# Pulsewidth Modulation-Based Algorithm for Spike Phase Encoding and Decoding of Time-Dependent Analog Data

Ander Arriandiaga, Eva Portillo, Josafath Israel Espinosa-Ramos, and Nikola K. Kasabov, *Fellow, IEEE*

***Abstract*— This article proposes a new spike encoding and decoding algorithm for analog data. The algorithm uses the pulsewidth modulation principles to achieve a high reconstruction accuracy of the signal, along with a high level of data compression. Two benchmark data sets are used to illustrate the method: stock index time series and human voice data. Applications of the method for spiking neural network (SNN) modeling and neuromorphic implementations are discussed. The proposed method would allow the development of new applications of SNNs as regression techniques for predictive time-series modeling.**

***Index Terms*— Analog data, data compression, spike encoding, spike series decoding, spiking neural networks (SNNs), streaming data.**

## I. Introduction: A Review of the Methods for Encoding of Analog Signals Into Spike Sequences

THE basic idea behind artificial neural networks (ANN) such as the perceptron neuron is basically a computing system, whose central theme is borrowed from the analogy of biological neural networks [1]. However, real biological neurons communicate with each other using electrical pulses called "spikes" [2]. A chain of spikes emitted by a single neuron is called a spike train; a sequence of stereotyped events occur at regular or irregular intervals [3]. Since all spikes of a given neuron look similar, it is assumed that the form of the spikes does not carry any information, so it is the number and the timing of spikes that matter. The action potential or spike is the elementary unit of signal transmission [3]. Therefore, based on this idea, spiking neuron models were proposed in [4] and [5]. Actually, in spiking neural networks (SNNs), the "information" is transmitted as temporal spike sequences.

Due to the temporal encoding of the spikes, SNNs inherently possess the capacity to manage temporal data, i.e., they are more suited for modeling dynamic data evolution. SNNs have been widely used for classifying temporal data such as ultrafast image recognition [6], image compression and reconstruction [7], detection and classification of visual objects [8], odor recognition [9], epilepsy detection [10], and speech recognition [11], to mention only few of them. Finally, more recently, a morphologic framework based on spiking neurons has been presented for modeling spatio-temporal data such as brain and multisensory environmental data, along with video and speech [12]. However, much less effort has been made in order to model dynamic evolutions such as one-step ahead, multistep ahead forecasting, or complete dynamic evolutions of future events. Only, few works of the same research team can be found in [13] and [14]. These works are based on the polychronization [15]. However, these works perform a classification task rather than a true forecasting task. Moreover, there is no analog data reconstruction, and the output resolution is dependent on the number of spiking neurons rather than on new and optimal encoding mechanisms.

The main reason is that so far there is not a proper method to encode analog data into spikes and reconstruct the original data precisely. Among different encoding methods used so far, two gained major attention: rate coding and temporal coding [16]. In rate coding, information is encoded by the number of spikes in a short-time moving window [17]. A common rate encoding of analog signals follows a Poisson distribution, where the firing rate is proportional to the amplitude of the analog signal within a short-time window. This encoding method was used for digit recognition [18] and [19]. However, Poisson distribution encoding is not recommended for real world applications due to its imprecision when mapping analog signals into spike trains [17]. Another encoding algorithm introduced was the Hough Spiker algorithm (HSA) [20]. The basic idea behind this algorithm is to try to do a reverse convolution of the stimulus by a finite-impulse response (FIR) reconstruction filter. The idea is that if the impulse response of the linear filter is smaller than or equal to the input, then there has to be a spike in order to reproduce the signal [21]. Based on the HAS, the Ben's spike algorithm (BSA) was proposed [21]. Like the HSA algorithm, this algorithm assumes the use of an FIR reconstruction filter. At every instant of time $\tau$, the algorithm calculates two error metrics. If the first error is smaller than the second minus a threshold, then produce a spike and subtract the filter from the input, else do nothing. In general, it is possible to reconstruct the original signal to

Manuscript received September 11, 2017; revised May 2, 2018, October 4, 2018, February 4, 2019, and August 30, 2019; accepted October 3, 2019. Date of publication November 14, 2019; date of current version October 6, 2020. This work was supported in part by UPV/EHU PPGA19/48. *(Corresponding author: Ander Arriandiaga.)*

A. Arriandiaga and E. Portillo are with the Department of Automatic Control and Systems Engineering, Faculty of Engineering, University of the Basque Country, 48080 Bilbao, Spain (e-mail: ander.arriandiaga@ehu.eus).

J. I. Espinosa-Ramos and N. K. Kasabov with the Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, Auckland 1010, New Zealand.

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2019.2947380

a certain degree using these algorithms. However, there is a significant lack of precision in the reconstruction of the analog signal. Another encoding method is based on the simple thresholding of the volume of the signal, used in several address-event representation (AER) protocols [22]. With this encoding algorithm, signal intensity changes over a given threshold are encoded as spikes, where "ON" and "OFF" events are dependent on the sign of the changes of the signal. However, if the signal intensity changes dramatically, it is hard to recover the original signal [12]. Based on the thresholding AER protocols, an adaptive threshold-based (ATB) encoding algorithm was proposed [23]. In ATB, the threshold was calculated using the mean and the standard deviation of the signal gradient and, thus, self-adapt to input signal changes.

Although rate encoding is widely used, recent experimental evidences have suggested that precise spike timing preserves and reveals information from the original signal that is not available in rate codes [24]–[27]. Temporal coding offers substantial benefits because it can use time as both communication and computation resource in SNNs [28]. One of the most frequently used approaches for time encoding is phase encoding because it is able to encode the analog signal with high spatial and temporal selectivity [29]. It is a slightly modified approach of time-to-first-spike coding where the measurement of the relative timing is based on periodic background oscillations in the considered neural system [30]. This encoding scheme could allow neurons to encode information that is not encoded in their firing rate using their temporal pattern of spikes [31]. Increased evidence shows that phase encoding is also used in biological neurons [31], [32].

In [29], a phase encoding method was proposed using gamma alignment. First, with latency encoding, the analog signal is encoded into spikes in the encoding layer. Then, the spikes are aligned to the nearest subthreshold membrane potential oscillations (SMOs). With the gamma alignment, it was possible to reconstruct the original signal. However, due to the alignment, there is an error during the reconstruction of the original signal. Based on the SMO, another encoding algorithm was proposed [33]. In this case, instead of one neuron input, three neurons were used: one positive neuron, one negative neuron, and one output neuron. With this approach, the alignment phase is avoided. However, the decoding phase is based on the sequence recognition instead of using the oscillation signal (SMO) for reconstructing the signal. In fact, the decoding process is quite similar to the approach proposed in [13]. In another different approach, leaky integrate-and-fire (LIF) neurons were suggested for phase encoding [34]. However, in comparison to the time constant of the LIF neuron, the input analog signal should be quasi-static [17]. In order to overcome this drawback, the wavelet decomposition data preprocessing was proposed [17]. The decomposed wavelet spectrum amplitude was encoded into synchronized spike trains. However, in both cases that use the LIF neuron for phase encoding, the works do not show if it is possible to reconstruct an analog signal. In addition, in almost all works studied the encoding is carried out in a neuron layer with the consequent increase of neurons in the SNN. Moreover, these encoding algorithms cannot be used for forecasting because they do not ensure that there is a spike every time step of the time series.

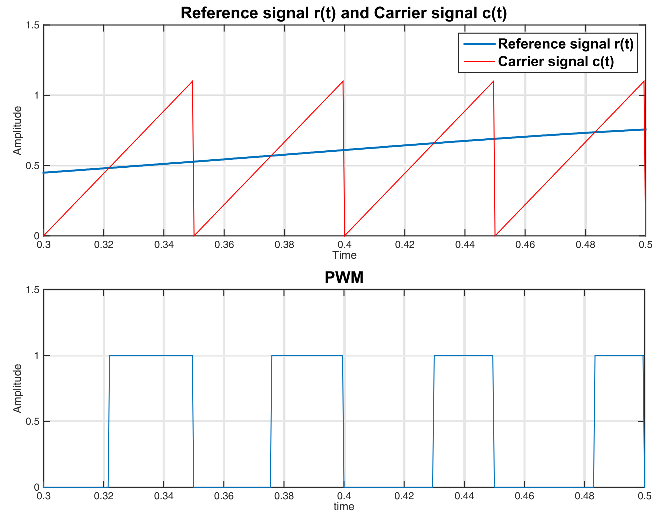


Fig. 1. PWM example.

In this article, a new phase-encoding algorithm is proposed based on the well-known pulsewidth modulation (PWM) for encoding analog signals into spike sequences. It should be noted that it is not the objective of this article to develop a biologically plausible encoding algorithm but develop an encoding algorithm for spiking neurons with capabilities to be used for practical regression applications. The presented approach does not need complex mathematical algorithms and enables an easy hardware implementation to be used in neuromorphic hardware such as SpiNNaker [35] or TrueNorth [36]. In addition, in order to be used for modeling dynamic evolutions of signals and for forecasting of the signal in future times, the proposed algorithm can reconstruct precisely the encoded analog signal, and the method is invariant to the frequency components of the signal.

## II. PWM-Based Spike Encoding–Decoding Algorithm for Analog Data

### A. PWM Principles

PWM is one of the most commonly used techniques to perform analog-to-digital conversion in applications of diverse areas, including: motor control, signal processing, communication, and power electronics [37]. It is a fundamental technique used for controlling power electronic circuits [38].

PWM is in itself a modulation technique used to encode a reference signal $r(t)$ into a pulsing signal that can be produced simply by comparing the reference signal $r(t)$, with a carrier signal, $c(t)$ that is commonly represented by a sawtooth. The binary PWM output can be mathematically written as follows:

$$b_{\text{pwm}}(t) = \text{sgn}[c(t) - r(t)] \quad (1)$$

where "sgn" is the sign function.

As illustrated in Fig. 1, if the reference signal amplitude is higher than the carrier signal amplitude, the modulated signal is represented with a high-amplitude rectangular pulse. On the contrary, if the reference signal amplitude is lower than the

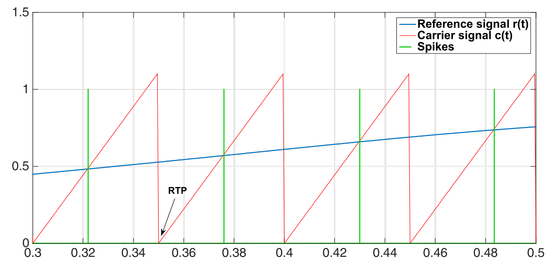


Fig. 2. PWM-based encoding example.

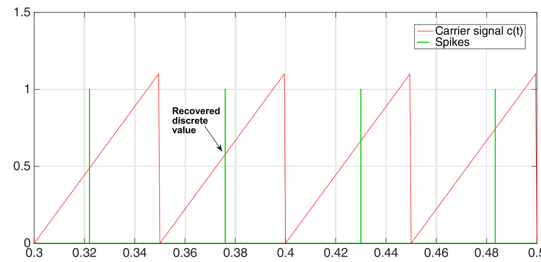


Fig. 3. PWM-based decoding example.

carrier signal amplitude, the modulated signal is represented with low amplitude. Therefore, the modulated signal is a quadratic signal with different pulsewidths.

### B. Proposed PWM Spike Encoding Algorithm

As shown in Fig. 1, the higher the amplitude of the reference signal, the narrower the generated pulse is, which means that the amplitude of the reference signal is somehow encoded in the time domain. Thus, it is possible to use the idea behind PWM to establish a new encoding method within the temporal paradigm just by considering that each rising edge of the PWM quadratic signal represents one spike or, in other words, to generate spikes in the intersections between the reference signal $r(t)$ (which is indeed the signal to be encoded) and the carrier signal $c(t)$, as shown in Fig. 2.

It can be noticed that in this new encoding algorithm, the spikes are generated in respect to a reference time point (RTP), which is the 0-value point of the sawtooth, so as the lower the amplitude of the reference signal is, the closer the spike to RTP. In the same way, the higher the amplitude, the farther the spike is generated from the RTP. With this method, it is very easy to reconstruct the original signal by just doing the opposite: The original values of the signal are given by the intersections between the carrier signal and the corresponding spikes (note that the same carrier signal, used for encoding, is also used for decoding). Once the original discrete values are recovered, the complete original signal can be reconstructed by interpolation. In order to illustrate this encoding algorithm, Fig. 3 highlights the recovered points during the reconstruction process, and Fig. 4 summarizes the encoding and decoding processes through the PWM-based encoding algorithm. In Figs. 5 and 6, the pseudocode for encoding and decoding is shown. Note that this new simple method makes possible to fire one spike at each time step, which is an essential issue in time-series forecasting, i.e., stock exchange close price every day, hourly mean temperature, and monthly unemployment. Therefore, this new and simple method covers an important gap in the state of the art concerning SNN.

For on-line use purposes, it should be noted that the hardware PWM is a very well-established technology, making it feasible to adapt it to the PWM-based encoding algorithm in order to directly sample the analog signals and generate the corresponding spike trains. Actually, a simple solution can be the use of a microcontroller so that the input analog signals are captured through ADCs, then, the PWM-based algorithm applied, and finally, the spikes transmitted to digital outputs.

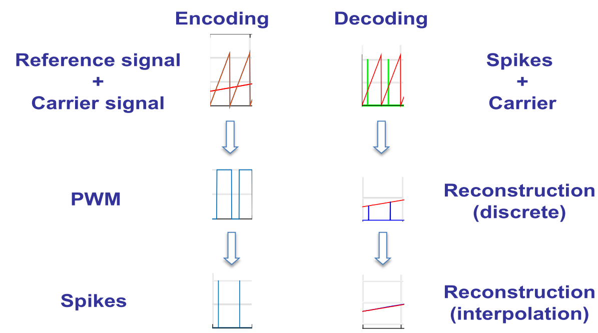


Fig. 4. Graphical representation of the PWM-based algorithm for spike encoding and decoding of analog signal.

```
for t=1 to end
    if  c(t) > r(t)
        pwm(t) = 1
    else
        pwm(t) = 0
end
for t=1 to end
    if ((pwm(t) == 1) && (pwm(t-1) == 0
        spikes(t) = 1;
     else
        spikes(t)=0;
end
```

Fig. 5. Algorithm for encoding analog signals into spikes.

```
for t=1 to end
    if  c(t) == 1
        r(k) = c(t)
        k=k+1
end
CurveFitting (x(k)) → O(t)
```

Fig. 6. Algorithm for decoding the spike trains into analog signals.

### C. Parameter Selection for the Proposed PWM-Based Encoding–Decoding Algorithm

There are two signals or time evolutions involved in the proposed encoding algorithm: the reference signal $r(t)$, which is the signal to be encoded, and the carrier signal $c(t)$. The reference signal is defined by the application problem. However, some decisions should be made about the parameters of the carrier signal $c(t)$. Those decisions should be taken, presumably, considering the characteristics of the signal to be encoded, i.e., the reference signal $r(t)$. Thus, first of all, the objective is to identify the parameters that characterize the

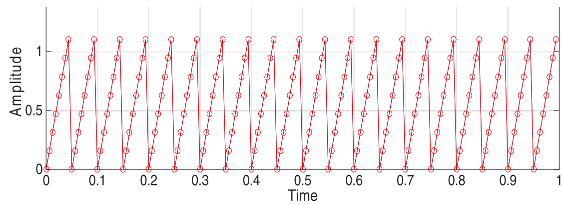


Fig. 7. Carrier signal example. Parameters: nc = 20 and npc = 8.

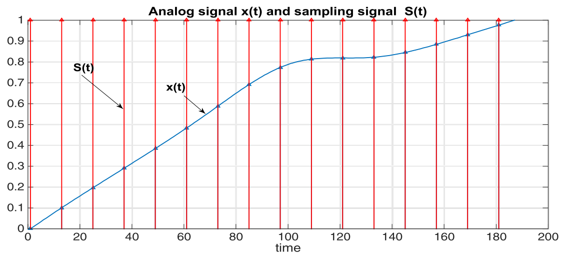


Fig. 8. Classical sampling.

carrier signal, and second, some criteria and hypothesis are established about the values of those parameters.

There are two important parameters that have a great influence over the reconstruction accuracy: 1) the number of carrier (nc) waves, which is directly related to the carrier pulsewidth and 2) the number of points per carrier (npc) wave.

For instance, the carrier signal shown in Fig. 7 consists of 20 carrier waves of 0.05 units of time each, and the npc wave used to encode the reference signal is 8. In this case, the values of the parameters of this carrier signal are as follows.

1) nc = 20.
2) npc = 8.

As stated above, it is reasonable to think that proper values of nc and npc parameters can be established depending on the characteristics of the signal to be encoded $r(t)$. Actually, in the case of the nc waves, the sampling rate (the sampling requirements) of the original data $r(t)$ can be taken into consideration. As it is well-known, the Nyquist–Shannon sampling theorem states that a bandlimited baseband $x(t)$ within the frequency bandwidth $B$ can be exactly reconstructed from its sample values by low-pass filtering if the sampling rate is higher than $2B$ [39]. A classical sampling is a process of multiplying the analog signal $x(t)$, with a sampling signal $s(t)$, which is a train of impulses (delta dirac), where one value is evenly captured per impulse (see Fig. 8).

PWM, on the other hand, represents a signal by using pulses of constant amplitude but variable widths. In this sense, PWM is a substitute for classical sampling [40]. Making an analogy with classical sampling, in which "one value is captured per impulse," in the PWM-based encoding algorithm, "one spike is generated per carrier pulse." In other words, and conceptually speaking, the sampling signal in the PWM-based encoding algorithm is the carrier instead of the train of impulses (see Fig. 9).

Taking into account that a proper selection of the sampling rate is essential to guarantee a proper signal reconstruction, the nc waves (i.e., the width of the carrier pulse) should be directly related to the sampling requirements of the signal to be encoded. At this point, it is important to highlight that

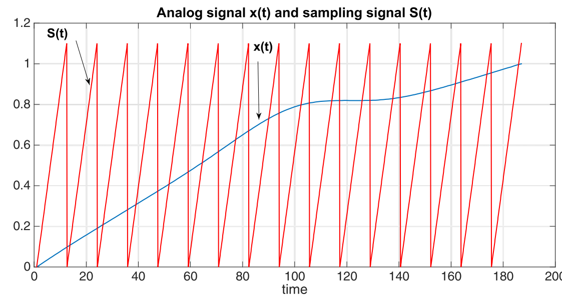


Fig. 9. PWM-based encoding "sampling."

in most cases, the available data to train SNNs are already digitized and stored in a computer system. The assumption made here is that the original data to be encoded have been acquired at a properly selected sampling rate (if more points than necessary are acquired, well-known methods to reduce the number of points could be applied; in the case where data have been acquired at an insufficient sampling rate, no matter which technique is employed to process the data, the results may not be adequate). Our hypothesis is that if nc is equal to the number of points acquired (minus one), this should provide a satisfactory recovery accuracy.

For the task of forecasting applications, it is necessary to have one value at each time step. With the state-of-the-art encoding methods presented in the Introduction, it was not possible to perform true forecasting with SNN because those methods cannot generate one spike per time step. By contrast, using the encoding method proposed here, it is possible to generate one spike at each time step by setting nc equal to the number of time steps (minus one) of the time series.

Regarding the decision about the npc wave, another important parameter in classical sampling can be taken into consideration: the resolution. As is well known, the resolution is the smallest detectable change in the signal and has a direct impact on the recovery accuracy. Making again an analogy with classical sampling, in which the higher the resolution is, the better the reconstruction can be achieved, since smaller changes in the value of the signal are captured, in the case of the proposed PWM-based encoding algorithm, the smallest detectable change is determined by the number of points within the carrier pulse. To illustrate this idea, in Figs. 10 and 11, a reconstruction is shown for different number of points (npc) with the same nc waves. Obviously, the hypothesis in this case is that the more the npc, the better is the accuracy; Figs. 10 and 11 show that when the npc is higher, the accuracy of the reconstruction increases. Of course, this means that depending on the application requirements, the lowest possible resolution should be selected so as not to increase excessively the number of points of the carrier.

## III. Experimental Results on Spike Encoding and Decoding of Stock Exchange Time-Series Data, Human Voice Data, and EEG Data

In order to demonstrate the proposed PWM-based spike encoding–decoding method, two benchmark data sets are used: stock exchange time series and human voice records

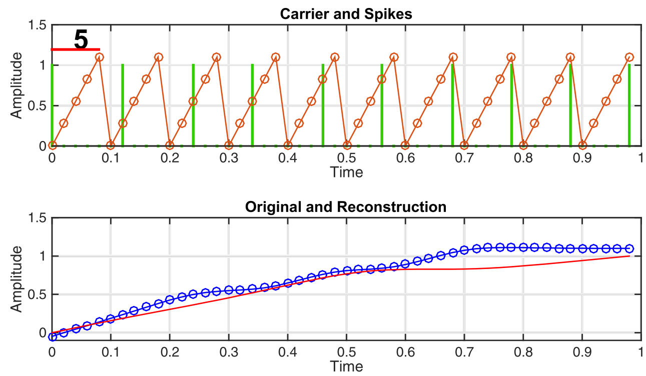


Fig. 10. Original analog signal ("-" red) and reconstruction signal ("-o" blue) with npc = 5.

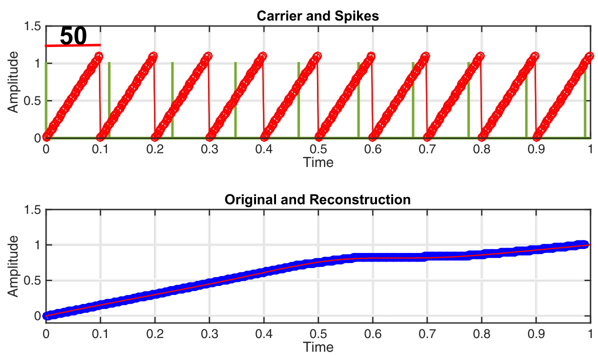


Fig. 11. Original analog signal ("-" red) and reconstruction signal ("-o" blue) with npc = 50.

TABLE I

DATA USED IN THE EXPERIMENTS

| Time dependent sequence | Sample rate | Number of points |
|---|---|---|
| Stock exchange time series | 1 data per day | 369 |
| Human voice record | 32 kHz | 112960 |

TABLE II

nc WAVES AND npc WAVE USED TO ANALYZE THE PROPOSED METHOD FOR ENCODING–DECODING OF STOCK TIME SERIES

| IBM closing stock price | | | | | |
|---|---|---|---|---|---|
| number of carrier waves (*nc*) | 46 | 92 | 184 | 368 | 736 |
| number of points per carrier wave (*npc*) | 32 | 64 | 128 | 256 | 512 |

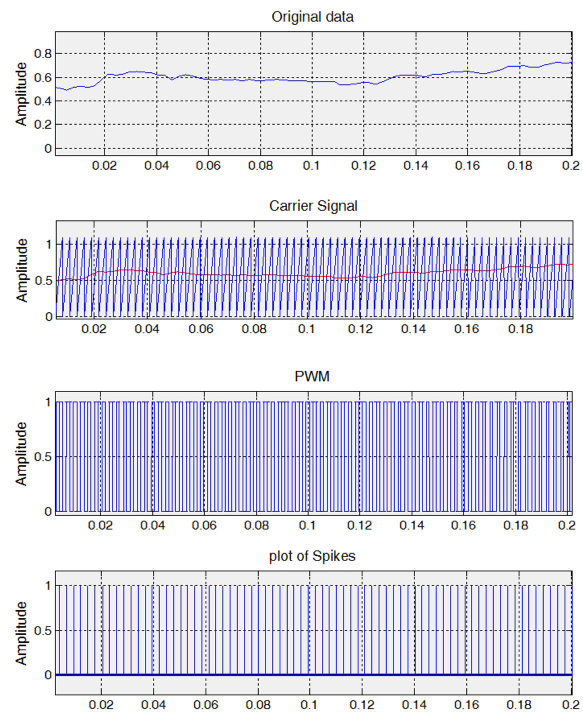


Fig. 12. Encoding process of the IBM stock time-series signal into spikes for nc = 184 and npc = 64. For a better understanding of the process, only the first 20% of the used data are shown.

(see Table I). These data sets have different sampling rates to see the effect of the nc waves and the npc wave over time-dependant sequence reconstruction.

### *A. Stock Exchange Time Series*

As a time-series benchmark data set, the well-known IBM closing stock price is used. This time series represents the common daily closing stock price of IBM from May 17, 1961 to November 2, 1962. One value per day is collected, and it consists of 369 points. This benchmark is widely used for time-series forecasting and has been also used for forecasting with SNN [13]. As mentioned in Section II, in forecasting, at each time step, it is necessary to have one value; thus, the nc waves must be set to 368. In order to analyze the effect of nc and npc values, a different combination of them is also studied (see Table II).

In Fig. 12, the encoding process is shown for the IBM time series with nc = 184 and npc = 64 for the first 20% of the signal. Only initial 20% is shown to illustrate better the encoding and decoding processes. The graph above shows the original data scaled within the range [0, 1]. As explained in Section II, this is done because the sawtooth wave is generated within that range, as shown in the second graph (so-called "Carrier Signal"). Then, comparing the original signal and the sawtooth wave, the quadratic signal is generated (third graph). Finally, the spike train is generated based on the rising edges of the quadratic signal (fourth graph).

In Fig. 13, the reconstruction of the original signal from Fig. 12 is shown. The spike train used for the reconstruction is the same one generated and shown in Fig. 12. In the middle, the sawtooth wave signal is represented, and on the bottom, both the analog signal and the reconstructed one after the spike encoding–decoding are shown. Although the npc value is one of the lowest considered, the reconstruction is excellent (see the graph below in Fig. 13). Moreover, the mean square error (MSE) for the reconstruction of the original signal with nc = 184 and npc = 64 is lower than U.S. $50 (see Fig. 14).

In Fig. 14, the MSE values for different nc waves and npc wave are shown. Analyzing the nc waves, one can see that

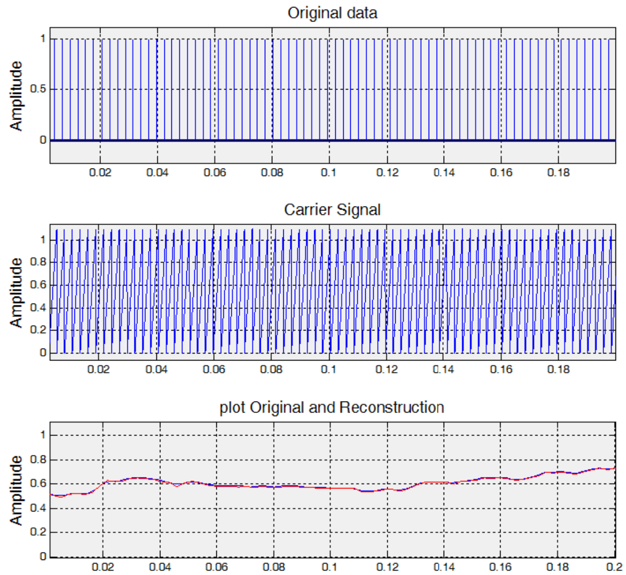


Fig. 13. Decoding process of the IBM time series for nc = 184 and npc = 64. For better understanding of the process, only the first 20% of the used data are shown.

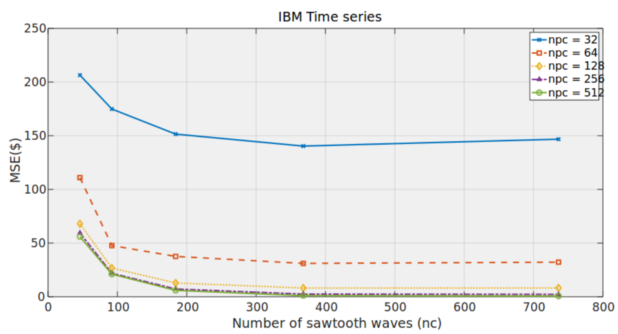


Fig. 14. MSE results of the reconstruction of the IBM stock time series for different nc = [46, 92, 182, 368, 736] and npc = [32, 64, 128, 256, 512].

increasing the nc waves improves the reconstruction results for all "resolutions" or npc. However, after 368 carrier waves, the improvement is minimal. Therefore, the results confirm the hypothesis of Section II: "nc equal to the number of points acquired (minus one) should provide a satisfactory accuracy." This means that using equal number of points of the original data, it is possible to use SNNs for forecasting because it ensures that there will be at least one spike at each time step.

Regarding the npc wave, the results are quite similar: At the beginning, the results improve dramatically increasing the resolution, but after npc = 128, the improvement decreases as we increase the npc. It is noteworthy that by increasing the npc, the number of points required for reconstruction and, consequently, the computational requirements increase. Therefore, when selecting the npc, as said in Section II, the lowest possible resolution should be selected depending on the application and the resolution requirements.

In this particular case, the MSE value yielded with nc = 368 and npc = 128 is satisfactory. Of course, with npc = 256 (see Fig. 15) and npc = 512, the accuracy of

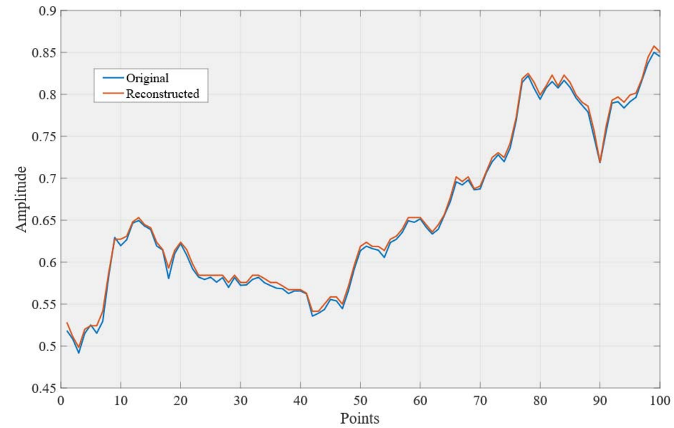


Fig. 15. First 100 points of the original and reconstructed signals of the IBM stock time series for nc = 368 and npc = 256.

TABLE III

AUDIO PROPERTIES OF THE HUMAN VOICE RECORDING DATA USED ("arctic_a0001.wav")

| | |
|---|---|
| Length | 3,53 seconds |
| Sample rate | 32kHz |
| Sample size | 16 bit |
| Bit rate | 1014 bps |
| Data points | 112960 |

TABLE IV

nc WAVES AND npc WAVE USED TO ANALYZE THE PROPOSED ENCODING METHODOLOGY FOR A HUMAN VOICE RECORDING

| Human voice recording | | | | | |
|---|---|---|---|---|---|
| number of carrier waves (*nc*) | 14120 | 28240 | 56480 | 112960 | 225920 |
| number of points per carrier wave (*npc*) | 32 | 64 | 128 | 256 | 512 |

signal reconstruction increases with a consequent increase in the number of points, double for npc = 256 and quadruple with npc = 512. It should be noted that in Fig. 15, the results are within the normalization range [0, 1].

### *B. Human Voice Data Encoding–Decoding*

The second time-dependent data set used is a human voice record from Carnegie Mellon University ARCTIC speech databases [41]. Each voice recording data is acquired with a sample rate of 32 kHz. From the data available, the so called "arctic_a0001.wav" file of the subfolder "US bdl (US male)" is used in this work. The audio properties of the file are shown in Table III.

Taking into account that the sampling rate of 32 kHz during 3.5 s corresponds to 112 960 data points, the following nc waves and npc are selected to compare the encoding and reconstruction of the signal, as shown in Table IV.

The MSE for the encoding and reconstruction of the human voice recording signal are shown in Fig. 16. After nc = 112960, the accuracy improvement is minimal which confirms the hypothesis of Section II. It means that increasing

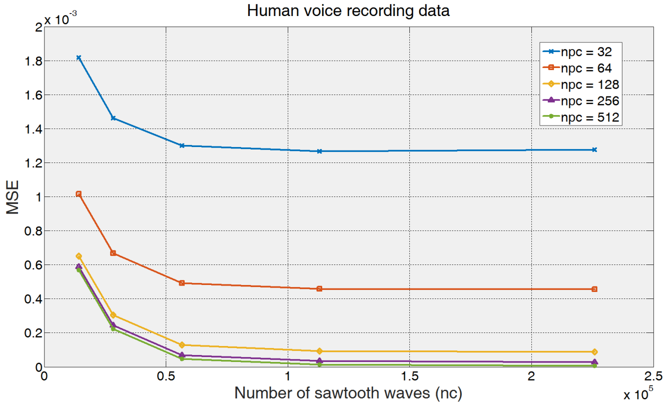


Fig. 16. MSE results of the reconstruction of the human voice recording time series for different nc = [14 120, 28 240, 56 480, 112 960, 225 920] and npc = [32, 64, 128, 256, 512].

TABLE V

MSE RESULTS FOR DIFFERENT nc AND npc WITH THE SAME NUMBER OF POINTS PER DATA

| Number of carrier waves (*nc*) | Number of points per carrier wave (*npc*) | Points | Mean Squared Error (MSE) |
|---|---|---|---|
| 112960 | 128 | 14458880 | 9.3732e-05 |
| 56480 | 256 | 14458880 | 6.7580e-05 |

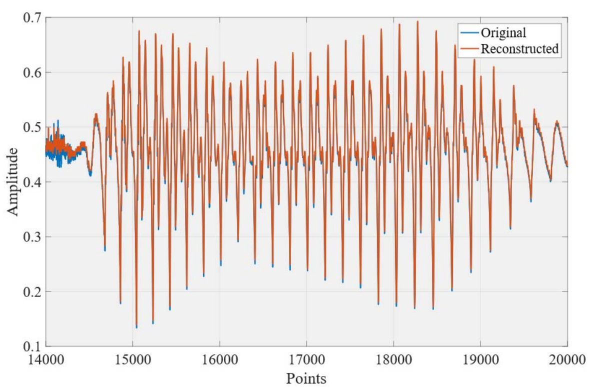


Fig. 17. 6000 points of the original and reconstructed signals of the human voice recording signal for nc = 56 480 and npc = 256. The results are shown within the normalization range [0, 1].

the nc waves beyond the number of points of the original signal increases the number of points in the encoded data, with no remarkable improvement in the reconstruction of the signal. Thus, it is not recommended to use more carrier waves than data points of the original signal.

For the IBM time-series forecasting problem above, at least one spike must fire in the output spiking neuron. However, in other applications, there may not be such a restriction, so the output time-dependent data could have fewer data points than the original one. In those cases, it could be possible to achieve higher reconstruction accuracy with another combination of nc and npc, being a combination close to the number of points of the original signal (see Table V). Although it is possible to reconstruct the original signal with higher accuracy, the improvement is not significant (see Fig. 17). Therefore, unless for high-accuracy applications with limited computational requirements, it is highly recommended to use nc waves equal to the number of points of the original data, only changing the npc wave based on the resolution needed in the reconstruction accuracy.

In order to show the effectiveness of the proposed algorithm for signal reconstruction, a comparison between the BSA and the proposed algorithm is carried out. Similar to the HSA algorithm, the BSA is strongly nonlinear. Therefore, it is not possible to use linearity properties and perform classical optimisation techniques for parameter optimization. For this purpose, a classical differential evolution (DE) is applied (DE/rand/bin/1) [42], an iterative heuristic continuous space optimizer, for fitting both the FIR filter and the threshold. The DE generates trial parameter vectors and creates new points that are perturbations of the existing points. It adds the weight difference between two randomly chosen vectors to a third vector. In this case, every candidate solution is an R3 vector whose elements represent values for the order of the filter, the cutoff frequency, and the threshold of the BSA.

The algorithm is initialized with a constant population of 30 vectors, which is ten times the dimensionality of the problem as suggested in [42]. The weighting factor is set to $f = 0.05$, and crossover rate toc $= 0.7$. Every vector competes against another vector with the same index in the current population to form the next generation of solutions. The fitness of every new vector is assessed using MSE between the original signal and the reconstructed one. In the presented case, the iterative process continued until the best candidate solution achieved the optimal value, MSE $= 0$, or until 100 iterations are complete.

The MSE result achieved for the reconstruction of the signal with the best candidate is 4.5782e-04 dB, far from the results yielded with the proposed algorithm (see Table V). In order to see and compare the effectiveness of the proposed methodology, original and reconstructed audio signals are provided as supplementary material.

### C. Training

In this section, an SNN is trained using DE. Optimization methods make possible to train small SNNs without computing the derivative of the spiking neuron as it is done with backpropagation (BP). The main objective is to show that is possible to train spiking neurons for forecasting using MSE as loss function and the PWM-based proposed algorithm.

SNNs with Izhikevich neurons [4] are used for four different forecasting scenarios: using 1 input and 1-step-ahead prediction, 2 inputs and 2-step-ahead prediction, 3 inputs and 3-step-ahead prediction, and 3 inputs and 5-step-ahead prediction. It is noteworthy that the inputs are delayed onstep; for example, for the 2 inputs 2-step-ahead prediction, two inputs ($t$ and $t - 1$) are used to predict $t + 2$. The Izhikevich neuron is configured using $a = 0.02$, $b = 0.2$, $c = -65$ and $d = 6$. To carry out this proof of concept, the IBM data set previously introduced is used. The data are preprocessed subtracting the previous value from each value in the data to remove the trend. Five synaptic weights in the

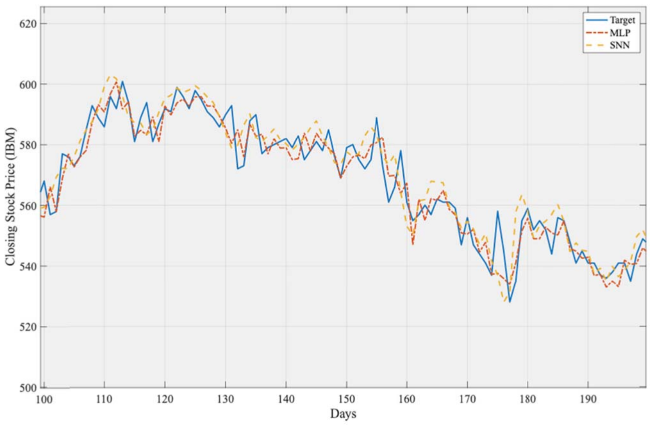


Fig. 18. Target (blue), MLP output (red), and spiking neuron output (orange) after training the MLP and spiking neuron using one input for forecasting IBM closing stock price for one day ahead.

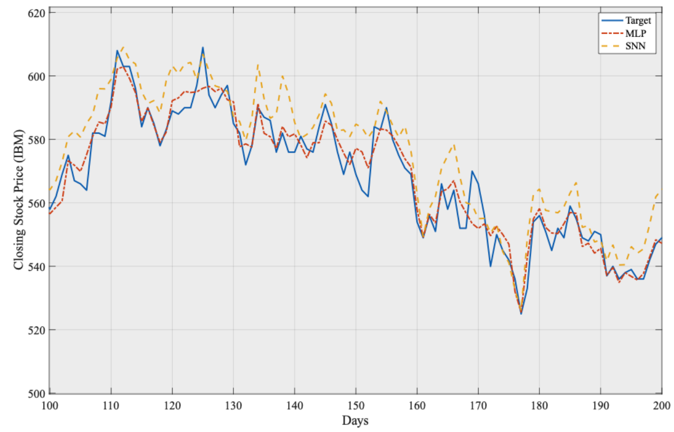


Fig. 19. Target (blue), MLP output (red), and spiking neuron output (orange) after training the MLP and spiking neuron using three inputs ($t$, $t-1$, and $t-2$) for forecasting IBM closing stock price for five days ahead ($t+5$).

TABLE VI

RMSE AND MAPE RESULTS FOR THE FOUR FORECASTING

| Forecasting | RMSE | | MAPE | | MAAPC | |
|---|---|---|---|---|---|---|
| | MLP | SNN | MLP | SNN | MLP | SNN |
| 1 input, 1 step-ahead | 7.833 | 9.453 | 6.80e-04 | 3.37e-03 | 1.366 | 2.048 |
| 2 inputs, 2 step-ahead | 6.901 | 10.363 | 2.33e-04 | 8.81e-03 | 2.137 | 2.242 |
| 3 inputs, 3 step-ahead | 6.973 | 11.194 | 6.55e-04 | 1.49e-02 | 2.408 | 3.185 |
| 3 inputs, 5 step-ahead | 6.886 | 29.033 | 7.89e-05 | 1.29e-02 | 3.02 | 3.576 |

hidden layer initialized to zero are used. One of the drawbacks of using optimization methods for training SNNs is that the more the synaptic weights the network contains, the more the parameters to optimize and the longer the training time.

The DE is configured with 40 (20 × parameters to optimize) population members, 0.8 step size, and 20 iterations. The two synaptic weights are optimized to minimize the MSE between the output of the spiking neuron and the ground truth or target. To do this, the output spikes are decoded using the proposed algorithm, therefore making it possible to compute the MSE.

To evaluate the performance of the spiking neuron, the results are compared with a multilayer perceptron (MLP) neural network. The MLP has two inputs, ten neurons in the hidden layer and one output. The Levenberg–Marquardt algorithm is used to train the network.

Root-mean-square error (RMSE), mean absolute percentage error (MAPE), and mean absolute average percentage change (MAAPC) (2) are used to compare the error of ANN and SNN for the four forecasting cases

$$\text{MAAPC} = 100\% \sum_{t=1}^{n} \left[ \frac{F_{t+s} - A_t}{A_t} \right] \tag{2}$$

where $F_{t+s}$ is the forecasted value $s$ steps ahead and $A_t$ is the real value.

The results show (see Table VI) that the performance of the SNN for the four forecasting cases gets worse when the complexity of the task increases. The best result is yielded with one input and one step-ahead prediction (RMSE equal to 9.453) and the worst with three inputs and five step-head prediction (RMSE equal to 29.033). This can also be observed by analyzing MAPE and MAAPC results. It can be noted that although it is possible to train SNNs for forecasting, the results are not yet close to those achieved with ANNs. However, given that in this work, an optimization method and five neurons in the hidden layer are used; there is still much room for improvement regarding the training algorithms for SNNs, which presumably will yield much better results.

The forecasting results are shown in Figs. 18 and 19 for the days between 100 and 200 using one input for forecasting IBM closing stock price: for one day ahead and three inputs ($t$, $t-1$, and $t-2$), and five days ahead ($t+5$) and also three inputs ($t$, $t-1$, and $t-2$), respectively.

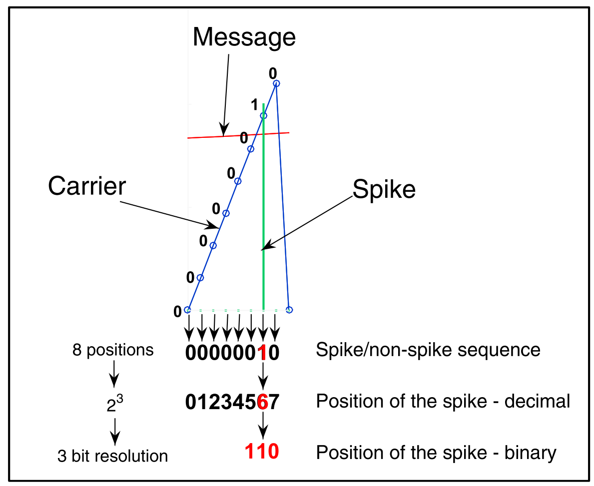


Fig. 20. Compressing the data required for PWM encoding.

## IV. LEVEL OF COMPRESSION ACHIEVED BY THE PWM-BASED ENCODING ALGORITHM

In view of the proposed PWM-based encoding method, it is evident that many bits would be required to store the encoded data, indeed, as many as the npc. Many applications will require the use of large data sets that often are generated remotely (e.g., in the cloud or in computer clusters), imposing a significant cost of transferring data. One of the most important approaches to deal with this bottleneck is to remove any redundancy in the data, e.g., using data compression [52]. Nevertheless, it is also important to highlight those scientific applications that work with large arrays of floating-point numbers demand compression methods that still preserve the data with high accuracy.

Data compression algorithms are commonly classified into either lossless or lossy. Lossless data compression involves a transformation of the original data set such that it is possible to reproduce exactly the original data set by the decompression process. In contrast, in lossy data compression, it is not possible to reproduce exactly the original data set, such that performing the decompression permits only an approximate representation to be recovered [53].

Most of the works on compression of double-precision data have focused on lossless compression because very high precision is common in high-accuracy demanding applications [52]. Most of the proposed methods use linear prediction and encode the smaller residuals using some variant of nonstatistical [54]–[56] or statistical [57]–[59] variable-length codes (e.g., entropy codes). Although important in many applications, lossless methods rarely achieve more than 1.5× compression on double-precision data and have only limited impact on bandwidth reduction [52].

Regarding lossy compression, one of the main research lines is focused on volume rendering within the visualization scope. Furthermore, it is unknown what the effect of these methods would be on nonvisual, quantitative tasks other than on volume rendering. In [43], an evaluation of lossy compression on a simulation task was performed. This work proposes a fixed-rate (fixed length bit stream) scheme for compressing 3-D arrays of double precision numbers, tailored to the high dynamic range and precision demands of scientific applications. Despite the method being lossy, the author claims that it allows the user to specify the exact amount of compression level, making possible to achieve a lossless mode.

Data compression ratio $C$ is defined as the ratio between the size of the original data and the size of the compressed data [52]

$$C = \frac{\text{Uncompressed size}}{\text{Compressed Size}}. \tag{3}$$

It can also be defined as the reduction $R$ in the original data quantity [53], given by

$$R = \frac{\text{Uncompressed size-Compressed Size}}{\text{Uncompressed Size}}. \tag{4}$$

For measuring the compression ratio of the proposed algorithm, as usual, the standard IEEE 754 is taken as baseline although the algorithm could also be applied to similar formats [60]. In this case, we consider the binary floating-point basic formats encoded with 32 and 64 bits, the well-known single-precision floating-point format (binary32), and double-precision floating-point format (binary64), respectively. The IEEE 754 standard specifies a binary32 as having one sign bit, an 8-bit exponent, and a mantissa with 23 bits. Regarding binary64, the standard specifies an 11-bit exponent and a mantissa with 52 bits.

TABLE VII

VARIATION OF $C$ AND $R$ WITH THE npc PULSE

| Number of points per carrier pulse (*npc*) | Resolution (bit) | | Data compression ratio C (N bit /resolution) | | Reduction in the original data R (1- resolution /N bit) | |
|---|---|---|---|---|---|---|
| | | | Binary32 | Binary64 | Binary32 | Binary64 |
| 16 | 4 | Lossy | 8 | 16 | 0.875 | 0.9375 |
| 32 | 5 | | 6.4 | 12.8 | 0.84375 | 0.921875 |
| 64 | 6 | | 5.33 | 10.66 | 0.8125 | 0.90625 |
| 128 | 7 | | 4.57 | 9.14 | 0.78125 | 0.890625 |
| 256 | 8 | | 4 | 8 | 0.75 | 0.875 |
| 512 | 9 | | 3.55 | 7.11 | 0.71875 | 0.859375 |
| 1024 | 10 | Lossless | 3.2 | 6.4 | 0.6875 | 0.84375 |

Regarding a single-precision and double-precision floating-point number, uncompressed size is 32 and 64 bits, respectively, and compressed size is the number of bits required by the proposed encoding algorithm. As stated above, the proposed PWM-based encoding method requires many bits to store the encoded data (as many as the npc). This means that this method would require, in general, more bits than the standard IEEE 754 to encode floating point numbers. However, this drawback can be easily solved by storing the position of the spike within the carrier pulse, instead of storing the whole sequence of bits. For instance, for npc = 8, 3 bits are needed to encode the position of the spike within the carrier pulse, i.e., the resolution of the encoding algorithm is 3 in this particular case (see Fig. 20). Thus, to take the most advantage of the binary encoding of the position of the spike within the carrier pulse, the npc should be the maximum allowed by the selected resolution, which is, after all, the corresponding power of two.

Under this approach, it can also be noticed that $C$ and $R$ only depend on the selected npc pulse, i.e., on the resolution. Obviously, the more the resolution, the less the compression ratio $C$ and the less reduction in the original data $R$, as illustrated in Table VII.

For example, in the case of the human voice record from the Carnegie Mellon University ARCTIC speech databases [41] (see Section III-B), the original size of each sample size is 16 bit (see Table III). Thus, if the selected npc is 512 (resolution = 9), $C$ is 1.77, and $R$ is 0.4375.

Actually, the proposed PWM base encoding method provides many of the advantages that are usually claimed for the compression algorithms [52].

1) *Simplicity:* This method is significantly far from the complicated mathematical definitions of other methods.
2) *signal-to-noise ratio (SNR) scalability; lossy → lossless:* It allows the user to specify the exact amount of compression (and, consequently, the quality). Furthermore, it depends on one unique eligible parameter, the selected resolution.

## V. CONCLUSION

In this article, we propose a new phase-encoding algorithm based on the well-known PWM for encoding analog signal into spikes following the assumption that for modeling dynamic evolution of analog using SNN, a precise reconstruction of the analog signal may be needed regardless of the frequency components of the signal.

There are two major applications of the proposed method:

1) analog data encoding for SNN and neuromorphic implementations;
2) data compression for remote communication systems.

More specifically, the following conclusions can be made.

1) A new phase-encoding algorithm is proposed based on the well-known PWM for encoding analog signal into spikes. The proposed algorithm can precisely reconstruct an analog signal no matter what the frequency components of the signal are, ensuring that for forecasting, at least one spike is emitted in each time step of the time series.
2) It is easily possible to use the idea behind PWM to establish a new encoding method within the temporal paradigm just by considering that each rising edge of the PWM quadratic signal represents one spike or, in other words, by generating spikes in the intersections between the reference signal and the carrier signal.
3) For a high reconstruction accuracy, two parameters have a great influence: 1) the nc waves, which is directly related to the carrier pulsewidth and 2) the npc wave.
4) The results confirm the hypothesis that the nc waves equal to the number of points acquired (minus one) should provide a satisfactory accuracy. This means that using an equal number of points of the original data, it is possible to use SNN for forecasting because this ensures that there will be at least one spike at each time step.
5) In applications, where the output time-dependent data could have fewer data points than the original one, it may be possible to achieve higher reconstruction accuracy with other combinations of nc and npc, being a combination close to the number of points of the original signal. However, unless for high-accuracy applications with limited computational requirements, it is highly recommended to use the nc waves equal to the number of points of the original, only changing the npc wave based on the resolution needed in the reconstruction accuracy. The reconstruction MSE for the two benchmark data used with different sample rates is very low and confirms that the encoding algorithm used is suitable for regression and forecasting with SNN. Thus, for the IBM stock time series, the MSE value is lower than U.S. \$50 for nc $\geq$ 92 and npc $\geq$ 64. Likewise, for human voice recording, MSE value is lower than $0.2 \times 10^{-03}$ dB for nc $\geq$ 56 480 and npc $\geq$ 128.
6) Using the proposed method, it is possible to train an SNN for forecasting. The results achieved are worse than those achieved with the MLP network. However, there is room for improvement developing new training algorithms to extract all the potential from the temporal characteristics of SNNs.
7) The encoded data can be compressed by storing the position of the spike within the carrier pulse, instead of storing the whole sequence of bits, which allows the user to specify the exact amount of compression (and, consequently, the quality). Moreover, it depends on one parameter, the selected resolution.
8) The proposed method can be used for analog stream data encoding as a preprocessing phase for on-line learning and signal value prediction with SNN on various streaming data in such applications as: fMRI data [61], [62]; EEG data [63]–[66]; environmental data for personalized modeling and individual stroke and cardio event prediction [67]; remote sensing data for horticulture and agriculture [68], [69]; radio-astronomy; and brain–computer and brain-to-brain telecommunication systems.

## Acknowledgment

The authors would like to thank the reviewers, the Associate Editor, and the Editor-in-Chief for their detailed, precise, and constructive comments and suggestions. They would also like to thank Erasmus Mundus Action 2 PANTHER, "Pacific Atlantic Network for Technical High Education and Research" (references: PA/TG1/AUT/ST/08/2015 and PA/TG1/AUT/PhD/02/2015), for the support.

## References

[1] K. Mehrotra, C. K. Mohan, and S. Ranka, *Elements of Artificial Neural Networks*. Cambridge, MA, USA: MIT Press, 1997.

[2] A. Grüning and S. M. Bohte, "Spiking neural networks: Principles and challenges," presented at the 22nd Eur. Symp. Artif. Neural Netw., Comput. Intell. Mach. Learn. (ESANN), Bruges, Belgium, Apr. 2014.

[3] W. Gerstner and W. M. Kistler, "Elements of neuronal systems," in *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, 1st ed. Cambridge, U.K.: Cambridge Univ. Press, 2002, pp. 11–14.

[4] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003, doi: 10.1109/TNN.2003.820440.

[5] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *J. Physiol.*, vol. 117, no. 4, pp. 500–544, 1952.

[6] J. H. Shin *et al.*, "Recognition of partially occluded and rotated images with a network of spiking neurons," *IEEE Trans. Neural Netw.*, vol. 21, no. 11, pp. 1697–1709, Nov. 2010, doi: 10.1109/TNN.2010.2050600.

[7] L. Perrinet and M. Samuelides, "Sparse image coding using an asynchronous spiking neural network," in *Proc. ESANN*, Apr. 2002, pp. 313–318.

[8] R. Guyonneau, R. VanRullen, and S. J. Thorpe, "Temporal codes and sparse representations: A key to understanding rapid processing in the visual system," *J. Physiol.-Paris*, vol. 98, nos. 4–6, pp. 487–497, 2004, doi: 10.1016/j.jphysparis.2005.09.004.

[9] L. A. Finelli, S. Haney, M. Bazhenov, M. Stopfer, and T. J. Sejnowski, "Synaptic learning rules and sparse coding in a model sensory system," *PLoS Comput. Biol.*, vol. 4, no. 4, Apr. 2008, Art. no. e1000062.

[10] S. Ghosh-Dastidar and H. Adeli, "A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection," *Neural Netw.*, vol. 22, no. 10, pp. 1419–1431, 2009, doi: 10.1016/j.neunet.2009.04.003.

[11] R. Gütig and H. Sompolinsky, "Time-warp–invariant neuronal processing," *PLoS Biol.*, vol. 7, no. 7, Jul. 2009, Art. no. e1000141, doi: 10.1371/journal.pbio.1000141.

[12] N. Kasabov *et al.*, "Evolving spatio-temporal data machines based on the NeuCube neuromorphic framework: Design methodology and selected applications," *Neural Netw.*, vol. 78, pp. 1–14, Jun. 2016, doi: 10.1016/j.neunet.2015.09.011.

[13] D. Reid, A. J. Hussain, and H. Tawfik, "Financial time series prediction using spiking neural networks," *PLoS ONE*, vol. 9, no. 8, 2014, Art. no. e103656, doi: 10.1371/journal.pone.0103656.

[14] D. Reid, H. Tawfik, A. J. Hussain, and H. Al-Askar, "Forecasting weather signals using a polychronous spiking neural network," presented at the 11th Int. Conf. Intell. Comput. Theories Methodologies (ICIC), Fuzhou, China, Aug. 2015.

[15] E. M. Izhikevich, "Polychronization: Computation with spikes," *Neural Comput.*, vol. 18, no. 2, pp. 245–282, 2006, doi: 10.1162/089976606775093882.

[16] S. Panzeri, N. Brunel, N. K. Logothetis, and C. Kayser, "Sensory neural codes using multiplexed temporal scales," *Trends Neurosci.*, vol. 33, no. 3, pp. 111–120, 2010, doi: 10.1016/j.tins.2009.12.001.

[17] Z. Wang, L. Guo, and M. Adjouadi, "Wavelet decomposition and phase encoding of temporal signals using spiking neurons," *Neurocomputing*, vol. 173, no. 3, pp. 1203–1210, Jan. 2016, doi: 10.1016/j.neucom.2015.08.078.

[18] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers Comput. Neurosci.*, vol. 9, p. 99, Aug. 2015.

[19] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, "Real-time classification and sensor fusion with a spiking deep belief network," *Frontiers Neurosci.*, vol. 7, p. 178, Oct. 2013, doi: 10.3389/fnins.2013.00178.

[20] M. Hough, H. De Garis, M. Korkin, F. Gers, and N. E. Nawa, "SPIKER: Analog waveform to digital spiketrain conversion in ATR's artificial brain (CAM-brain) project," in *Proc. Int. Conf. Robot. Artif. Life*, 2009, pp. 1–4.

[21] B. Schrauwen and J. Van Campenhout, "BSA, a fast and accurate spike train encoding scheme," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Portland, OR, USA, Jul. 2003, pp. 2825–2830.

[22] J. Lazzaro and J. Wawrzynek, "A multi-sender asynchronous extension to the AER protocol," in *Proc. ARVLSI*, W. J. Dally, J. W. Poulton, and A. T. Ishii, Eds. Chapel Hill, NC, USA: IEEE Computer Society Press, 1995, pp. 158–169.

[23] E. Tu, N. Kasabov, and J. Yang, "Mapping temporal variables into the NeuCube for improved pattern recognition, predictive modeling, and understanding of stream data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 6, pp. 1305–1317, Jun. 2016, doi: 10.1109/TNNLS.2016.2536742.

[24] V. L.-D. Santos, S. Panzeri, C. Kayser, M. E. Diamond, and R. Q. Quiroga, "Extracting information in spike time patterns with wavelets and information theory," *J. Neurophysiol.*, vol. 113, no. 3, pp. 1015–1033, Feb. 2015, doi: 10.1152/jn.00380.2014.

[25] S. M. Bohte, "The evidence for neural information processing with precise spike-times: A survey," *Natural Comput.*, vol. 3, no. 2, pp. 195–206, 2004, doi: 10.1023/B:NACO.0000027755.02868.60.

[26] M. Jamali, M. J. Chacron, and K. E. Cullen, "Self-motion evokes precise spike timing in the primate vestibular system," *Nature Commun.*, vol. 7, Oct. 2016, Art. no. 13229, doi: 10.1038/ncomms13229.

[27] M. A. Montemurro *et al.*, "Role of precise spike timing in coding of dynamic vibrissa stimuli in somatosensory thalamus," *J. Neurophysiol.*, vol. 98, no. 4, pp. 1871–1882, Oct. 2007, doi: 10.1152/jn.00593.2007.

[28] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, 1997, doi: 10.1016/S0893-6080(97)00011-7.

[29] Z. Nadasdy, "Information encoding and reconstruction from the phase of action potentials," *Frontiers Syst. Neurosci.*, vol. 3, p. 6, Jul. 2009, doi: 10.3389/neuro.06.006.2009.

[30] F. Walter, F. Röhrbein, and A. Knoll, "Computation by time," *Neural Process. Lett.*, vol. 44, no. 1, pp. 103–124, Aug. 2016, doi: 10.1007/s11063-015-9478-6.

[31] A. Cattani, G. T. Einevoll, and S. Panzeri, "Phase-of-firing code," Apr. 2015, *arXiv:1504.03954*. [Online]. Available: https://arxiv.org/abs/1504.03954

[32] M. N. Havenith, S. Yu, J. Biederlack, N.-H. Chen, W. Singer, and D. Nikolić, "Synchrony makes neurons fire in sequence, and stimulus properties determine who is ahead," *J. Neurosci.*, vol. 31, no. 23, pp. 8570–8584, Jun. 2011, doi: 10.1523/JNEUROSCI.2817-10.2011.

[33] Q. Yu, R. Yan, H. Tang, K. C. Tan, and H. Li, "A spiking neural network system for robust sequence recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 621–635, Mar. 2016, doi: 10.1109/TNNLS.2015.2416771.

[34] T. Rumbell, S. L. Denham, and T. Wennekers, "A spiking self-organizing map combining STDP, oscillations, and continuous learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 894–907, May 2014, doi: 10.1109/TNNLS.2013.2283140.

[35] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker project," *Proc. IEEE*, vol. 102, no. 5, pp. 652–665, May 2014, doi: 10.1109/JPROC.2014.2304638.

[36] P. A. Merolla *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, Aug. 2014, doi: 10.1126/science.1254642.

[37] X. Meng, Z. Meng, T. Chen, D. V. Dimarogonas, and K. H. Johansson, "Pulse width modulation for multi-agent systems," *Automatica*, vol. 70, pp. 173–178, Aug. 2016, doi: 10.1016/j.automatica.2016.03.012.

[38] J. Sun, "Pulse-width modulation," in *Dynamics and Control of Switched Electronic Systems. Advanced Perspectives for Modeling, Simulation and Control of Power Converters*. London, U.K.: Springer, 2012.

[39] C. E. Shannon, "Communication in the presence of noise," *Proc. IRE*, vol. 37, no. 1, pp. 10–21, Jan. 1949.

[40] J. Huang, K. Padmanabhan, and O. M. Collins, "The sampling theorem with constant amplitude variable width pulses," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 6, pp. 1178–1190, Jun. 2011, doi: 10.1109/TCSI.2010.2094350.

[41] J. Kominek and A. W. Black. (2003). CMU ARTIC Databases for Speech Synthesis. Carnegie Mellon Universitym. [Online]. Available: http://festvox.org/cmu_arctic/

[42] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997, doi: 10.1023/A:1008202821328.

[43] R. G. Andrzejak, K. Lehnertz, C. Rieke, F. Mormann, P. David, and C. E. Elger, "Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 64, Nov. 2001, Art. no. 061907.

[44] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers Neurosci.*, vol. 10, p. 508, Nov. 2016, doi: 10.3389/fnins.2016.00508.

[45] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Killarney, Ireland, Jul. 2015, pp. 1–8, doi: 10.1109/IJCNN.2015.7280696.

[46] B. Hunsberger and C. Eliasmith, "Training spiking deep networks for neuromorphic hardware," Nov. 2016, *arXiv:1611.05141*. [Online]. Available: https://arxiv.org/abs/1611.05141

[47] S. K. Esser *et al.*, "Convolutional networks for fast, energy-efficient neuromorphic computing," *Proc. Nat. Acad. Sci. USA*, vol. 113, no. 41, pp. 11441–11446, 2016.

[48] S. Schmitt *et al.*, "Neuromorphic hardware in the loop: Training a deep spiking network on the BrainScaleS wafer-scale system," Mar. 2017, *arXiv:1703.01909*. [Online]. Available: https://arxiv.org/abs/1703.01909

[49] S. M. Bohte, J. N. Kok, and H. L. Poutre, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, nos. 1–4, pp. 17–37, 2002.

[50] S. B. Shrestha and Q. Song, "Robust learning in SpikeProp," *Neural Netw.*, vol. 86, pp. 54–68, Nov. 2016, doi: 10.1016/j.neunet.2016.10.011.

[51] S. Yin *et al.*, "Algorithm and hardware design of discrete-time spiking neural networks based on back propagation with binary activations," Sep. 2017, *arXiv:1709.06206*. [Online]. Available: https://arxiv.org/abs/1709.06206

[52] P. Lindstrom, "Fixed-rate compressed floating-point arrays," *IEEE Trans. Vis. Comput. Graphics*, vol. 20, no. 12, pp. 2674–2683, Dec. 2014, doi: 10.1109/TVCG.2014.2346458.

[53] H. Al-Bahadili, "A novel lossless data compression scheme based on the error correcting Hamming codes," *Comput. Math. Appl.*, vol. 56, no. 1, pp. 143–150, Jul. 2008, doi: 10.1016/j.camwa.2007.11.043.

[54] M. Burtscher and P. Ratanaworabhan, "High throughput compression of double-precision floating-point data," in *Proc. Data Compress. Conf. (DCC)*, Mar. 2007, pp. 293–302.

[55] P. Ratanaworabhan, J. Ke, and M. Burtscher, "Fast lossless compression of scientific floating-point data," in *Proc. Data Compress. Conf. (DCC)*, Mar. 2006, pp. 133–142, doi: 10.1109/DCC.2006.35.

[56] V. Engelson, D. Fritzson, and P. Fritzson, "Lossless compression of high-volume numerical data from simulations," in *Proc. Data Compress. Conf. (DCC)*, Mar. 2000, p. 574, doi: 10.1109/DCC.2000.838221.

[57] P. Lindstrom and M. Isenburg, "Fast and efficient compression of floating-point data," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 5, pp. 1245–1250, Sep./Oct. 2006, doi: 10.1109/TVCG.2006.143.

[58] M. Isenburg, P. Lindstrom, and J. Snoeyink, "Lossless compression of predicted floating-point geometry," *Comput.-Aided Des.*, vol. 37, no. 8, pp. 869–877, Jul. 2005.

[59] J. E. Fowler and R. Yagel, "Lossless compression of volume data," in *Proc. Symp. Volume Vis.*, Oct. 1994, pp. 43–50, doi: 10.1145/197938.197961.

[60] *IEEE Standard for Binary Floating-Point Arithmetic*, ANSI/IEEE Standard 754, American National Standards Institute, Washington, DC, USA, 1985.

[61] N. K. Kasabov, M. G. Doborjeh, and Z. G. Doborjeh, "Mapping, learning, visualization, classification, and understanding of fMRI data in the NeuCube evolving spatiotemporal data machine of spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 4, pp. 887–899, Apr. 2017, doi: 10.1109/TNNLS.2016.2612890.

[62] N. Kasabov, L. Zhou, M. G. Doborjeh, Z. G. Doborjeh, and J. Yang, "New algorithms for encoding, learning and classification of fMRI data in a spiking neural network architecture: A case on modeling and understanding of dynamic cognitive processes," *IEEE Trans. Cogn. Develop. Syst.*, vol. 9, no. 4, pp. 293–303, Dec. 2017, doi: 10.1109/TCDS.2016.2636291.

[63] N. Kasabov and E. Capecci, "Spiking neural network methodology for modelling, classification and understanding of EEG spatio-temporal data measuring cognitive processes," *Inf. Sci.*, vol. 294, pp. 565–575, Feb. 2015, doi: 10.1016/j.ins.2014.06.028.

[64] M. G. Doborjeh and N. Kasabov, *Dynamic 3D Clustering of Spatio-Temporal EEG Data Streams in the NeuCube Spiking Neural Network Architecture*. Englewood, CO, USA: Evolving Systems, 2017.

[65] M. G. Doborjeh, G. Wang, N. Kasabov, R. Kydd, and B. Russel, "A spiking neural network methodology and system for learning and comparative analysis of EEG data from healthy versus addiction treated versus addiction not treated subjects," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 9, pp. 1830–1841, Sep. 2016.

[66] E. Capecci, G. Y. Wang, and N. Kasabov, "Analysis of connectivity in NeuCube spiking neural network models trained on EEG data for the understanding of functional changes in the brain," *Neural Netw.*, vol. 68, pp. 62–77, Aug. 2015.

[67] N. Kasabov *et al.*, "Evolving spiking neural networks for personalised modelling, classification and prediction of spatio-temporal patterns with a case study on stroke," *Neurocomputing*, vol. 134, pp. 269–279, Jun. 2014.

[68] T. Gao and N. Kasabov, "Adaptive cow movement detection using evolving spiking neural network models," *Evolving Syst.*, vol. 7, no. 4, pp. 277–285, Dec. 2016.

[69] P. Bose, N. K. Kasabov, L. Bruzzone, and R. N. Hartono, "Spiking neural networks for crop yield estimation based on spatiotemporal analysis of image time series," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 11, pp. 6563–6573, Nov. 2016.


**Ander Arriandiaga** received the M.Sc. degree in control engineering, automatics and robotics from the University of the Basque Country, Bilbao, Spain, in 2011, and the Ph.D. degree in engineering from the Faculty of Engineering, University of the Basque Country in 2016.

He is currently a Research Fellow with the University of the Basque Country. His current research interests include machine learning, deep learning, and neuromorphic engineering.

**Eva Portillo** received the Ph.D. degree in engineering from the Faculty of Engineering, University of the Basque Country, Bilbao, Spain, in 2007.

She is currently an Associate Professor with the Department of Automatic Control and Systems Engineering, Higher Technical Engineering School, University of the Basque Country, where she is also the Vice-Director of the Master and Doctoral School. Her current research interests include the application of intelligent techniques and real-time analysis to manufacturing processes such as grinding, drilling, electrical discharge machining, and robotics and bioengineering.

Dr. Portillo received a grant funded by the Basque Government and a prize for outstanding doctoral thesis in 2010.

**Josafath Israel Espinosa-Ramos** received the M.Sc. degree in cybernetics from La Salle University, Mexico City, Mexico, in 2012, and the Ph.D. degree in computer sciences from the Computer Research Centre, National Polytechnic Institute, Mexico City, in 2016.

He is currently a Research Fellow with the Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, Auckland, New Zealand. His current research interests include computational neuroscience, evolutionary algorithms, and machine learning.

**Nikola K. Kasabov** (F'10) received the M.Sc. degree in electrical engineering, specially computer science, and the Ph.D. degree in mathematical sciences from Technical University, Sofia, Bulgaria, in 1971 and 1975, respectively.

He is currently the Director of the Knowledge Engineering and Discovery Research Institute and the Personal Chair of knowledge engineering with the Auckland University of Technology, Auckland, New Zealand. He has authored over 550 works in the areas of intelligent systems, neural networks, connectionist and hybrid connectionist systems, fuzzy systems, expert systems, bioinformatics, and neuroinformatics.

Dr. Kasabov is a fellow of the Royal Society of New Zealand and a Distinguished Visiting Fellow of the Research Assessment Exercise (RAE), U.K. He was a Past President of the International Neural Network Society (INNS) and the Asia–Pacific Neural Network Assembly (APNNA). He is currently a member of the INNS and APNNA Governing Boards.