can often be challenging. The non-switching circuit components and the applied integration approach influence the transient behaviours. Additionally, in applications with high frequencies, the virtual power loss resulting from the charging and discharging of capacitors and inductors in the model can present challenges.

3. Eigenvalue Analysis for Optimum Switch Parameter Selection

The selection of the parameter Y_{sw} value holds the capacity to influence the transient spikes introduced into the performance of the ADC model, as it directly represents the values of the switch inductance and capacitance. A non-optimized value of Y_{sw}, for example, could result in a negative C_{sw} value, and the discrete-time system becomes unstable. To examine it, initially, the derivation of the state-space equation is undertaken. Additionally, it is necessary to convert continuous equations into discrete equations. In the end, the eigenvalues of the matrix representing the system are calculated. The determination of the optimal value of Y_{sw} can be achieved by the analysis of the eigenvalues.

By utilizing continuous state space equations, where A represents the state matrix and B represents the input matrix, and by applying the Laplace transform and then working with the backward Euler technique, the substitution of s by $(z-1)/z\Delta t$ in Equation (13) allows for the discretisation process:

$$\begin{aligned} x^{'}(t) &= A \cdot x(t) + B \cdot u(t) \\ s \cdot X(s) &= A \cdot X(s) + B \cdot U \\ x(t+1) - x(t) &= A \cdot \Delta t \cdot x(t+1) + B \cdot \Delta t \cdot u(t+1) \\ x(t+1) &= A^{'} x(t) + B^{'} u(t). \end{aligned} \tag{13}$$

$A^{'}$is the discrete state matrix and equal to

$$A^{'} = (I - A \cdot \Delta t)^{-1}.$$

This enables the formation of a relationship between the variables z and s, as indicated in Equation (14). In this context, Δt represents the time step, whereas s is the complex number $s = \sigma + j\omega$. By employing the backward Euler integration technique, it is possible to transform the left-hand side of the s-plane onto the z-plane, yielding a circular layout. The circle under consideration is the red circle, centred at the complex number $0.5 + 0j$ and has a radius of 0.5, as shown in Figure 2.

$$s = \frac{1 - \frac{1}{z}}{\Delta t} \tag{14}$$

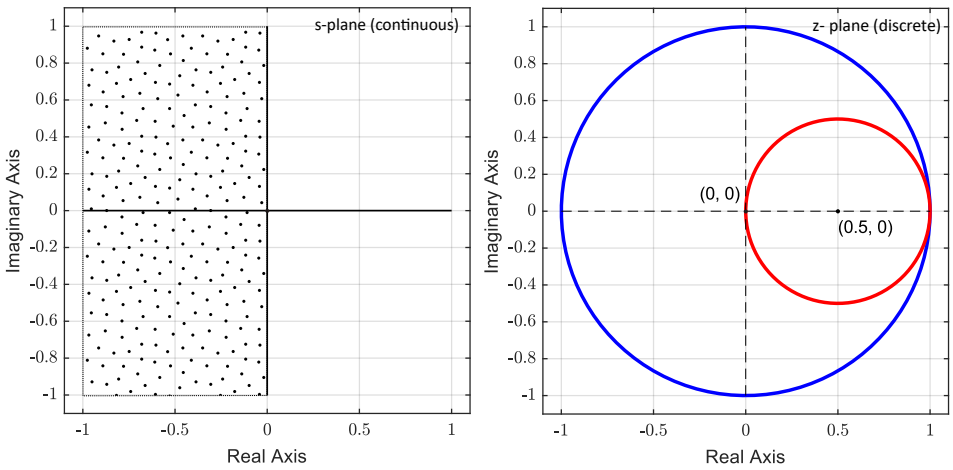

Figure 2. The s-plane left-hand side mapped onto the z-plane (red circle), inside the stability region of a discrete system (blue circle), using the Backward Euler method.

When discussing a continuous system that responds to an input of unity step, the settling t_s can be calculated using Equation (15), with a tolerance of $\pm 2\%$.

$$t_s \cdot \sigma \approx 4 \tag{15}$$

The real component of the pole, $-\sigma$, is determined when the pole lies on the s-plane. Consequently, the discrete system transfer function can be represented as shown in Equation (16). The variable z denotes a complex number that provides an expression of the poles of the discrete system [12,13]:

$$4\Delta t \cdot |z|^2 \approx (Re(z) - |z|^2) * t_s. \tag{16}$$

According to the stability criteria of a second-order system, the link between the angle of the poles and the damping ratio can be established by means of substitution, resulting in Equation (17):

$$\frac{\theta_z}{\sqrt{1-\zeta^2}} \approx \omega_n \Delta t. \tag{17}$$

The damping ratio is denoted by ζ, whereas the natural frequency of an undamped system is represented by ω_n, as mentioned in reference [12]. In analysing a system characterised by eigenvalues, it is fundamental to ensure that the angle associated with these eigenvalues tends towards zero or is minimised, as this mitigates the occurrence of overshoot. By taking this step, the imaginary part of the complex number z tends towards zero (or $Re(z) \approx |z|$), resulting in a modification of Equations (16)–(18) [13]:

$$t_s \approx \frac{4\Delta t|z|}{1-|z|}. \tag{18}$$

A stable system's eigenvalue magnitude ($|z|$) is bounded within the interval of 0 and 1. It is crucial to minimise the settling time by ensuring that the magnitude of the variable z, represented as $|z|$, is adequately small, as indicated by Equation (18). Therefore, the eigenvalues must be located in close proximity to the origin point $(0, 0)$. Figure 3 depicts the regions in which the system exhibits a rapid settling time and low overshoot, as indicated by Equations (17) and (18). The following is a representation of the regions situated within the complex plane. The shaded area represents one region of interest derived from the intersection of these two entities. As multiple minimum-overshoot and minimum-settling-time regions do exist inside the circle, multiple Y_{sw} proposed regions can be highlighted. Hence, the ADC model guarantees the absence of excessive overshoot or incorrect settling time by evaluating the discrete-state matrix's eigenvalues (or dominating poles) and their appropriate placement within the designated region [13].

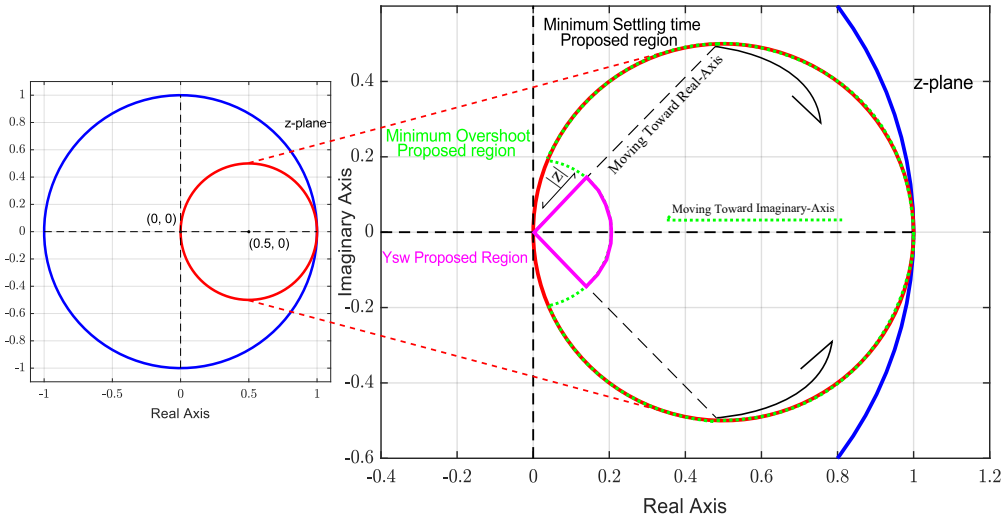

Figure 3. Approximate regions in the complex plane (z-plane) for optimal Y_{sw} value, with minimum overshoot (green dashed-curve) and minimum settling time (black dashed-lines) proposed regions.

4. ADC Switch Model Optimum-Parameters-Value-Selection Algorithms

The methodology employed in this research focused on determining and calculating the value of the ADC switch conductance by locating the system's eigenvalues within a designated region in the z-plane. The primary aim was to reduce the negative effects of overshoot and settling time. The poles of the system were determined through the utilisation of the artificial bee colony (ABC), the firefly (FFA), and the genetic (GA) algorithms. The main rationale behind using nature-inspired optimisation methods and comparing their effectiveness (computation time and convergence rate) was their global exploration, which meant that the algorithms were able to search through a wide range of solution spaces. They were robust while dealing with complex systems and held a certain degree of easiness during implementation. No other algorithms were considered, which opens up the discussion to developing other optimisation methods in the future.

4.1. Artificial Bee Colony (ABC)

The ABC parameters consist of several factors, with the initial population size being the primary consideration. The user determines this value, which serves as a reflection of the optimisation problem's complex nature. A larger initial population size has the potential to enhance the search process inside the designated search space. However, it is essential to note that an increased computing load may accompany this advantage. A limited initial population may restrict the range of search-space exploration yet facilitate faster convergence. The number of iterations for the second parameter is dependent upon the difficulty of the task. The selection of a significant number of iterations in the algorithmic process may facilitate the identification of the optimal solution. However, this approach may also increase the overall time necessary to complete the optimisation process. A low number of iterations could decrease the algorithm's ability to sufficiently navigate the search space, hence decreasing the accuracy in identifying the ideal answer. It is customary to decide on an iteration number within a range from 100 to 1000, thereafter adjusting it in response to performance evaluations and the availability of computational resources [14].

4.2. Firefly Algorithm (FFA)

The firefly algorithm (FFA) is an optimisation technique that draws inspiration from nature and emulates the flashing activity seen by fireflies to attract other fireflies. Fireflies employ bio-luminescence as a means of drawing in potential mates or prey, hence generating intricate patterns of luminous flashes inside low-light environments.

The firefly algorithm employs fireflies as a representation for the solutions, and the appearance degree of the solution is based upon its fitness value. Fireflies tend to be drawn towards other fireflies, influenced by the brightness emitted by the latter, which is closely linked to their overall fitness. More optimal solutions exhibit a greater degree of attraction among other fireflies.

The fundamental procedures of the firefly algorithm start with the creation of an initial population of fireflies, which will serve as the set of potential solutions. The fitness function is a mathematical representation used to assess the quality or effectiveness of solutions within the search space. The attraction of each firefly (solution) is determined by evaluating its brightness (fitness) in relation to other fireflies. The fireflies are directed towards more luminous fireflies inside the search region, with their positions being modified according to their attraction and the distance between them. The current light-intensity (fitness) status is being provided as an update. The fireflies' light intensity (fitness) is modified per their updated spatial coordinates.

A well-constructed fitness function, also called the cost function, that accurately quantifies the performance of the problem in hand will ensure the solution quality by encapsulating the desired optimisation goals and mapping the fitness to the brightness. At the same time, the algorithm stores the optimal solution and compares it to the previous

iteration's best fitness solution. Ultimately, the optimal solution with the best fitness—in other words, the best metaphorical brightness—is outputted.

The process of attraction, movement, and intensity update should be iteratively repeated for a certain number of iterations or until a convergence requirement is satisfied. The utilisation of firefly migration towards brighter fireflies inside this algorithm facilitates efficient solution-space exploration. Customizing the algorithm's attraction function and movement strategy is contingent upon the problem being addressed [14].

4.3. Genetic Algorithm (GA)

The genetic algorithm (GA) is a widely utilised evolutionary computation methodology that draws inspiration from natural selection and genetics mechanisms. This algorithm is employed for the purpose of optimising and solving search-related problems.

An initial population comprised of potential solutions, sometimes referred to as individuals, is created. Typically, these solutions are encoded as binary strings; however, alternative representations may be employed, based on the nature of the problem, such as real numbers or permutations. The fitness function serves to measure the quality of a solution in relation to the specific problem being addressed. Solutions characterised by higher fitness values are indicative of superior performance in individuals.

The algorithm involves selecting existing population members as progenitors for future generations. Fitness levels often determine selection, with fitter population members being chosen. Selected parent pairs undergo a crossover process to produce children. Then, the algorithm mutates a portion of the offspring. Natural mutations cause tiny and random changes in the progeny's genetic material. Genetic diversity is preserved in populations, to avoid early convergence to suboptimal solutions. Mixing the starting population, parents, and children creates a new generational cohort, as they replace less-fit individuals.

The GA algorithm is an effective approach to finding the optimal solution through selection, crossover, mutation, and replacement. The selection is linked to the number of generations, which is also linked to the number of iterations. Crossover and mutation are expressed in the algorithm as a probability. High crossover probability will increase the exploration by creating more diverse offspring, while low crossover probability will push for exploration with more maintained genetic information from the parent. The mutation probability indicates the likelihood of an individual's chromosome going through mutation. High mutation probability will increase the exploration by adding random genetic changes, while low probability will set the algorithm exploration to be reliant on crossover. The typical values for crossover probability are between 0.6 and 0.9, and mutation probability is between 0.01 and 0.1. In our cases, the selection of 0.8 and 0.01 for the crossover and the mutation probabilities led to achieving the optimal answer.

The process of selection, crossover, mutation, and replacement should be iteratively performed for a certain number of generations or until a predetermined termination criterion is satisfied, such as reaching a suitable solution [15].

For all the used algorithms implemented, by trial and error, increasing the initial population to more than 400 will not give different optimal results. It will increase the computation time. Decreasing the initial population to less than 400 will not give the optimal answer. This selected initial population did not lead to premature convergence, did not affect the algorithm's robustness, and did not limit the search space. This stands for the selected test cases in this study.

4.4. Algorithms Cost Function

The cost function for the three algorithms (ABC, FFA, GA) is designed to minimise both overshoot and settling time. It is formulated based on the analysis done to the eigenvalue, as in Section 3. The cost function is validated with the performance of real-world power converters as implemented in [10,13]. The initial element of the cost function, as outlined in Equation (19), seeks to surpass the overshoot by minimising the total number of the squared eigenvalue angles:

$$C_1(Y_{sw}) = \sum_{i=1}^{k} |\theta_{\lambda i}(Y_{sw})|^2. \quad (19)$$

Similarly, the second element of the cost function, derived from Equation (20), can be expressed as the sum of the squared magnitudes of the eigenvalues. In both instances, K represents the number of iterations. Multiplying each component of the cost function, denoted as $\theta_{\lambda i}(Y_{sw})$ and $\lambda_i(Y_{sw})$, by itself decreases the sensitivity of the eigenvalues that are in close proximity to the origin:

$$C_2(Y_{sw}) = \sum_{i=1}^{k} |\lambda_i(Y_{sw})|^2 \quad (20)$$

5. Case Studies

The typologies under investigation were full-bridge inverter, which is the basic building block for multiple-level inverters, like the three-phase nine-level CHB inverter; in this context, we examined one cell of the inverter. The second test case was the buck converter, to prove the diversity for the selection method for different types of switching devices. The associated-discrete-circuit-(ADC) method was employed, in order to construct a model for each of the cases.

The ADC approach employs an inductor and a capacitor to symbolize each switch during its activation and deactivation phases, as stated by source [16]. The discrete formulation of an electrical switch is obtained by expressing it as a composite system consisting of a current source operating in parallel with a conductance. One significant advantage of this methodology is that irrespective of the switch's state the conductance may be fixed by appropriately choosing a suitable time interval. Therefore, the circuit configuration may remain unchanged [9,11]. Additionally, the utilisation of the backward Euler (BE) integration method, renowned for its inherent damping characteristics, will lead to the discretisation of the elements inside the converter circuit and the derivation of the circuit's matrix. The determination of nodal voltages and branch currents is an obvious outcome of resolving the discrete circuit matrix.

Three optimisation algorithms were proposed, in order to determine the optimal parameter values for the full-bridge and buck-converter models, which were implemented using ADC. The purpose of these algorithms was to address the limitations and disadvantages associated with the use of ADC. The primary advantage of utilising an algorithm for optimising ADC settings is its ability to tackle difficult and discrete optimisation problems efficiently. The utilisation of the above approach facilitated the practical examination of a broad spectrum of potential solutions, leading to faster convergence towards the ideal parameters [17].

By employing network modelling techniques, like modified nodal analysis (MNA), it was possible to represent any conversion circuit equations in the form illustrated in Equation (21). In this format, the switch admittance and its inverse were computed in advance, and the values of current and voltage were determined at each time interval Δt by solving Equation (21). The matrix $B_{(n)}$ was subject to change based on the switch state:

$$\begin{aligned} Y \cdot X_{(n)} &= B_{(n)} \\ X_{(n)} &= Y^{-1} \cdot B_{(n)}. \end{aligned} \quad (21)$$

The computation of the circuit parameters occurred through a two-step process. The initial step was the computation of $X_{(n)}$, whereas the subsequent phase relied on the history of the current sources. The process of updating was reliant upon the prior values of the voltages and currents. The utilisation of the network modelling technique allowed for the construction and simulation of the full-bridge circuit and buck converter, incorporating an ADC-based switch model.

Figure 4 shows the process of designing a power converter with the optimal parameter values of an ADC-switch-based model.

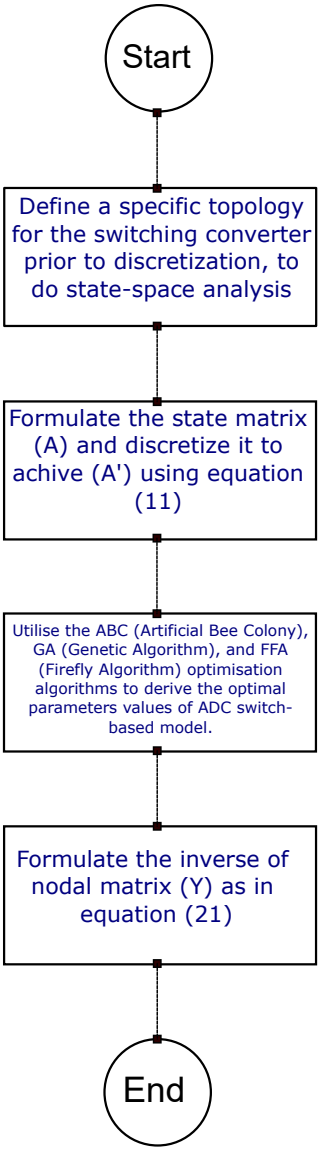

Figure 4. Steps of the design of a power converter with the optimal parameter values of an ADC-switch-based model.

5.1. Full-Bridge Inverter

A block diagram of a full-bridge circuit with a resistive load is seen in Figure 5. Matlab Simulink allowed us to conduct a simulation of the circuit, wherein the switch conductance was set to a value of 1. The switching frequency utilised in this study was 2 kHz. The modulation signal employed was a sine signal characterised by an amplitude of 0.95 and a frequency of 100π. This signal was compared to a triangle signal with a frequency of 2 kHz, to generate the gate signals. Additionally, the parameters used in the present study included a resistance value of 10 ohms (R), an input voltage of 100 volts (E), and a time interval of 200 nanoseconds (Δt). The effect of the resistance value on the output result was recorded, but no further sensitivity analysis was conducted, while the selected value, R = 10 ohms, ensured a minimum effect in our test case. The simulation results are depicted in Figures 6 and 7. The occurrence of transient spikes in both the output voltage and current was noted, resulting in notable virtual power losses inside the ADC model.

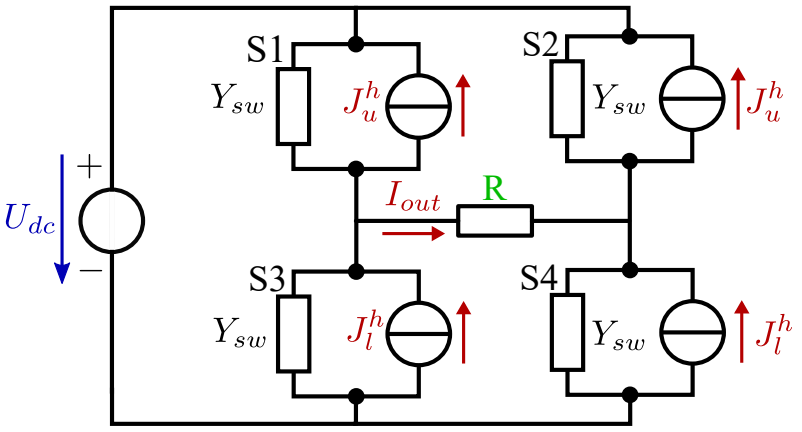

Figure 5. Full-bridge inverter, represented using an ADC-switch-based model.

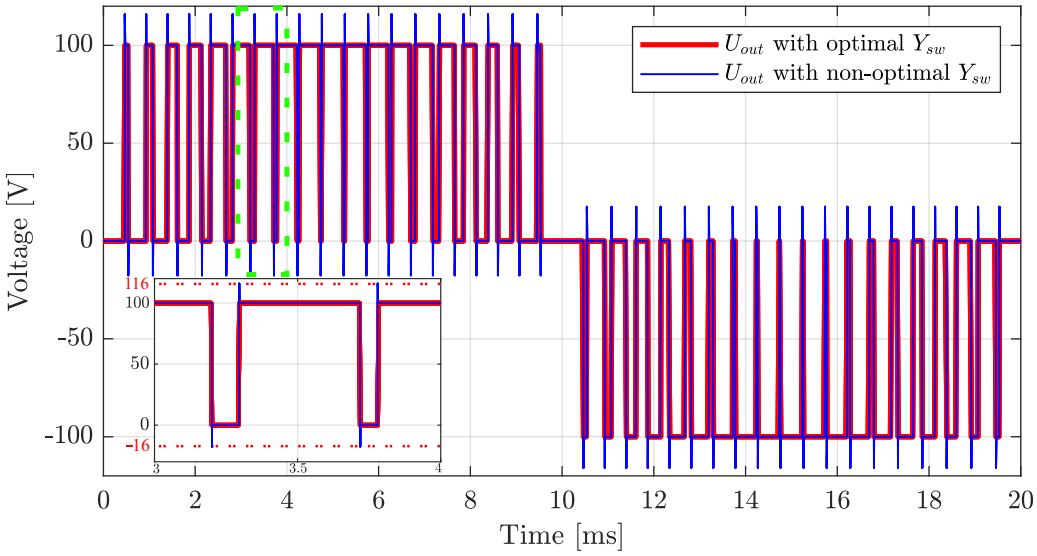

Figure 6. Full-bridge output voltage with $Y_{sw} = 1$ admittance and optimal admittance value $Y_{sw} = 0.1$.

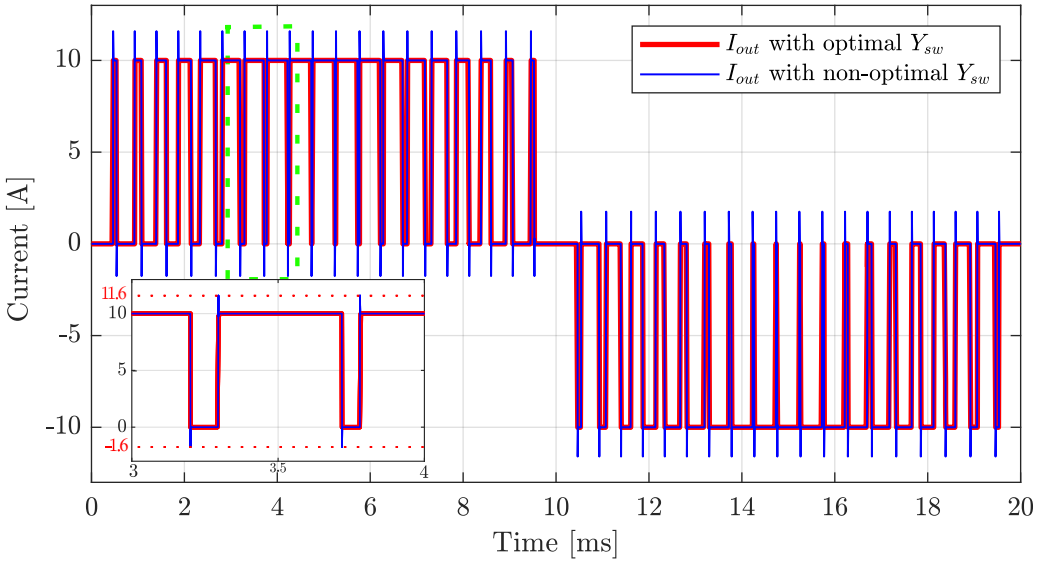

Figure 7. Full-bridge output current with $Y_{sw} = 1$ admittance and optimal admittance value $Y_{sw} = 0.1$.

The full-bridge circuit can be simplified into two distinct topologies. The first topology occurred when switches 1 and 4 were closed, whereas the second topology occurred when switches 2 and 3 were closed. The state-space matrix of the circuit was constructed and subsequently discretised using the backward-Euler-integration approach. Additional details regarding the derivation and discretisation procedure can be found in the publication referenced as [13]. Equation (22) displays the state-space matrix of the full bridge:

$$A = \begin{bmatrix} 0 & \frac{-1}{L} \\ \frac{1}{C} & -\frac{2}{RC} \end{bmatrix}. \tag{22}$$

The discrete-state-space matrix could be obtained based on the variable Y_{sw} as stated by backward Euler, as in Equation (23). According to the eigenvalues of Equation (23), the ABC, GA, and FFA algorithms were employed to find the optimal values of Y_{sw}:

$$A^{'} = \begin{bmatrix} 1 & Y_{sw} \\ \frac{-1}{Y_{sw}} & 1 + \frac{2}{Y_{sw}R} \end{bmatrix}^{-1}. \tag{23}$$

Table 1 shows the optimized values for Y_{sw}, for which the eigenvalues of the $A^{'}$ matrix are located in the marked area as described in Section 3. Within the boundaries of Y_{sw}, an optimum answer for Y_{sw} was found for each run of all the algorithms, resulting in an adequate outcome. Table 2 gives a performance comparison between the algorithms, while locating the conductance optimal value for the full-bridge inverter. Even when changing the number of iterations, as this was one of the primary factors that influenced the computational speed and the convergence rate, it can be seen that the FFA had the lowest convergence rate—hence, the highest computation time. By contrast, the GA algorithm achieved the most down computation time and the highest convergence rate. ABC stood in the middle with both values. Changing the number of iterations did not change the optimal value achieved, it was observed to be between 0.09966 to 0.10098, with no effect on the output voltage and current.

Table 1. The obtained values of parameter Y_{sw} for the ADC-switch-based full-bridge inverter model, using the ABC, FFA, and GA algorithms with a particles band between 0 and 1.

ADC Parameters	Y_{sw}	Initial Population	Iteration
Optimum value by (ABC, FFA, GA)	0.1	400	1000

Table 2. Algorithms performance comparison while locating the conductance optimal value for the full-bridge inverter.

Algorithm	Y_{sw}	Number of Iterations	Computation Time	Convergence Rate
ABC	0.10098	1500	1.3557	0.0011667
	0.1	1000	1.8056	0.00094392
	0.09986	500	2.208583	0.00074035
FFA	0.1	1500	70.9361	0.00012665
	0.1	1000	92.205	0.00010374
	0.09966	500	115.385	0.000077332
GA	0.1085	1500	0.763534438	0.004452068
	0.1	1000	1.0306	0.0037614
	0.0999	500	1.3414	0.003032621

Figure 8 show the presence of two light-blue poles on the z-plane within the positive and negative Imaginary axis with non-optimal settings resulting in an overshoot. On the other hand, the green poles located on the Real axis were associated with ideal parameters.

The specified region within the complex plane might be considered an approximate area. Therefore, not all poles were expected to be situated in a precise and specific location.

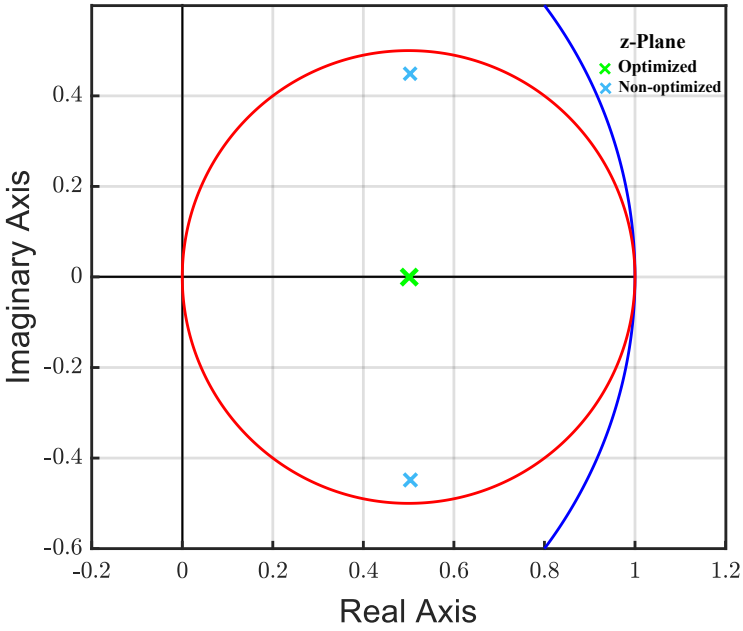

Figure 8. Pole locations of full-bridge inverter with Optimum and non-optimum Y_{sw} values, in s-plane left-hand side mapped onto the z-plane (red circle), inside the stability region of a discrete system (blue circle).

5.2. Buck Converter

As shown in Figure 9, a buck converter was modelled using ADC, a switch ($T1$), and a diode ($D1$), with no dead time. We can use Equation (24) to represent the buck converter state-space matrix, capturing the switches in the ON state and OFF state. In the steady-state analysis, there were two cases: the first case included the upper switch ($T1$) to be turned on and the diode ($D1$) turned off; in the second case, the upper switch ($T1$) to be turned off and the diode ($D1$) to be forward biased, which signified, in ADC modelling terms, that the model would change from capacitor to inductor, to indicate that the diode was in the ON state and, as stated previously, a virtual power loss would take place. The state-space matrix of the buck converter is shown in the following equations, where the discrete-state-space matrix can be obtained based on the variable Y_{sw}, as stated by backward Euler, as in Equation (25). Figure 10 shows the circuit's output voltage and current, with non-optimal conductance value.

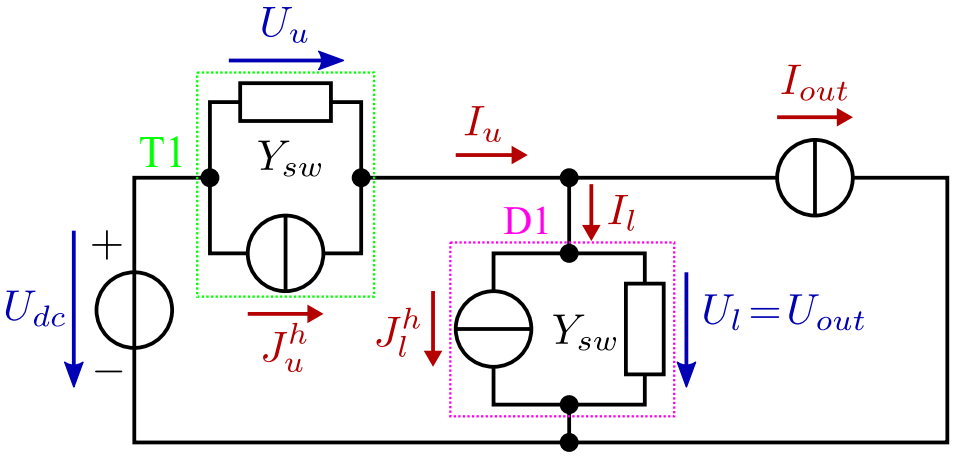

Figure 9. Simplified ADC circuit model of the switching components of the buck converter.

$$A = \begin{bmatrix} 0 & \frac{-1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix} \tag{24}$$

$$A^{'} = \begin{bmatrix} 1 & Y_{sw} \\ \frac{-1}{Y_{sw}} & 1 + \frac{1}{Y_{sw}R} \end{bmatrix}^{-1} \tag{25}$$

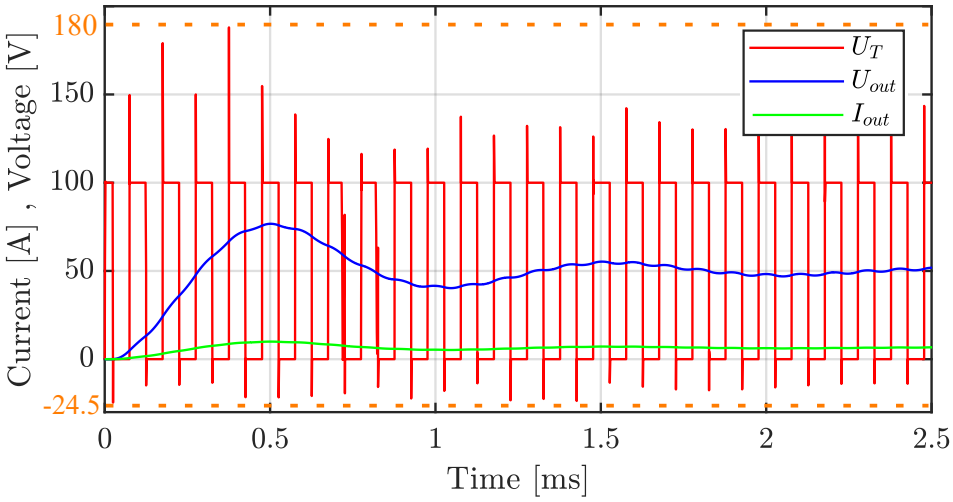

Figure 10. Transistor voltage, output current, and voltage of the buck converter while admittance value $Y_{sw} = 0.1$.

Table 3 shows the optimized obtained values of parameter Y_{sw} for the ADC-switch-based model of the buck converter using the ABC, FFA, and GA algorithms. Figure 11 shows the circuit's output voltage and current, with the optimal values achieved by all three algorithms. Table 4 gives a performance comparison of the algorithms while locating the conductance optimal value for the buck converter. Once again, the ABC is set in the middle, regarding the computational-time and convergence-rate values, while the FFA was the slowest to complete the process to locate the optimized value, and the GA was the fastest to converge to the optimal value. The case remained the same, even when changing the number of iterations.

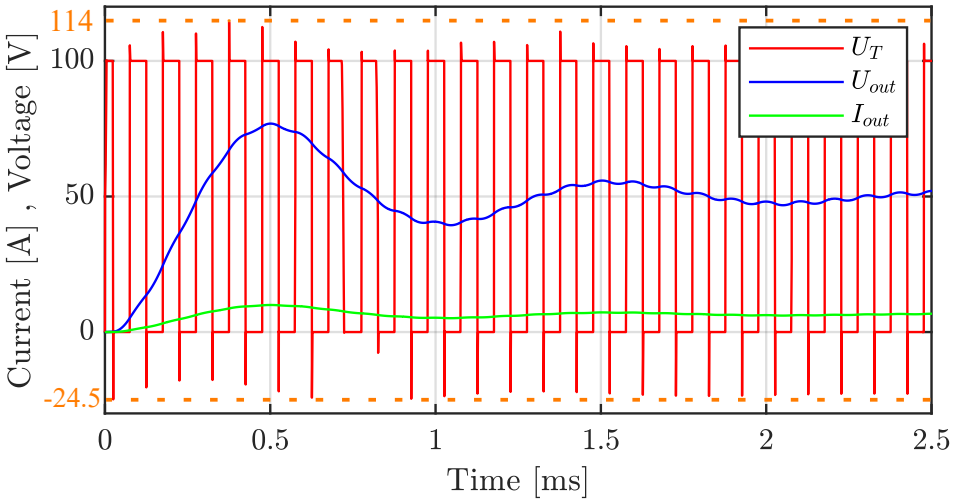

Figure 11. Transistor voltage, output current, and voltage of the buck converter while admittance value $Y_{sw} = 0.5$.

The values for percentage overshoot (PO) and settling time are provided in Table 5. When both systems were configured with optimum values (Y_{sw} = 0.1) for the full-bridge inverter and (Y_{sw} = 0.5) for the buck converter, the full-bridge circuit exhibited reduced overshoot and settling time for both the output voltage and current compared to the arbitrarily chosen non-optimal parameters. The outcome obtained through the utilisation of these optimal values bears a resemblance to an inverter circuit with ideal switch behaviour. The buck-converter circuit exhibited significantly reduced overshoot and settling time of the transistor voltage, with a percentage overshoot reduction of around 64% and settling time decreased from 1.5 µs to 0.5 µs, which brought the buck converter to more realistic behaviour.

Table 3. The obtained values of parameter Y_{sw} for the ADC-switch-based model of the buck converter, using the ABC, FFA, and GA algorithms with a particles band between 0 and 1.

ADC Parameters	Y_{sw}	Initial Population	Iteration
Optimum value by (ABC, FFA, GA)	0.5	400	1000

Table 4. Algorithms performance comparison while locating the conductance optimal value for the buck converter.

Algorithm	Y_{sw}	Number of Iterations	Computation Time	Convergence Rate
ABC	0.5008	1500	1.3995	0.00042871
	0.5	1000	1.8247	0.0003486
	0.5	500	2.2319	0.00028385
FFA	0.501	1500	170.87	0.000294945
	0.5	1000	219.39	0.00024274
	0.4999	500	274.54	0.00019066
GA	0.4999	1500	0.8861	0.009874301
	0.5	1000	1.1734	0.0084309
	0.4999	500	1.5272	0.006798

Table 5. The percentage overshoot and settling time for the ADC-based switch model optimal and non-optimal parameter Y_{sw}

Topology	Y_{sw}	Overshoot %	Settling Time µs
Full-bridge inverter			
Optimal	0.1	≈0	0.9
Non-optimal	1	16	1.1
Buck converter			
Optimal	0.5	14	0.5
Non-optimal	0.1	80	1.5

More test cases need to be conducted to confidently generalize this method. Although, based on the literature and the result in this paper, we can apply the method to not only power converters but also to transmission lines [10]. The limitation to this method is the complexity of the system, as it will be harder to obtain the system's state-space matrix with ADC implementation and to move forward with the optimisation.

6. Conclusions

This paper presents a novel approach to selecting appropriate switch settings for the ADC-switch-based model, widely used in the simulation of real-time power-electronics converter circuits. This paper thoroughly examined and analysed the two key aspects crucial to determining the optimal selection of the switch parameters. One of the primary considerations was to reduce the nearness of the eigenvalues of the state-space matrix system to the origin point (0,0) on the z-plane. This practice was implemented in order

to reduce the settling time required by the system to arrive at a condition of stability. The second factor referred to the angle reduction formed by the system matrix's eigenvalues. This action was undertaken to increase the system's damping factor. The artificial bee colony, firefly, and genetic algorithms were utilised to effectively place the eigenvalues of the system's discrete model within the specified region on the z-plane. It was observed that the genetic algorithm was the fastest algorithm to converge and give an optimal solution, while the firefly algorithm needed the most computational time to convergence. The artificial bee colony showed balanced behaviour between computation time and convergence rate, even with changing of the number of iterations to effect the computational time and convergence rate. The proposed technique was validated by employing full-bridge-inverter and buck-converter topologies. The simulations demonstrated that the proposed methodology effectively mitigates overshoot and reduces the output voltage and current settling time in an ADC-based switch model full-bridge inverter and buck converter.

Author Contributions: Conceptualisation, S.A.; data curation, S.A.; formal analysis, S.A. and Z.S.; funding, Z.S. acquisition, S.A. and Z.S.; investigation, S.A. and Z.S.; methodology, S.A.; project administration, S.A.; resources, S.A. and Z.S.; supervision, Z.S.; validation, S.A. and Z.S.; visualisation, S.A.; writing—original draft preparation, S.A.; writing—review and editing, S.A. and Z.S. All authors have read and agreed to the published version of the manuscript .

Funding: The research reported in this paper is part of project no. BME-NVA-02, implemented with the support provided by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021 funding scheme.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to ongoing research in the area of power converters real-time simulation. .

Acknowledgments: The research reported in this paper was carried out at the Department of Automation and Applied Informatics, Budapest University of Technology and Economics.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhang, B.; Hu, R.; Tu, S.; Zhang, J.; Jin, X.; Guan, Y.; Zhu, J. Modeling of Power System Simulation Based on FRTDS. *Energies* **2018**, *11*, 2749. [CrossRef]
2. Guo, X.; Yuan, J.; Tang, Y.; You, X. Hardware in the Loop Real-Time Simulation for the Associated Discrete Circuit Modeling Optimization Method of Power Converters. *Energies* **2018**, *11*, 3237. [CrossRef]
3. Futo, A.; Kokenyesi, T.; Varjasi, I.; Suto, Z.; Vajk, I.; Balogh, A.; Balazs, G.G. Real-Time HIL Simulation of the Discontinuous Conduction Mode in Voltage Source PWM Power Converters. *J. Power Electron.* **2017**, *17*, 1535–1544. [CrossRef]
4. Wang, K.; Xu, J.; Li, G.; Tai, N.; Tong, A.; Hou, J. A Generalized Associated Discrete Circuit Model of Power Converters in Real-Time Simulation. *IEEE Trans. Power Electron.* **2019**, *34*, 2220–2233. [CrossRef]
5. Karaagac, U.; Mahseredjian, J.; Cai, L.; Saad, H. Offshore Wind Farm Modeling Accuracy and Efficiency in MMC-Based Multiterminal HVDC Connection. *IEEE Trans. Power Deliv.* **2017**, *32*, 617–627. [CrossRef]
6. Biolkova, V.; Kolka, Z.; Biolek, D. State-space averaging (ssa) revisited: On the accuracy of ssa-based line-to-output frequency responses of switched dc-dc converters. *WSEAS Trans. Circuits Syst. Arch.* **2010**, *2*, 1.
7. Lin, N.; Dinavahi, V. Dynamic Electro-Magnetic-Thermal Modeling of MMC-Based DC–DC Converter for Real-Time Simulation of MTDC Grid. *IEEE Trans. Power Deliv.* **2018**, *33*, 1337–1347. [CrossRef]
8. Lamo, P.; de Castro, Á.; Brañas, C.; Azcondo, F.J. Emulator of a boost converter for educational purposes. *Electronics* **2020**, *9*, 1883. [CrossRef]
9. Terlizzi, C.; Bifaretti, S.; Lampasi, A. Current Sharing Control Strategy for parallel-connected H-Bridges DC-DC Converter: Modelling, Analysis and HIL test. In Proceedings of the 2021 IEEE Energy Conversion Congress and Exposition (ECCE), Virtual, 10–14 October 2021; pp. 2777–2783.
10. Rezaei Larijani, M.; Zolghadri, M.R. Design and implementation of an ADC-based real-time simulator along with an optimal selection of the switch model parameters. *Electr. Eng.* **2021**, *103*, 2315–2325. [CrossRef]
11. Alsarayreh, S.; Sütő, Z. Power Converter Half-bridge Models in the Light of Real-time FPGA Implementation. In Proceedings of the 2021 IEEE 19th International Power Electronics and Motion Control Conference (PEMC), Gliwice, Poland, 25–29 April 2021; pp. 153–160. ISSN 2473-0165. [CrossRef]
12. Ogata, K. *Modern Control Engineering*; Pearson: London, UK, 2016.

13. Larijani, M.R.; Zolghadri, M.R.; Shahbazi, M. Design and implementation of an FPGA-based Real-time simulator for H-Bridge converter. In Proceedings of the 2016 7th Power Electronics and Drive Systems Technologies Conference (PEDSTC), Tehran, Iran, 16–18 February 2016; pp. 504–510. [CrossRef]
14. Saad, A.E.H.; Dong, Z.; Karimi, M. A Comparative Study on Recently-Introduced Nature-Based Global Optimization Methods in Complex Mechanical System Design. *Algorithms* **2017**, *10*, 120. [CrossRef]
15. Oh, S.; Yoon, J.; Choi, Y.; Jung, Y.A.; Kim, J. Genetic Algorithm for the Optimization of a Building Power Consumption Prediction Model. *Electronics* **2022**, *11*, 3591. [CrossRef]
16. Wu, P.; Xu, J.; Wang, K.; Li, Z.; Li, G. A fractional time-step simulation method suitable for the associated discrete circuit model of power electronic system. *IET Renew. Power Gener.* **2023**, *17*, 176–185. [CrossRef]
17. Márquez, A.E.; Expósito-Izquierdo, C. An Overview of the Last Advances and Applications of Artificial Bee Colony Algorithm. In *Handbook of Research on Soft Computing and Nature-Inspired Algorithms*; Shandilya, S.K., Shandilya, S., Deep, K., Nagar, A.K., Eds.; IGI Global: Hershey, PA, USA, 2017; pp. 520–540. [CrossRef]