Reconstruction, Control and Filtering Methodologies for Spiking Signals

The University of Sheffield

A thesis submitted to the University of Sheffield for the degree of Doctor of Philosophy

Yannan Huang

4th year of submission

Reconstruction, Control and Filtering Methodologies for Spiking Signals

A thesis submitted to the University of Sheffield for the degree of Doctor of Philosophy

Yannan Huang

Department of Automatic Control and Systems Engineering

September 2023

Acknowledgements

I would like to thank my first supervisor Prof. Daniel Coca for his advices during my PhD period. He gave many professional and helpful suggestions to my research. I would also like to thank my second supervisor Dr. Guliano Punzo for helping me deal with the problems occurred these years. Thanks to Dr. Ivan Minev, who gave me many useful suggestions on time management.

Many thanks to my family for their companion during my study and research.

Acronyms

IF	integrate-and-fire.
IIF	ideal integrate-and-fire.
LIF	leaky integrate-and-fire.
TEM	time encoding machine.
TDM	time decoding machine.
SI&G	Spike Integrate & Generate.
STD	Spike Temporal Derivative.
SH&F	Spike Hold & Fire.
ZOH	Zero-Order hold.
FOH	Firstly-Order hold.
C-TEM	crossing TEM.
EIEO	encoded-input-encoded-output.
SNN	spiking neural network.
LTI	linear time invariant

Abstract

Integrate-and-fire Time Encoding Machines (IF-TEMs) map an input signal onto a strictly increasing sequence of time instants. Time encoding and decoding theory holds promise for reducing the amount of data generated and transmitted and the energy costs in data transmission, improved robustness to noise and low communication latency as well as providing a basis for implementing novel signal processing, communication and control approaches that use time encoded signals. However, there are a number of challenges to address including: real time decoding of spike-time sequences, development of novel control architectures that operate on spike-trains and methodologies to map conventional signal processing operations. This thesis addresses three major problems in the field of real-time decoding of spikes and spiking signal operations: 1. Designing of real-time decoding algorithms for time encoded signals; 2. Designing of closed-loop spike-based control systems; 3. Designing of spiking filters.

This thesis introduces real-time decoding algorithms for signals encoded using IF-TEM. The decoding algorithm provides a piecewise-constant approximation of the original signal over the spike time intervals. Analytical bounds are derived for the approximation error at the mid-point of the spike-time interval. The analytical error bound show explicitly the dependence of the error on the key parameters of IF-TEM i.e., the neuron threshold and the additive bias.

The fast decoding algorithm, facilitates the utilization of time encoding within a real-time control framework. To that end, this thesis introduces a spike-based network control system, a key component of which is a spike-based PID control law. The spike-based PID controller proposed in this thesis responds to spiking signals and the closed-loop system is able to track reference signals with high responding speed and accuracy. The output of a spike-based control system is significantly close to that of a conventional continuous control system.

In an ideal scenario, time encoding should replace conventional amplitude-sampling used not just in control but for general signal processing. To that end, this thesis introduces a system identification approach for developing Encoded-Input-Encoded-Output (EIEO) models of dynamical systems, which can be used to implement basic filtering operations by directly processing spike time sequences. This methodology enables the creation of spike-based linear and nonlinear operators as well as filters.

Contents

Chapter 1

Introduction

1.1. Background

In the field of systems and control engineering, signal processing underpins all modern technologies [1]. However, in some practical application scenarios, reducing energy consumption, the amount of data transmitted and increasing computational efficiency and robustness to noise are more important than ever given the huge increase in the volume of sensor data that is being generated and processed [2].

Time Encoding Machines (TEMs) are computational models inspired by the behaviour of neurons in the brain. They are designed to efficiently encode and process continuous data using spike trains. These spike trains represent information in the form of precisely timed events, akin to the way neurons communicate in biological systems. The block diagram of a TEM is shown in Fig. 1.1.

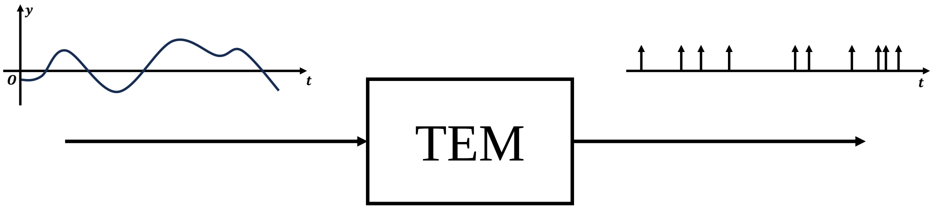


Fig. 1.1. Block diagram of a TEM. The TEM maps a continuous signal into a spike train.

Time Decoding Machines (TDMs) reconstruct the continuous original signal from the encoded spikes and recover the information carried by this signal.

Time encoding and decoding processes enable the data transmission in the format of time sequencies, such as spikes [3]. The information of the original signal is carried using the arriving time of the spikes. Therefore, through the time encoding process, the information is transmitted using spikes. The information can also be recovered by employing some time decoding methodologies under the assumption that the average sampling frequency is larger than the Nyquist frequency.

Time encoding, a technique used to represent and process information through spike trains, offers several advantages in this regard:

- Energy Efficiency: Spike-based transmission of control and feedback systems is highly energy-efficient. Instead of continuously transmitting signals, information is transmitted only when there is a relevant event or change in the data. This leads to significant energy savings, making this approach suitable for low-power applications [4].

- Simpler devices used for receiving and sending spiking signals [4]: For spiking data transmission, these only need to respond to the changing of potential and send binary signals.
- Sparse Data Representation: Time encoding of signals represents data in a sparse manner. Most of the time, no spikes are transmitted, and only relevant spikes are sent. This sparse data representation simplifies data processing and reduce the amount of data that needs to be transmitted or stored.
- Robustness to noise: The spiking signals are robust to noise Spike-based communication systems are inherently robust to noise and interference [5]. The information is encoded in the timing and pattern of spikes, and this temporal coding can withstand various types of noise without significant degradation in performance. [5].
- Event-Driven Processing: Spike-based communication promotes event-driven processing, where computations occur only when necessary. This leads to efficient utilization of computational resources and can reduce power consumption in electronic systems.

Considering the advantages of spiking data transmission, one might intend to employ time encoding and decoding methodologies in a real-time control system. In order to establish a spike-based control system, some spiking signal operations can be applied [6~8]. Alternatively, one can employ time encoding methodologies to transmit the spiking data, and decode the spiking data to recover the information so that the system can respond to the decoded signal. Through the application of spiking data transmission, a spike-based control system can be established.

Besides spike-based control system, one can also design spike-based components using spiking signals. One can construct components that respond to spikes and have also spiking outputs. By designing spike-based components, one can significantly reduce the cost of the construction of a spike-based system.

1.2. Motivation

This thesis addresses several problems in the field of time decoding and spike data transmission as well as spike-based systems.

While time encoding and decoding theory holds promise for reducing energy costs in data transmission, there are still challenges to address. The primary issues include achieving flawless time decoding, ensuring the recovery of the original signal without any loss of information when dealing with discrete signals. Additionally, there is a need to enable real-time decoding to achieve real-time control of a dynamical system.

A proposed solution by Lazar [3] involves the utilization of integral-and-fire neurons for time encoding and information-loss-free reconstruction. Lazar's algorithm is capable of accurately recovering the original signal without any loss of information. However, it is important to note that this algorithm does not currently support real-time time encoding and decoding, presenting a significant area for improvement.

Therefore, this thesis proposes a fast decoding algorithm that can recover the original signal in real-time and has bounded reconstruction error. Through this algorithm, it no longer requires the entire spike train to be fully transmitted before decoding. The waiting times of decoding are restricted by the spike intervals, which is much smaller compared with the entire spike train. The fast decoding algorithm is based on the estimation of the original signal at spike mid-points. The estimation error at the mid-points is proven to be bounded and smaller than Lazar's reconstruction algorithm. One can employ a Zero-Order hold or other signal extrapolation methodologies to reconstruct a continuous signal from the discrete mid-point estimation.

This thesis also presents the methodology of constructing a real-time closed-loop control system which is based on spiking data transmission. Spike-based control system has been discussed by some papers [6~8]. Most of the papers focused on some simple operations of spikes. This thesis emphasises the operation of data based on time encoding and decoding. A spike-based PID controller is proposed in this thesis in order to establish a spike-based control law that can be employed by a system. The proposed spike-based PID controller outputs control signal by responding to the encoded input spikes. All the proportional, integral and derivative terms of the PID controller are calculated by the controller using the encoded spiking input. Through the designing of the spike-based PID controller, one is able to establish a remote control system that uses spiking signals to transmit data between different components.

Spike-based components have been developed in paper [6]. The paper focused mainly on spike operations such as Spike Integrate & Generate (SI&G), Spike Temporal Derivative (STD) and Spike Hold & Fire (SH&F). These are simple components that can only execute simple spike operations. This thesis proposes a designing methodology for spike-based components such as spiking filters and linear/nonlinear spiking systems. This designing methodology is based on System Identification. Through the identification of encoded input and output signals, one can design components that respond to spikes and output spiking signals.

1.3. Overview of the thesis

Chapter 2 reviews approaches of time encoding and decoding. Crossing time encoding machines are reviewed in Chapter 2 and relative decoding algorithms are presented. Chapter 2 focuses on the theory of integrate-and-fire time encoding machines (IF-TEMs). The encoding algorithms of different IF-TEMs are presented, including ideal IF-TEMs, IF-TEMs with refractory periods and Leaky IF-TEMs. Different reconstruction algorithms are reviewed in Chapter 2. The chapter also reviews a stitching reconstruction algorithm which intends to realize a real-time recovery of signal. The encoding and decoding algorithms for multi-channel IF-TEMs are also reviewed in Chapter 2.

Chapter 3 presents a novel fast real-time reconstruction algorithm to decode the signal sampled by integrate-and-fire TEMs. This algorithm estimates the original input signal at the midpoint of two spikes. The methodology is to calculate the derivative of the integral of an IF-TEM or the postsynaptic potential of an IF neuron, and use this derivative to estimate the original signal. This algorithm is able to make the estimation using only two consecutive spikes. Through numerical study and some simulation examples, it is illustrated that the estimation error is bounded and can be restricted by adjusting TEM parameters. Notice that the estimation of the original signal is discrete and therefore a filter is applied in order to reach a continuous reconstruction.

Chapter 4 presents the designing of a spike-based PID control system. The system is designed to realize the remote control that transmit data using spiking signals. In order to employ spiking data transmission, a spike-based PID controller is required. Therefore, Chapter 4 proposes the designing of a spike-based PID controller. The spike-based controller calculates proportional, integral and derivative terms using the spiking input, which is encoded from a continuous original signal by an IF-TEM. The calculation is proposed in the chapter and it has been proved that the calculation error is bounded. Through numerical study, it is shown that the spike-based PID control system has very similar performance with a conventional continuous-time control system, which indicates that this spike-based control system can model a conventional system, and one can construct a spike-based PID control system using the proposed methodology. It has also been shown that the spike-based PID controller can use the same PID gains as a standard PID controller. One can also retune the spike-based controller slightly in order to have a better tracking performance.

Chapter 5 proposes a designing methodology of spike-based components. Besides PID controllers, it also requires other types of components in practical industrial applications. If one intends to construct a spike-based control system, an approach is applying TEMs and TDMs to each component in the system, which is significantly costly. Therefore, Chapter 5 proposes a designing methodology of spike-based components based on System Identification. The chapter illustrates the filtering of time-encoded signals. The encoded-input-encoded-output model is presented in Chapter 5. The chapter also summarizes different types of System Identification approaches. Numerical study has been carried out in Chapter 5 to present the designing of spike-based linear/nonlinear models. The chapter addresses the causality problem introduced by the encoded-input-encoded-output model, and proposes a solution to this problem.

Chapter 6 presents the conclusion and future work.

Chapter 2

Time encoding and decoding approaches

2.1. Introduction

Sampling is a methodology designed to transmit data with minimized data loss using limited resources [9]. Conventional uniform sampling where amplitude samples are measured or recorded at fixed and evenly spaced time intervals controlled by a clock that are consume power, especially at high sampling rates and are subject to electromagnetic interference [10]. As an alternative, Time Encoding Machines (TEM) offer an alternative representation of analogue signals [11]. Unlike conventional analogue-to-digital converters, TEMs operate asynchronously, eliminating the need for a global clock. This results in lower power consumption and reduced electromagnetic interference [4]. In this sampling approach, an analogue signal is represented by a collection of time instants at which the input signal or its function crosses a specific threshold.

Time encoding is a real-time sampling of a signal, where the amplitude information is mapped into a time sequence [3]. Time encoding and decoding methodologies are reviewed in this chapter. This chapter focuses on time encoding and decoding based on integrate and fire neurons [3], which involves mapping the original signal $u(t)$ into a spike train $\{t_k\}$, where the information from the signal is carried by spike arrival times t_k. As an inverse process, time decoding reconstructs the original signal $u(t)$ using the encoded spike train $\{t_k\}$. In summary, spiking time encoding is a process of representing the signal using spikes, while time decoding aims to recover the original continuous signal from the spike train.

This chapter is organized as follows: Section 2.2 reviews existing time encoding methodologies with a focus on Crossing TEMs (C-TEMs). The reconstruction algorithm for C-TEMs is also reviewed. Section 2.3 reviews the literatures of integrate-and-fire TEMs. Different types of IF-TEMs are illustrated in this section. Section 2.4 reviews perfect reconstruction algorithms for IF-TEMs. Section 2.5 presents multi-channel time encoding and decoding algorithms. Section 2.6 draws the conclusion and discusses the limitation of existing reconstruction algorithms.

2.2. Time encoding methodologies

Crossing TEM [12] as well as Integrate-and-fire TEM [3] are two main types of time encoding machines.

A C-TEM uses a continuous test function $\Phi(t)$ and a comparator to sample the input signal $f(t)$. A

C-TEM generates an increasing tome sequence (spike times) indicating the times the input signal crosses the test function, $\{t_n : f(t_n) = \Phi(t_n)\}$. C-TEMs can be constructed either without feedback or with feedback.

When implementing a C-TEM, the test function $\Phi(t)$ can be defined as an independent periodic function (Fig. 2.1).

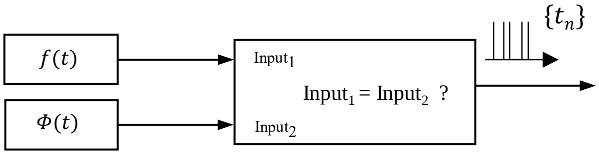
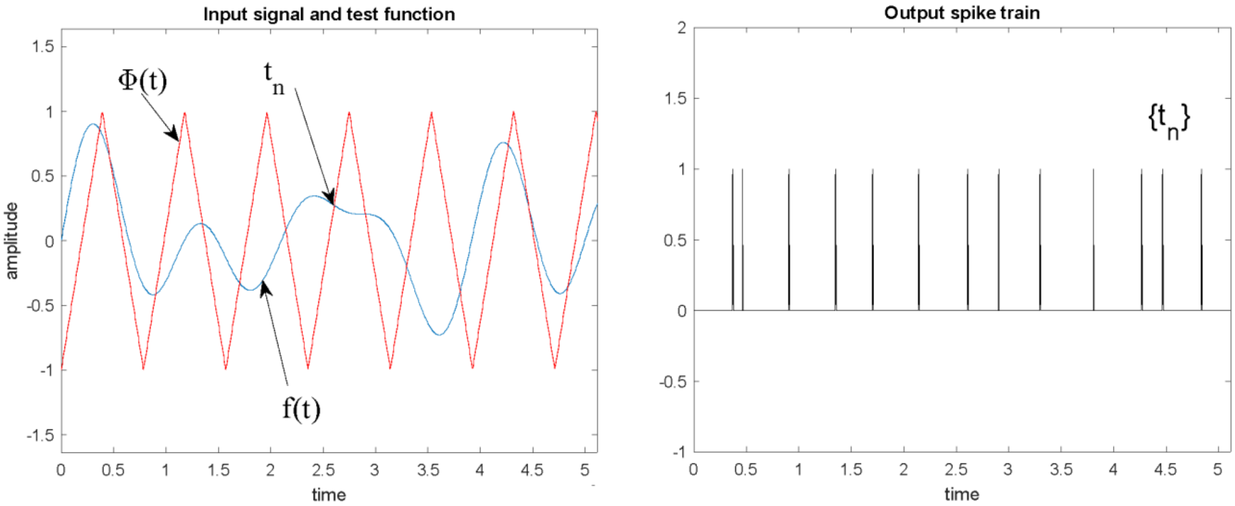


Fig. 2.1. Block diagram of a C-TEM. $\Phi(t)$ is independent to input and output. Therefore, this C-TEM is constructed without feedback.



a) Sampling the input signal $f(t)$ using independent test function $\Phi(t)$.

b) Output spike train

Fig. 2.2. Time encoding methodology of a C-TEM without feedback and the encoded signal.

Fig. 2.2.a) and Fig. 2.2.b) illustrate the operation of a C-TEM without feedback using a triangular wave function $\Phi(t)$ and the resulting output spike train.

A C-TEM can also be constructed with a feedback signal. A feedback loop is introduced so that the test function $\Phi(t)$ is updated when the TEM outputs a spike. The test function $\Phi_n(t)$ is defined as:

$$\Phi_n(t) = \begin{cases} k_\Phi(t - M \cdot T) + b, & M \cdot T \le t < \left(M + \frac{1}{2}\right) \cdot T \\ -k_\Phi\left(t - \left(M + \frac{1}{2}\right) \cdot T\right) + b, & \left(M + \frac{1}{2}\right) \cdot T \le t < (M+1) \cdot T \end{cases}, \quad M = 0,1,2, \ldots.$$

(2.1)

Where k_{Φ} is the slope of $\Phi_n(t)$, b is the lower limit of $\Phi_n(t)$, T is the period of the test function.

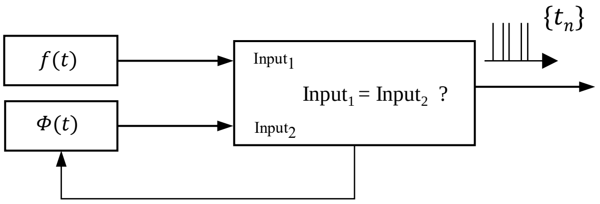
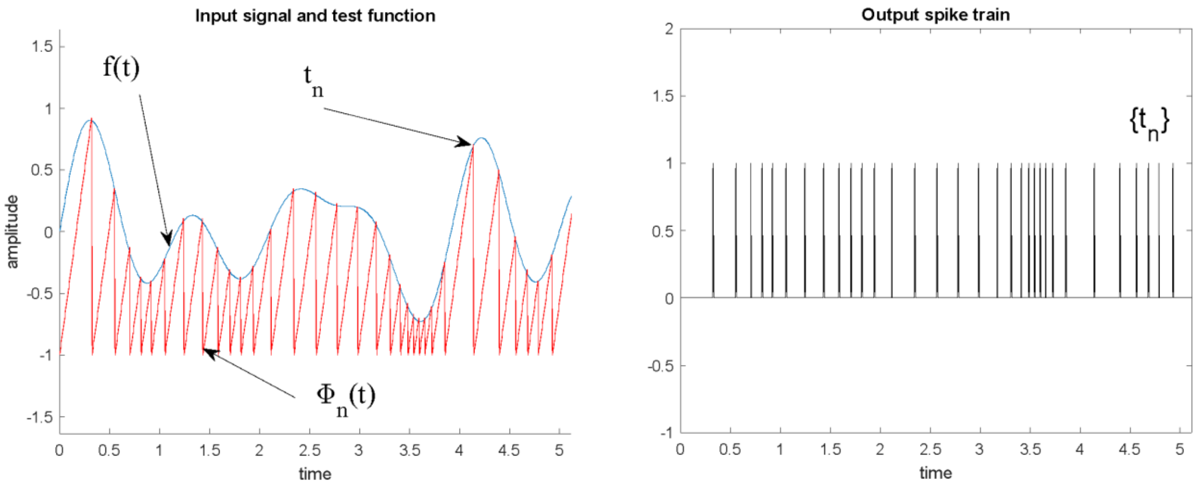


Fig. 2.3. Block diagram of a C-TEM. $\Phi(t)$ is updated when spikes are output. Therefore, this C-TEM is constructed with feedback.

As shown in Fig. 2.3, $\Phi_n(t)$ is updated when the C-TEM outputs a spike. Therefore, $\Phi_n(t)$ is refreshed every time the amplitude of $f(t)$ exceeds $\Phi_n(t)$. The test function $\Phi_n(t)$ is defined as:

$$\Phi_n(t) = \begin{cases} k_{\Phi}t + b, & 0 \leq t \leq t_1 \\ k_{\Phi}(t - t_n) - b, & t_n < t \leq t_{n+1} \end{cases}, \quad n = 1,2,3, \ldots. \tag{2.2}$$

An example showing the sampling method of a C-TEM with feedback is presented in Fig. 2.4.a) and Fig. 2.4.b).



a). Sampling the input signal $f(t)$ using a refreshing test function $\Phi(t)$.

b). Output spike train.

Fig. 2.4. Time encoding methodology of a C-TEM with feedback and the encoded signal.

Fig. 2.4.a) and Fig. 2.4.b) illustrate the sampling process of a C-TEM employing a feedback loop. The difference with a C-TEM without feedback is that the test function $\Phi_n(t)$ is updated during the sampling process.

Compared with the C-TEMs with feedback, C-TEMs without feedback do not require feedback components to encode input signals. However, for C-TEMs with feedback, the reconstruction will not be affected by time shifts between encoders and decoders [12].

The continuous original signal $f(t)$ is mapped into a time sequence $\{t_n\}$ by a C-TEM. Conversely, the original signal $f(t)$ can be reconstructed using the iterative algorithm provided by [12]:

Define that

$$G_\lambda(\omega) = \left(\sum_{k\in\mathbf{Z}}\left|\hat{\lambda}(\omega + 2k\pi)\right|^2\right)^{\frac{1}{2}}, \tag{2.3}$$

where $\hat{\lambda}$ is the classic Fourier transform of λ,

$$H^1(\mathbf{R}) = \{f \in L^2, \|f\|_{H^1} < \infty\} \tag{2.4}$$

$$\text{with } \|f\|_{H^1}^2 = \|f\|_{L^2}^2 + \|f'\|_{L^2}^2 = \frac{1}{2\pi}\int_{-\infty}^{\infty}(1+\omega^2)\left|\hat{f}(\omega)\right|^2 d\omega, \tag{2.5}$$

$$E = \{\lambda \in H^1(\mathbf{R}), 0 < A \leq G_\lambda(\omega) \leq B < \infty \; and \; G_{\lambda'}(\omega) < \infty\}, \tag{2.6}$$

$$V = Vf(t) = \sum_{n=1}^{\infty} f(t_n) \cdot \mathbb{1}_{[s_n, s_{n+1})}(t), \tag{2.7}$$

where $s_n = \frac{t_{n-1}+t_n}{2}$, $\mathbb{1}_{[s_n, s_{n+1})}$ is the Voronoï region of t_n.

Then, let $\lambda \in E$ and

$$\tau = \pi \cdot \lim_{\omega\to\infty} \frac{G_\lambda(\omega)}{G_{\lambda'}(\omega)} > 0, \tag{2.8}$$

for all $T < \tau$, if $t_{n+1} - t_n \leq T \; \forall n$, the C-TEM encoding process is invertible. If $\{t_n\}$ is the resulting encoded time sequency of $f(t)$ using the C-TEM, then the reconstruction of $f(t)$ is given iteratively by:

$$f_1 = PVf_0$$
$$f_{k+1} = f_1 + (Id - PV)f_k, \tag{2.9}$$

where P is the orthogonal projector on $V^2(\lambda)$ and $\|Id - V\| < 1$.

2.3. Time encoding machines

An alternative implementation of a Time Encoding Machine is the integrate-and-fire TEM (IF-TEM) that mimics the operation of biological neurons [3]. A spike is generated by an IF-TEM each time the integration of the input signal reaches (crosses) a specific threshold. In essence the IF-TEM is a crossing TEM with feedback where the test function, which is a constant function, is compared with the integral of the input signal.

The following sections summaries different types of IF-TEMs.

2.3.1. Ideal IF-TEMs

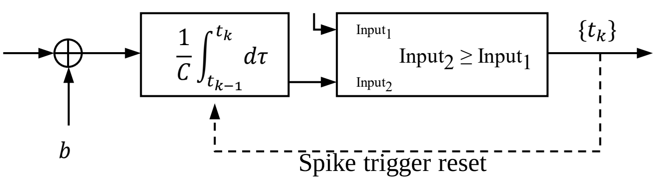


Fig. 2.5. The block diagram of an ideal integrate-and-fire TEM

As shown in Fig. 2.5, $u(t)$ is integrated. When the integral of $u(t)$ reaches the threshold δ, the TEM fires a spike and the integral of $u(t)$ is reset to zero.

An Ideal Integrate-and-Fire TEM (IIF-TEM) is defined as:

For an input u satisfying $|u| \leq c < b$, an IIF-TEM generates a spike train $\{t_n\}_{n\in Z}$ when

$$\int_{t_n}^{t_{n+1}} u(\tau)d\tau = C\delta - b(t_{n+1} - t_n), \forall n \in Z, \tag{2.10}$$

where C is the constant of integrate, b is the bias which is larger than the bound of $u(t)$.

The information of the original input signal $u(t)$ is mapped into the spike arriving times during the time encoding process. It has been proven in [13] that the intervals of spike arriving times are bounded:

For an input u that $|u| \leq c < b$, the upper and lower bound of the spiking time interval $t_{n+1} - t_n$ is given by

$$\frac{C\delta}{b+c} \leq t_{n+1} - t_n \leq \frac{C\delta}{b-c}, \forall n \in Z. \tag{2.11}$$

According to [3], the input $u(t)$ can be perfectly recovered without loss of information if it is bandlimited.

2.3.2. IF-TEMs with absolute refractory period

According to Section 2.3.1, the integral of the input signal $u(t)$ is reset to zero when a spike is generated. If the IF-TEM or the IF neuron is ideal, the reset takes no time. Here it is assumed that the reset of the integral takes a time period Δ which is small. An ideal IF-TEM with refractory period is defined as:

Consider that the original signal u satisfies $|u| \leq c < b$, a spike is generated by an ideal IF-TEM with refractory period when

$$\int_{t_n+\Delta}^{t_{n+1}} u(\tau)d\tau = C\delta - b(t_{n+1} - t_n - \Delta), \forall n \in Z, \tag{2.12}$$

where C is the constant of integrate, b is the bias which is larger than the bound of $u(t)$.

The spiking time intervals are also proven to be bounded as:

For an input u that $|u| \leq c < b$, the upper and lower bound of the spiking time interval $t_{n+1} - t_n$ is given by

$$\frac{C\delta}{b+c} + \Delta \leq t_{n+1} - t_n \leq \frac{C\delta}{b-c} + \Delta, \forall n \in Z. \tag{2.13}$$

2.3.3. Leaky IF-TEMs

A shortcoming of the IF neuron model is that it does not account for current leakage. If the model receives a below-threshold short current pulse at some time, it will retain that voltage boost forever - until another input later makes it fire. Leaky Integrate-and-Fire TEMs (LIF-TEMs) incorporate a term in the membrane potential equation which models the diffusion of ions through the membrane, to reflect the fact that the membrane is not a perfect insulator. The equations describing the LIF model are as follows

For an input u satisfying $|u| \leq c < b$, an LIF-TEM generates a spike train $\{t_n\}_{n\in Z}$ when

$$\int_{t_n}^{t_{n+1}} u(\tau)e^{-\frac{t_{n+1}-\tau}{RC}}d\tau = C(\delta - bR) + C\big(bR - y(t_0)\big)e^{-\frac{t_{n+1}-t_n}{RC}}, \forall n \in Z, \tag{2.14}$$

where $y(t_0)$ is the initial condition of the RC filter, and b is the bias which is larger than the bound of $u(t)$.

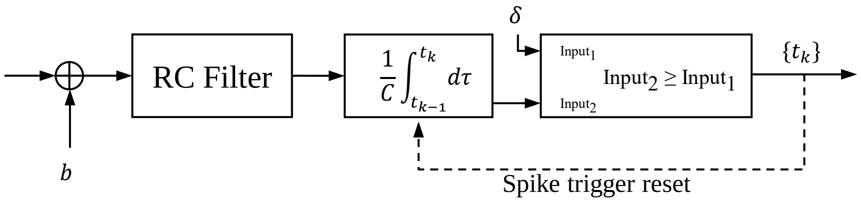


Fig. 2.6. The block diagram of a Leaky IF-TEM

The LIF neuron is represented by an RC circuit with a threshold, which results a slightly different definition as shown in (2.14).

2.4. Time decoding algorithms for IF-TEMs

2.4.1. Perfect recovery for ideal IF-TEMs

The reconstruction algorithm for ideal IF-TEMs is given by Lazar et al [3].

This section summarises the conditions for information-loss-free recovery of the original signal $u(t)$ from the sequence of trigger (spike) times produced by an IF-TEM with parameters.

Firstly, an operator *A* is defined as

$$\begin{aligned} A = Au &= \sum_{k\in Z} \int_{t_k}^{t_{k+1}} u(\tau)\, d\tau\, g(t - s_k) \\ &= \sum_{k\in Z} [C\delta - b(t_{k+1} - t_k)]\, g(t - s_k), \end{aligned} \tag{2.15}$$

where $g(t) = \frac{sin(\Omega t)}{\pi t}$ and $s_k = \frac{t_{k+1}+t_k}{2}$.

Define a sequence of bandlimited functions $u_l = u_l(t)$ as

$$u_{l+1} = u_l + A(u - u_l) \tag{2.16}$$

for all $l, l \in N$, and $u_0 = Au$.

Since $t_{k+1} - t_k$ is bounded by $\frac{C\delta}{b-c}$ according to (2.11), it follows that [14]

$$||I - A|| \leq \frac{2k\delta\Omega}{(b-c)\pi}. \tag{2.17}$$

Theorem 1

If the signal $u(t)$ is encoded by an IIF-TEM, then the signal can be reconstructed by

$$u_l(t) \ = u(t). \tag{2.18}$$

Proof: It holds that

$$u_l = \sum_{k=0}^{l} (I - A)^k Au. \tag{2.19}$$

Since $||I - A|| \leq r < 1$ where $r = \frac{2C\delta\Omega}{\pi(b-c)}$,

$$u_l \ = \sum_{k\in N} (I - A)^k Au = A^{-1}Au = u. \tag{2.20}$$

According to the mean value theorem, there exists a $\xi_k \in [t_k, t_{k+1}]$ such that

$$u(\xi_k)(t_{k+1} - t_k) = C\delta - b(t_{k+1} - t_k). \tag{2.21}$$

Therefore, $u(t)$ is firstly estimated by (2.15). There is an estimation error of $u(t)$ yet the error is reduced during the iteration.

Let $\boldsymbol{g} = [g(t - s_k)]$, $\boldsymbol{q} = [C\delta - b(t_{k+1} - t_k)]$ and $\boldsymbol{G} = \boldsymbol{G}_{lk} = [\int_{t_l}^{t_{l+1}} g(u - s_k)du]$, $l, k \in \boldsymbol{Z}$.

Therefore,

$$u_l(t) = \boldsymbol{g}^T \sum_{k=0}^{l} (\boldsymbol{I} - \boldsymbol{G})^k \boldsymbol{q} \tag{2.22}$$

Under the assumption in Theorem1, the input signal $u(t)$ can be recovered by [13]

$$u(t) = \lim_{l\to\infty} u_l(t) = \lim_{l\to\infty} \boldsymbol{g}^T \sum_{k=0}^{l} (\boldsymbol{I} - \boldsymbol{G})^k \boldsymbol{q} = \boldsymbol{g}^T \boldsymbol{G}^+ \boldsymbol{q}\,, \tag{2.23}$$

where $\boldsymbol{G}^+$ represents the Pseudo-inverse of $\boldsymbol{G}$.

■

2.4.2. Decoding algorithm for IF-TEMs with absolute refractory period

The signal encoded by an IF neuron with refractory period can be perfectly reconstructed [15] if

$$\frac{\delta\Omega}{\pi(1-c)} < \frac{1-\varepsilon}{1+\varepsilon}, \tag{2.24}$$

where $\varepsilon = \sqrt{\Delta/(\frac{C\delta}{b+c} + \Delta)}$.

The reconstruction algorithm for the TEM is given by [15]:

$$u = \boldsymbol{g}\boldsymbol{G}^+\boldsymbol{q}, \tag{2.25}$$

where $\boldsymbol{g} = [g(t - s_n)]^T, \boldsymbol{G} = \left[\int_{t_l+\Delta}^{t_{l+1}} g(\tau - s_n)d\tau\right], \boldsymbol{q} = [\int_{t_l+\Delta}^{t_{l+1}} u(\tau)d\tau]$ and $s_n = \frac{t_{n+1}+t_n}{2}$.

2.4.3. Decoding algorithm for leaky IF-TEMs

The signal encoded by a leaky IF neuron can be perfectly reconstructed if

$$RC \cdot ln\left(1 - \frac{\delta - y(t_0)}{\delta - (b-c)R}\right)\frac{\Omega}{\pi} < \frac{1-\epsilon'}{1+\epsilon'}, \tag{2.26}$$

where $\epsilon' = \frac{\delta - y(t_0)}{(b-c)R - y(t_0)}$.

The reconstruction can be realized by

$$u = \boldsymbol{g}\boldsymbol{G}^+\boldsymbol{q}, \tag{2.27}$$

where $\boldsymbol{g} = [g(t - s_n)]^T$, $\boldsymbol{G} = \left[\int_{t_l+\Delta}^{t_{l+1}} g(\tau - s_n)e^{-\frac{t_{n+1}-\tau}{RC}}d\tau\right]$, and $\boldsymbol{q} = [C(\delta - bR) + C\big(bR - y(t_0)\big)e^{-\frac{t_{n+1}-t_n}{RC}}]$.

2.4.4. An alternative reconstruction framework

An alternative reconstruction framework in which IF time encoding is reformulated as a uniform sampling [16].

Given a bandlimited signal u(t), let $T_u = \{t_k\}_{k\in Z}$ be the spike train encoded by an IF-TEM defined by $\{\delta, b, C\}$. Notice that the threshold δ of the TEM is fixed, the time encoding process is a uniform sampling of the inverse of the function

$$y(t) = \int_0^t (u(\tau) + b)d\tau \tag{2.28}$$

with sampling points $D = \{k\bar{\delta}\}_{k\in Z}$, where $\bar{\delta} = C\delta$ [16].

Since $y(t) = \int_0^t (u(\tau) + b)d\tau$ is a strictly increasing function, it holds that $y(t)$ has an inverse function ψ such that

1. $L_k^D\bar{\psi}' = \bar{q}_k, \forall k \in Z$,

   where $\bar{\psi}(x) \triangleq \psi(x) - \frac{x}{b}$, $D = \{k\bar{\delta}\}_{k\in Z}$ and $\bar{q}_k \triangleq (t_{k+1} - t_k) - \frac{\bar{\delta}}{b}$.

2. $\left|\bar{\psi}'(x)\right| \leq \frac{c}{b(b-c)}$, $\forall x \in R$ and $\|\bar{\psi}'\|_{L^2}^2 \leq \frac{1}{b^2(b-c)}\|u\|_{L^2}^2$,

   where $\|\cdot\|_{L^2}$ denotes the standard norm in $L^2(R)$.

Since ψ is the inverse of y, it follows that $\psi'(y(t)) = \frac{1}{y'(t)}$. Hence,

$$\bar{\psi}'(y(t)) = \frac{1}{y'(t)} - \frac{1}{b} = -\frac{u(t)}{b(u(t)+b)}. \tag{2.29}$$

By replacing t with $\psi(x)$, it follows that

$$u(\psi(x)) = -\frac{b^2\bar{\psi}'(x)}{b\bar{\psi}'(x)+1}. \tag{2.20}$$

In [16] it is shown that $\bar{\psi}'(x)$ can be approximated using a function $\bar{\psi}_M'$ that satisfies

$$supp(\bar{\psi}_M') \subseteq [-\Omega, \Omega],\ \ \Omega < \infty,$$

where $supp(\bar{\psi}_M')$ is the support of $\bar{\psi}_M'$.

Specifically, for any $M \in \mathbf{N}$, it holds that

$$\bar{\psi}' = \bar{\psi}_M' + e_M, \tag{2.21}$$

where

$$essBW_{\bar{\psi}_M'} \leq \frac{M\Omega}{b-c}, \|e_M\|_{L^2}^2 \leq \left(\frac{c}{b}\right)^{2M} \frac{b+c}{b-c}\|\bar{\psi}'\|_{L^2}^2. \tag{2.22}$$

Moreover, $\bar{\psi}' \in PW_{\bar{\Omega}_M}$ and $\bar{\psi}$, where $\bar{\Omega}_M = \frac{M\Omega}{b-c}$ and $\bar{\delta} < \frac{\pi}{\bar{\Omega}_M}$, can be reconstructed with arbitrary precision:

$$\bar{\psi}' = \bar{\boldsymbol{g}}_1^*\bar{\boldsymbol{Q}}^+\bar{\boldsymbol{q}}, \tag{2.23}$$

$$\bar{\psi} = \bar{\boldsymbol{g}}_2^*\bar{\boldsymbol{Q}}^+\bar{\boldsymbol{q}}, \tag{2.24}$$

where $[\bar{\boldsymbol{g}}_1]_m = \bar{g}(\cdot - \bar{s}_{m+1})$, $\bar{g} \triangleq \frac{sin\,(\bar{\Omega}_M\cdot)}{\pi\cdot}$, $\bar{s}_m \triangleq \frac{2m-1}{2}\bar{\delta}$, $[\bar{\boldsymbol{g}}_2]_m = \int_0^x \bar{g}(\tau - \bar{s}_{m+1})d\tau$, $\forall x \in R$, $[\bar{\boldsymbol{Q}}]_{m,k} = L_m^D(\bar{g}(\cdot - \bar{s}_{k+1}))$, and $[\bar{\boldsymbol{q}}]_k = L_k^D\bar{\psi}'$, $\forall k, m \in \boldsymbol{Z}$.

The functions $\bar{\psi}'$ and $\bar{\psi}$ are sampled on the uniform grid $\{k\varepsilon\}$, where ε is the sampling period for reconstruction. Using linear interpolation, the reconstructed signal $\tilde{u}$ is given by

$$\tilde{u}(t) = u(\psi(k\varepsilon)) + \frac{t-\psi(k\varepsilon)}{\psi((k+1)\varepsilon)-\psi(k\varepsilon)} \cdot \left[u\left(\psi((k+1)\varepsilon)\right) - u(\psi(k\varepsilon))\right]. \tag{2.25}$$

By employing this algorithm, the speed of reconstruction is significantly improved.

2.4.5. Lazar’s real-time decoding algorithm

Lazar, Simonyi, and Tóth proposed an algorithm for real-time decoding [17]. The algorithm divides the original input signal into short overlapping time intervals. Then the divided parts of the spike train are utilized to reconstruct the signal. The reconstructions approximate the original signal without loss of information at short time intervals. Then the approximations are stitched together as an overall approximation of the original signal.

Let $\{t_m, t_{m+1}, \dots, t_{m+L}\}$ be one of the short spike-time sequences and let the vector $\boldsymbol{q}_{m,L}$ and matrix $\boldsymbol{G}_{m,L}$ be defined as

$$\left[\boldsymbol{q}_{m,L}\right]_{k-m+1} = C\delta - b(t_{k+1} - t_k) \tag{2.26}$$

$$[\boldsymbol{G}_{m,L}]_{k-m+1,l-m+1} = \int_{t_k}^{t_{k+1}} g(t - s_l)dt, \tag{2.27}$$

for all $k, l = m, m+1, \ldots, m+L-1$.

Then, the original signal $u(t)$ is approximated as

$$u_{m.L}(t) = \sum_{l=m}^{m+L-1} [\boldsymbol{c}_{m,L}]_{l-m+1} g(t - s_l), \tag{2.28}$$

where $\boldsymbol{c}_{m,L}$ is calculated by

$$\boldsymbol{c}_{m,L} = \boldsymbol{G}_{m,L}^{+} \boldsymbol{q}_{m,L}, \tag{2.29}$$

where $\boldsymbol{G}_{m,L}^{+}$ denotes the pseudo-inverse of $\boldsymbol{G}_{m,L}$.

The reconstructed $u_{m.L}(t)$ can approximated $u(t)$ at a short time interval $[t_{m+M}, t_{m+L-M}]$ for some given M. Then define a window w as

$$w = w_n(t, L, M, K) = \begin{cases} 0 & \text{if } t \notin (\tau_n, \sigma_{n+1}], \\ \theta_n(t) & \text{if } t \in (\tau_n, \sigma_n], \\ 1 & \text{if } t \in (\sigma_n, \tau_{n+1}], \\ 1 - \theta_{n+1}(t) & \text{if } t \in (\tau_{n+1}, \sigma_{n+1}], \end{cases} \tag{2.30}$$

where $\theta_n(t)$ are chosen or arbitrary functions and K specifies the number of trigger times over which consecutive windows overlap, $\tau_n = t_{n,J+M}$, $\sigma_n = t_{n,J+M+K}$, $J = L - 2M - K$.

Finally, the overall reconstruction of the original signal $u(t)$ is

$$u_{L.M.K}(t) = \sum_{n \in \mathbf{Z}} w_n(t, L, M, K) x_{nJ,L}(t). \tag{2.31}$$

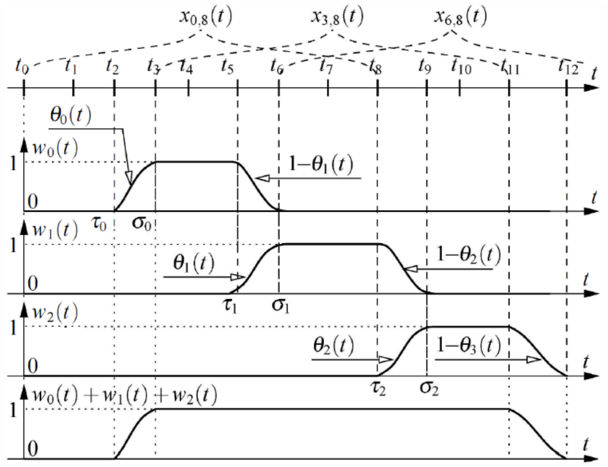


Fig. 2.7. [17] The stitching algorithm of reconstruction. $t_0 = 0$, $L = 8$, $M = 2$, $K = 1$.

For the standard reconstruction algorithm proposed by Lazar et al [3], the reconstruction is carried out by l iterations. The resulting reconstruction error is bounded by $\|u - u_l\|_2 \le \left(\frac{2C\delta}{b-c}\frac{\Omega}{\pi}\right)^{l+1} \|u\|_2$, where u_l is the reconstruction of u after l iterations.

For the alternative reconstruction algorithm proposed by Florescu et al [16], the estimation of the inverse function $\bar{\psi}'$ is given by $\left(\|\bar{\psi}' - \bar{\psi}'_M\|_2\right)^2 \le \left(\frac{c}{b}\right)^{2M} \frac{b+c}{b-c} \left(\|\bar{\psi}'\|_2\right)^2$ $\forall M \in \boldsymbol{N}$ and this can be used to quantify the reconstruction error.

The stitching algorithm [17] has high accuracy in a reduced range $[t_{m+M}, t_{m+L-M}]$ for some given M. However, the reconstruction error is large outside this range. The error can be quantified via experiments.

In terms of decoding speed, Lazar's algorithm [3] requires the calculation of $\boldsymbol{g}$, $\boldsymbol{G}$ and $\boldsymbol{q}$ for all spiking times. Florescu's algorithm [16], however, only needs to calculate $\overline{\boldsymbol{Q}}^{+}$ and $\overline{\boldsymbol{g}}_2$ offline, and only a few additions and multiplications are required to reconstruct subsequent spikes. The stitching algorithm [17] requires more calculation than the standard algorithm proposed in [3] since it calculates also the overlapped windows of spiking times. However, the stitching algorithm reduces the waiting time for a system by selecting the reconstruction windows.

2.5. Multi-channel time encoding and decoding

Lazar and Pnevmatikakis presented a single-input-multi-output TEM model in 2008 [18]. The input signal is filtered by N linearly independent linear filters and fed to N different IF neurons. Finally, the multi-channel TEM outputs N spike trains as shown in Fig. 2.8.

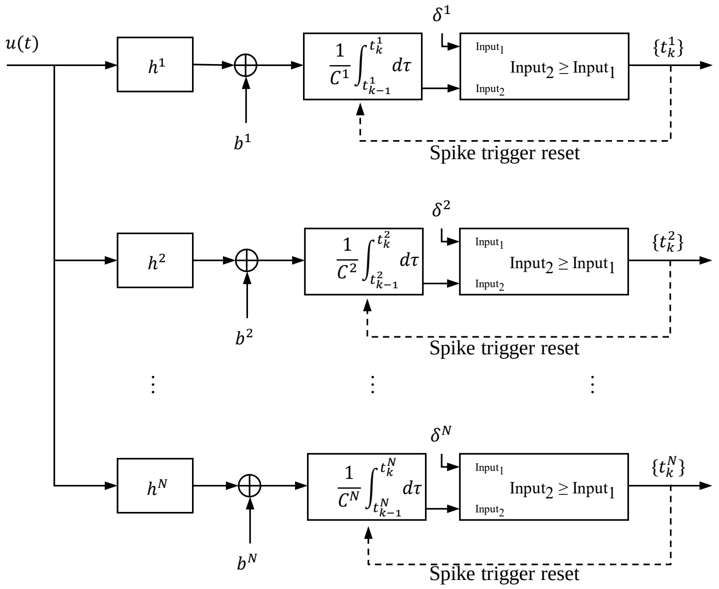


Fig. 2.8. Multi-channel TEM.

For the jth channel, the neuron fires a spike when

$$\frac{1}{C^j}\int_{t_k^j}^{t_{k+1}^j}\left((h^j * u) + b^j\right)(\tau)d\tau = \delta^j, \tag{2.32}$$

where $*$ denotes the convolution operator and $j = 1,2,\dots N$.

The filters $\{h^j\}$ are linearly independent if there do not exist real numbers $a_j, j = 1,2,\dots N$, not all equal to zero, and $\alpha^j, j = 1,2,\dots N$ such that

$$\sum_{j=1}^{N} a_j (h^j * g)(t - \alpha^j) = 0. \tag{2.33}$$

Define $\mathcal{T}_u^j = \{t_k^j\}_{k\in\mathbf{Z}}$, $\tilde{h}^j = h^j(-\,\cdot)$, and $\phi_{k,j}^{\mathcal{T}_u^j} = \tilde{h}^j * g * \mathbf{1}_{[t_k^j, t_{k+1}^j)}$, then the input signal u can be recovered as

$$u = \sum_{j=1}^{N}\sum_{k\in\mathbf{Z}} c_k^j \psi_k^j, \tag{2.34}$$

with $\boldsymbol{c} = [\boldsymbol{c}^1, \dots, \boldsymbol{c}^N]^T = \boldsymbol{G}^{+}\boldsymbol{q}$, $\boldsymbol{q} = [\boldsymbol{q}^1, \dots, \boldsymbol{q}^N]^T$, $q_k^j = \langle u, \phi_{k,j}^{\mathcal{T}_u^j}\rangle_{L^2}$, and

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}^{11} & \dots & \boldsymbol{G}^{1N} \\ \dots & \dots & \dots \\ \boldsymbol{G}^{N1} & \dots & \boldsymbol{G}^{NN} \end{bmatrix}, \boldsymbol{G}_{k,l}^{nj} = \int_{t_k^n}^{t_{k+1}^n} (h^n * \tilde{h}^j * g)(\tau - s_l^j) d\tau, \tag{2.35}$$

where $\psi_k^j = (\tilde{h}^j * g)(\cdot - s_k)$, and c_k^j are suitable coefficients, provided that

$$\sum_{j=1}^{N} \frac{1}{c^j \delta^j} \left(b^j - c \int_{\mathbf{R}} \left| h^j(s) \right| ds \right) > \frac{\Omega}{\pi}, |u(t)| \leq c. \tag{2.36}$$

2.6. Conclusions

This Chapter reviewed the theoretical foundations and practical algorithms for common time encoding (sampling) and decoding (reconstruction) methodologies using crossing and integrate-and-fire Time Encoding Machines. Specific conditions and algorithms for reconstruction of the original signal from spike-time sequences are given.

Both C-TEMs and IF-TEMs are able to map a continuous input signal into a spiking signal. The time encoding processes can also be treated as sampling processes. However, the sampling carried out by IF-TEMs are significantly different from that carried out by C-TEMs. C-TEMs sample the original signal while IF-TEMs sample the integral of the biased original signal. The sampling carried out by IF-TEMs can model the behaviour of spiking neurons in a brain [19]: the membrane potential of a spiking neuron is an integral, and the neuron fires an impulse signal when the potential reaches the neuron threshold. Therefore, the research on IF-TEMs can also benefit the field of biology. The reconstruction methodologies for C-TEMs and IF-TEMs are both based on iterative algorithms. It is proved that reconstruction algorithms for IF-TEMs have no information loss, which is not discussed in the case of C-TEMs. Therefore, considering the above advantages, this thesis focuses on the research of IF-TEMs.

A limitation of existing reconstruction algorithms is that they involve processing batches of time encoded samples which makes it challenging to incorporate TEMs into real-time practical applications. This issue is addressed in Chapter 3 which introduces a real-time decoding (reconstruction) algorithm suitable for control system implementation.

Chapter 3
Real-time algorithms for time decoding machines

3.1. Introduction

Lazar [3] has presented the framework of the ideal IF-TEMs, which are non-uniform sampling devices that map the amplitude of an analogue signal to a strictly increasing time sequence. A number of decoding algorithms have been proposed to reconstruct the original continuous signal from the discrete spike time sequence [1, 2, 20].

A major drawback of existing decoding algorithms is that these require the entire spike time sequence to perform the reconstruction.

The alternative proposed in [21] is to decode the signal over short time intervals $[t_m, t_{m+N}]$, based on the finite sequence of spike times $\{t_m, t_{m+1}, \dots, t_{m+N}\}$generated such that

$$u(t) = \boldsymbol{g}\boldsymbol{Q}^{+}\boldsymbol{q}, t \in [t_m, t_{m+N}], \tag{3.1}$$

where $\boldsymbol{g} = [g(t - s_{k+1})]^T$ and $g(t) = \frac{sin(\Omega t)}{\pi t}$, $s_{k+1} = \frac{t_{k+1}+t_k}{2}$, $\boldsymbol{Q}^{+}$ is the pseudo-inverse of $\boldsymbol{Q}$, $[\boldsymbol{Q}]_{k,l} = \int_{t_k}^{t_{k+1}} g(\tau - s_l)d\tau,\ k, l \in \boldsymbol{Z}^{+},\ \boldsymbol{q} = [-b(t_{k+1} - t_k) + C\delta]$.

Overall signal reconstruction involves stitching together the reconstructions over successive time intervals [21].

This approach however is still not ideal for implementation in real-time applications such as real-time control since from (2.11) it follows that

$$\frac{(N+1)C\delta}{b+c} \leq t_{m+N} - t_m \leq \Delta t \tag{3.2}$$

In essence, the time window used in reconstruction will have to be less than the uniform sampling period Δt used in the implementation of the real-time control system. This in turn will demand a higher firing rate (i.e. a smaller δ) for the IF-TEM that is at least ten times higher than the one dictated by the signal bandwidth in [21].

This chapter introduces a fast decoding algorithms that recovers in real-time the signal from the single- and multi-channel spike time sequences generated by IF-TEMs. Error analysis is presented and an upper bound for the approximation error is derived, which is a function of the TEM parameters and the bandwidth of the signal.

The chapter is organised as follows. Section 3.2 and 3.3 introduces the new fast reconstruction algorithms for single- and multi-channel IF-TEMs. Section 3.4 derives an upper bound for the reconstruction error introduced by the proposed algorithm. Numerical simulations demonstrating the performance of the algorithms are presented in Section 3.5. Final conclusions are provided in Section 3.6.

3.2. Real-time algorithms for single-channel Time Decoding Machines

Let $\{t_k\}$ be the spike time sequence generated by an IF-TEM with parameters $\{\delta, b, C\}$.

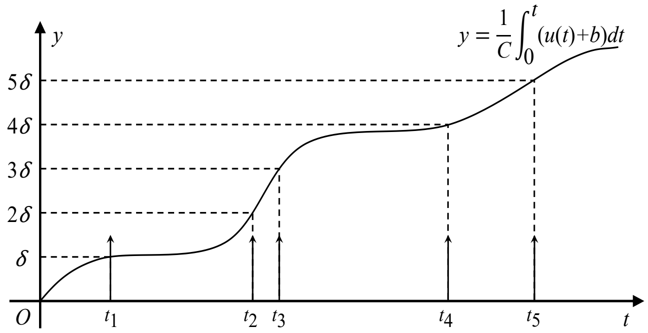


Fig. 3.1. Plot of $\Phi(t)$ showing the relationship between spike times and threshold values.

Define $\Phi(t)$ as

$$\Phi(t)=\frac{1}{C}\int_0^t(u(t)+b)dt \tag{3.3}$$

and therefore,

$$u(t)=C\Phi'(t)-b. \tag{3.4}$$

The idea is to reconstruct $u(t)$ at $t=s_{k+1}=\frac{t_k+t_{k+1}}{2}$ by approximating the derivative of $\Phi(t)$ at that point using centred differencing

$$\Phi'(s_{k+1})\approx\frac{\Phi(t_{k+1})-\Phi(t_k)}{t_{k+1}-t_k}, \tag{3.5}$$

The original signal $u(t)$ at $t=s_{k+1}$ is approximated as

$$u(s_{k+1})=C\Phi'(s_{k+1})-b\approx C\frac{\Phi(t_{k+1})-\Phi(t_k)}{t_{k+1}-t_k}-b=\tilde{u}(s_{k+1}). \tag{3.6}$$

According to (2.10),

$$\Phi(t_k)=\frac{1}{C}\int_0^{t_k}(u(t)+b)dt=k\delta,\quad k\in\boldsymbol{Z}^+, \tag{3.7}$$

such that

$$\tilde{u}(s_{k+1})=C\frac{\Phi(t_{k+1})-\Phi(t_k)}{t_{k+1}-t_k}-b$$

$$=\frac{C\delta}{t_{k+1}-t_k}-b, \tag{3.8}$$

The reconstructed signal $\tilde{u}(t)$ is a piecewise constant function over the non-uniform partition defined by the spike time sequence $\{0,t_1,\dots,t_k,t_{k+1},\dots\}$ i.e., $\tilde{u}(t)=\tilde{u}(s_{k+1})$ for $t\in(t_k,t_{k+1}]$, $k=1,2,\dots$.

Compared with Lazar's original reconstruction algorithm [3], the proposed algorithm uses only two most recent spikes, and does not involve block-based ill-conditioned algebraic inversions. From (2.11) it follows that the maximum waiting time between two successive signal reconstructions is upper bounded by $\frac{C\delta}{b-c}$ under the assumption that $|u(t)|\le c<b$.

3.3. Reconstruction algorithm for multi-channel IF-TEMs

The algorithm presented above can be extended to multi-channel TDMs.

A multi-channel TEM [22] consists of M identical integrate-and-fire neurons, i.e., same parameters C, δ and b, but have different initial integrator shifts $\varDelta^{(i)}, i = 1, 2, \dots, M$. Each channel ouputs a spike train $\{t_k^{(i)}\}$, where $k \in \boldsymbol{Z}^{+}$ and $i = 1, 2, \dots, M$. Multi-channel time-encoding allows encoding, transmitting and decoding signals with bandwidths that are M times the theoretical maximum bandwidth corresponding to a single-channel TEM with the same parameters. The structure of the M-channel TEM is illustrated in Fig. 3.2.

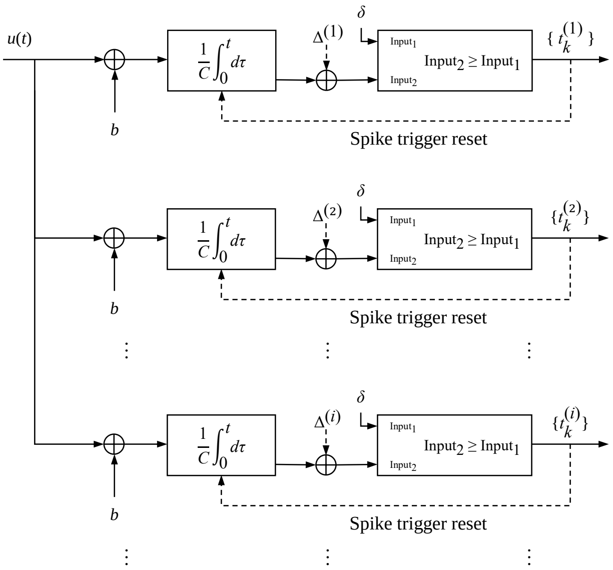


Fig. 3.2. Block diagram of a multi-channel TEM. The original signal $u(t)$ is input to each channel of the TEM. The bias of all the channels is b.

It is assumed that the integrator shifts satisfy $\delta > \varDelta^{(1)} > \varDelta^{(2)} > \cdots > \varDelta^{(M)} \geq 0$ such that the positive "distance" between ith channel and $(i+1)$th channel $D_{i,i+1}$ is given by

$$D_{i,i+1} = \varDelta^{(i)} - \varDelta^{(i+1)} > 0, i = 1, \dots, M-1, D_{M,1} = \delta + \varDelta^{(M)} - \varDelta^{(1)} > 0. \tag{3.9}$$

The outputs of the integrators are given by

$$y_i(t) = \begin{cases} \left(\varDelta^{(i)} + \frac{1}{C}\int_0^{t_1^{(i)}} (u(t) + b)dt\right) MOD\ \delta, & 0 \leq t \leq t_1^{(i)} \\ \left(\frac{1}{C}\int_{t_k^{(i)}}^{t_{k+1}^{(i)}} (u(t) + b)dt\right) MOD\ \delta, & t > t_1^{(i)},\ k = 1,2, \dots \end{cases} \tag{3.10}$$

and satisfy

$$y_{i+1}(t) = \left(y_i(t) + \varDelta^{(i+1)} - \varDelta^{(i)}\right) MOD\ \delta,\ i = 1, \dots, M-1,$$
$$y_1(t) = \left(y_M(t) + \delta + \varDelta^{(M)} - \varDelta^{(1)}\right) MOD\ \delta, \tag{3.11}$$

where MOD is the modulo operation.

As a consequence of the non-zero shift, the M single-channel TEMs will spike in order such that

$$0 < t_k^{(1)} < t_k^{(2)} < \cdots < t_k^{(M)}, \forall k,$$

$$t_{k-1}^{(i+1)} < t_k^{(i)} < t_k^{(i+1)}, \forall k, \forall i \ = \ 1,2,\dots,M. \tag{3.12}$$

where $t_k^{(i)}$ is the kth spike arriving time of the ith neuron. After firing the first spike, the neurons reset and the shifts $\Delta^{(i)}$ are eliminated.

Each neuron fires a first spike at $t \ = \ t_1^{(i)}$ such that

$$\Delta^{(i)} + \frac{1}{C}\int_0^{t_1^{(i)}} (u(t) + b)dt = \delta. \tag{3.13}$$

After firing the first spike, the neurons reset and the shift $\Delta^{(i)}$ is hence eliminated, which leads to

$$\frac{1}{C}\int_{t_k^{(i)}}^{t_{k+1}^{(i)}} (u(t) + b)dt = \delta, k \in \boldsymbol{Z}^{+}. \tag{3.14}$$

Evaluating the function $\Phi(t)$ in (3.13) at $t \ = \ t_k^{(i)}$ gives

$$\Phi(t_k^{(i)}) = \frac{1}{C}\int_0^{t_k^{(i)}} (u(t) + b)dt = k\delta - \Delta^{(i)}, k \in \boldsymbol{Z}^{+}. \tag{3.15}$$

As in the single-channel case, the original signal is reconstructed by estimating the derivative of $\Phi(t)$ as

$$\begin{aligned} u\big(s_k^{(i+1)}\big) &= C\Phi'\big(s_k^{(i+1)}\big) - b \\ &\approx C\frac{\Phi(t_k^{(i+1)}) - \Phi(t_k^{(i)})}{t_k^{(i+1)} - t_k^{(i)}} - b \\ &= \frac{C(\Delta^{(i)} - \Delta^{(i+1)})}{t_k^{(i+1)} - t_k^{(i)}} - b, \end{aligned} \tag{3.16}$$

where $s_k^{(i+1)} \ = \ \frac{t_k^{(i)} + t_k^{(i+1)}}{2}$.

Notice that the spike arriving time $t_k^{(i+1)}$ and $t_k^{(i)}$ are from different channels. As in [22], the spike trains from different channels are combined into a single spike train ordered by arrival time.

Consider two different spike arriving times $t_{k+1}^{(i)}$ and $t_k^{(j)}$, where $i, j \ = \ 1,2,\dots,M$. According to (3.15),

$$\Phi(t_{k+1}^{(i)}) = \frac{1}{C}\int_0^{t_{k+1}^{(i)}} (u(t) + b)dt = (k+1)\delta - \Delta^{(i)},$$

$$\Phi(t_k^{(j)}) = \frac{1}{C}\int_0^{t_k^{(j)}} (u(t) + b)dt = k\delta - \Delta^{(j)}. \tag{3.17}$$

Then $\Phi(t_{k+1}^{(i)}) - \Phi(t_k^{(j)}) = \delta - \Delta^{(i)} + \Delta^{(j)}$.

Under the assumption of $\delta > \Delta^{(1)} > \Delta^{(2)} > \cdots > \Delta^{(M)} \geq 0$, it follows that

$$\delta - \Delta^{(i)} + \Delta^{(j)} \geq \delta - \Delta^{(1)} + \Delta^{(M)} > 0. \tag{3.18}$$

Therefore $\Phi(t_{k+1}^{(i)}) - \Phi(t_k^{(j)}) > 0$. Considering $\Phi(t)$ is monotonically increasing, it holds that $t_{k+1}^{(i)} > t_k^{(j)}$ and thus these spikes in the new combined spike train are ordered in groups of M. The $(kM+1)$th, $(kM+2)$th, …, $(kM+i)\,th$, …, $(k+1)M$th spikes are ordered as $t_{k+1}^{(1)}$, $t_{k+1}^{(2)}$, …, $t_{k+1}^{(i)}$, …, $t_{k+1}^{(M)}$, respectively.

The spike times in the combined spike train $\{\tilde{t}_k\}, k \in \boldsymbol{Z}^{+}$, are ordered as

$$\{\ \underbrace{t_1^{(1)},\dots,t_1^{(M)}}_{first\ spike\ group}\ ,\ \ \underbrace{t_2^{(1)},\dots,t_2^{(M)}}_{second\ spike\ group}\ \ ,\dots,\ \underbrace{t_k^{(1)},\dots,t_k^{(M)}}_{kth\ spike\ group}\ ,\dots\}.$$

Therefore, the nth spike of the combined spike train $\tilde{t}_n$ is $t_{\lfloor\frac{n-1}{M}\rfloor+1}^{(MOD(n-1,M)+1)}$.

From (3.15) it follows that

$$\frac{1}{C}\int_{t_k^{(i)}}^{t_k^{(i+1)}}(u(t)+b)dt = \Delta^{(i)} - \Delta^{(i+1)}. \qquad (3.19)$$

In practice the integrator shifts are chosen such that

$$\Delta^{(M)} = \frac{\delta}{2M}$$

$$\Delta^{(i)} - \Delta^{(i+1)} = \frac{\delta}{M}, for\ i = 1,\dots,M-1 \qquad (3.20)$$

The combined spike train is therefore used to reconstruct the original signal as in the single-channel case, using only the two most recent consecutive spikes. In particular,

1) When $MOD(k,M) \neq 0$, the two consecutive spikes are from the same group, i.e., $\lfloor\frac{k}{M}\rfloor + 1 = \lfloor\frac{k-1}{M}\rfloor + 1$.

$$\begin{aligned}
u(\tilde{s}_{k+1}) &\approx C\frac{\Phi(\tilde{t}_{k+1}) - \Phi(\tilde{t}_k)}{\tilde{t}_{k+1} - \tilde{t}_k} - b \\
&= C\frac{\Phi\left(t_{\lfloor\frac{k}{M}\rfloor+1}^{(MOD(k,M)+1)}\right) - \Phi\left(t_{\lfloor\frac{k-1}{M}\rfloor+1}^{(MOD(k-1,M)+1)}\right)}{\tilde{t}_{k+1} - \tilde{t}_k} - b \\
&= \frac{C(\Delta^{(MOD(k-1,M)+1)} - \Delta^{(MOD(k,M)+1)})}{\tilde{t}_{k+1} - \tilde{t}_k} - b, \\
&= \frac{C\delta}{M(\tilde{t}_{k+1} - \tilde{t}_k)} - b, \qquad (3.21)
\end{aligned}$$

where $\tilde{s}_{k+1} = \frac{\tilde{t}_k + \tilde{t}_{k+1}}{2}$.

2) When $MOD(k,M) = 0$, two consecutive spikes are from different groups.

$$\begin{aligned}
u(\tilde{s}_{k+1}) &\approx C\frac{\Phi(\tilde{t}_{k+1}) - \Phi(\tilde{t}_k)}{\tilde{t}_{k+1} - \tilde{t}_k} - b \\
&= C\frac{\Phi(t_k^{(1)}) - \Phi(t_{k-1}^{(M)})}{\tilde{t}_{k+1} - \tilde{t}_k} - b \\
&= C\frac{(k\delta - \Delta^{(1)}) - ((k-1)\delta - \Delta^{(M)})}{\tilde{t}_{k+1} - \tilde{t}_k} - b \\
&= C\frac{\Delta^{(M)} - \Delta^{(1)} + \delta}{\tilde{t}_{k+1} - \tilde{t}_k} - b \\
&= C\frac{\delta}{M(\tilde{t}_{k+1} - \tilde{t}_k)} - b. \qquad (3.22)
\end{aligned}$$

As in the single-channel case, the reconstructed signal $\tilde{u}(t)$ is a piecewise constant function over the non-uniform partition defined by the combined spike time sequence $\{0, \tilde{t}_1, \dots, \tilde{t}_k, \tilde{t}_{k+1}, \dots\}$ i.e., $\tilde{u}(t) = \tilde{u}(s_{k+1})$ for $t(\tilde{t}_k, \tilde{t}_{k+1}]$.

3.4. Reconstruction error analysis

The reconstruction error at mid points is given by

$$e(s_{k+1}) = u(s_{k+1}) - \tilde{u}(s_{k+1}). \tag{3.23}$$

Let

$$h(t) = C\Phi(t) - bt. \tag{3.24}$$

From (3.13) it holds that

$$h(t) = \int_0^t u(t)dt \text{ and } h'(t) = u(t). \tag{3.25}$$

Since the algorithm approximates $h'(t)$ at $t = s_{k+1}$ using a second-order centred differencing method [23]

$$h'(s_{k+1}) \approx \frac{h(t_{k+1})-h(t_k)}{t_{k+1}-t_k}. \tag{3.25}$$

Combining the second-order Taylor expansion around s_{k+1} gives

$$u(s_{k+1}) = h'(s_{k+1}) = \frac{h(t_{k+1})-h(t_k)}{t_{k+1}-t_k} - (t_{k+1} - t_k)^2 \frac{h'''(\zeta_1)+h'''(\zeta_2)}{48}, \tag{3.26}$$

where $\zeta_1 \in [t_k, s_{k+1}]$ and $\zeta_2 \in [s_{k+1}, t_{k+1}]$. Hence

$$e = u(s_{k+1}) - \tilde{u}(s_{k+1}) = -(t_{k+1} - t_k)^2 \frac{h'''(\zeta_1)+h'''(\zeta_2)}{48}. \tag{3.27}$$

Given that the bandwidth of $u(t)$ is Ω and $|u(t)| \leq c$, it follows that

$$|h'''(\zeta)| = |u''(\zeta)| \leq 4\pi^2\Omega^2 c, \tag{3.28}$$

and combining this with $t_{k+1} - t_k \leq \frac{C\delta}{(b-c)}$ leads to

$$|e| \leq \frac{1}{6}\left(\frac{\pi\Omega C\delta}{b-c}\right)^2 c. \tag{3.29}$$

The reconstruction error can be reduced by increasing spike density which can be achieved by decreasing δ or increasing b. Alternatively, one can employ multiple TEMs and initialize the integral with different values. Combining the output spike trains of these TEMs leads to a denser spike train.

3.5. Numerical Study

Example 1. Single-channel encoding and decoding

The new decoding algorithm is demonstrated using a synthetic signal $u(t)$ generated as the sum of 25 sinusoidal functions with uniformly distributed random amplitudes and frequencies. The frequencies are chosen to be uniformly distributed in interval $(0,80]$ and their amplitudes are uniformly distributed in interval $[-0.5,0.5]$. $u(t)$ is then given by

$$u(t) = \sum_{n=1}^{25} A_n \sin(\omega_n t). \tag{3.30}$$

The signal $u(t)$ was encoded using an integrate-and-fire, single-channel TEM. The parameters of the simulation are given in Table 3.1.

The TEM-generated spike train consisting of 195 spikes was decoded using the proposed real-time algorithm. For comparison, the generated spike train was also decoded using Lazar's algorithm [3]. In practice, while the fast algorithm would reconstruct the algorithm in real-time as spikes are being generated, Lazar's algorithm relies on the full spike train.

Table 3.1. Simulation parameters.

Parameter	Explanation	Value
$\boldsymbol{T_{sim}}$	Overall simulation time	0.4 seconds
$\boldsymbol{dt}$	Sampling period	1×10^{-7} seconds
$\boldsymbol{\Omega}$	Signal bandwidth	80 rad/s
$\boldsymbol{c}$	Signal magnitude bound	1.5
$\boldsymbol{\delta}$	IF neuron threshold	0.01
$\boldsymbol{b}$	IF neuron bias	5
$\boldsymbol{C}$	IF integration constant	1

The original signal $u(t)$ and the piece-wise signal $\tilde{u}(t)$ decoded, using the real-time algorithm, are shown in Fig. 3.3.

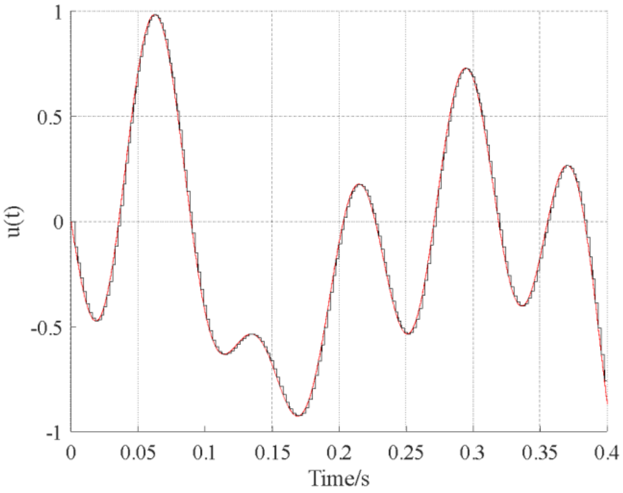


Fig. 3.3. Single-channel decoding using the real-time algorithm. Original signal (red) superimposed on the decoded (black) signals.

The decoded signal is significantly close to the original signal. A Zero-Order hold is employed here which leads to the constant value between estimation points.

The spike intervals in this case are approximately 0.002 seconds. Therefore, one only need to wait for about 0.002 seconds to carry out the next reconstruction step. Between estimation points, the decoded signal is still output with a constant value, which enables real-time signal processing. Notice that there is a delay of the decoded signal. This is introduced by the Zero-Order hold and the maximum delay is determined by the maximum spike interval.

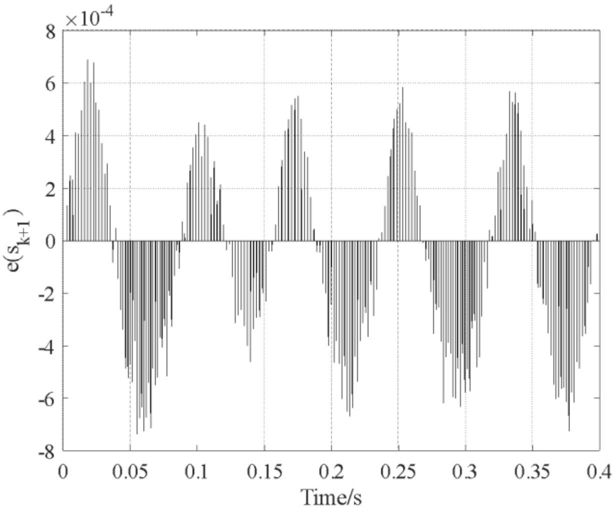


Fig. 3.4. Decoding error $e(s_{k+1})$, $k = 1, \dots, 194$ for the proposed fast algorithm. Maximum error magnitude $|e(s_{k+1})|_{max} = 7.4 \times 10^{-4}$ and the Mean Squared Error is 7.6×10^{-12}.

The error is larger at the extreme points of the original signal. This is because the estimating the derivative of $\Phi(t)$ using centered differencing has larger error at these points.

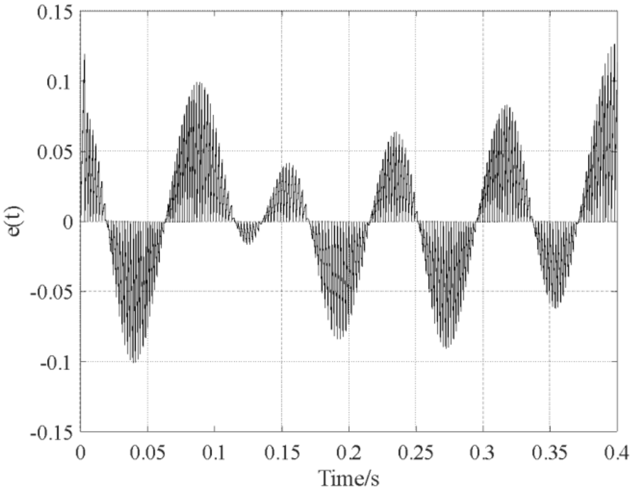


Fig. 3.5. Decoding error $e(t) = \tilde{u}(t) - u(t)$ for the proposed fast algorithm. Maximum error magnitude $|e(t)|_{max} = 1.27 \times 10^{-2}$ and the Mean Squared Error is 0.0012.

The decoding error is small at the estimation points and increases when the TDM is waiting for the next spike. Therefore, in order to reduce the decoding error, one can minimize the spike intervals by increasing the bias b or reducing the threshold δ.

The original and decoded signals, using Lazar's algorithm are shown in Fig. 3.6. The reconstruction errors for the fast algorithm, evaluated at $t = s_{k+1}$, are significantly smaller than the errors introduced by Lazar's algorithm (see Figs. 3.4, 3.7, 3.8).

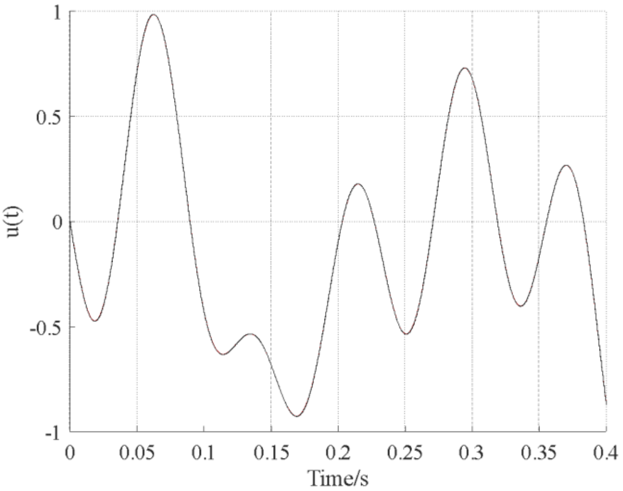


Fig. 3.6. Single-channel decoding using Lazar’s algorithm. Original signal (red) superimposed on the decoded (black) signals.

Compared with the decoding using the proposed fast algorithm, Lazar’s algorithm has much smaller reconstruction error. However, this reconstruction is carried out when the spike train is fully transmitted. In this case, one must wait for 0.4 seconds to carry out the decoding.

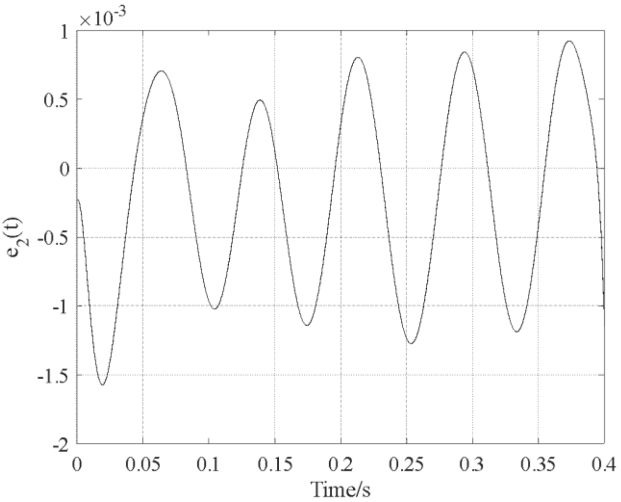


Fig. 3.7. Decoding error $e_2(t) = \tilde{u}_2(t) - u(t)$ for Lazar’s algorithm. Maximum error magnitude $|e_2(t)|_{max} = 1.6 \times 10^{-3}$ and the Mean Squared Error is 5.4×10^{-7}.

Also, the error is larger at the extreme points of the original signal.

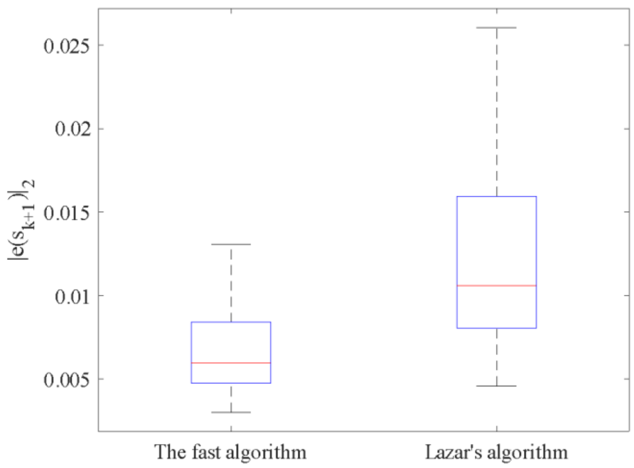


Fig. 3.8. Box plots of the l_2 error norms $|e(s_{k+1})|_2$ for 100 replications of the synthetic signal for the real-time and Lazar's algorithms.

Fig. 3.8 also indicates that the upper error bound derived for the fast algorithm is relatively small. However, when the error is evaluated at the original sampling points, Lazar's algorithm outperforms the real-time decoding algorithm. This is a consequence of the fact that digital-to-analogue conversion is performed using a zero-order hold method that produces a piece-wise constant signal over the non-uniform partition defined by the spike time sequence. Alternative, local interpolation or quasi-interpolation approaches used to reconstruct non-uniformly sampled signals can achieve near perfect reconstruction on the uniform grid [24].

Table 3.2 summarises the performance of the real-time algorithm for different values of the bias and threshold parameters, b and δ, of the TEM. All other simulation parameters are as given in Table 3.1. The maximum reconstruction error of Lazar's algorithm $|e_2(t)|_{max}$ and the theoretical upper bound derived in Section 3.3, are also listed in Table 3.2.

Table 3.2. Maximum decoding errors and theoretical upper error bounds for different values of the bias and threshold parameters.

b	δ	Maximum error of the fast algorithm $|e(s_{k+1})|_{max}$	Theoretical upper error bound of the fast algorithm	Maximum error of Lazar's algorithm $|e_2(t)|_{max}$
3	0.01	2.3×10^{-3}	1.8×10^{-2}	4.8×10^{-3}
4	0.01	1.2×10^{-3}	6.4×10^{-3}	2.5×10^{-3}
5	0.01	7.4×10^{-4}	3.3×10^{-3}	1.6×10^{-3}
5	0.02	2.8×10^{-3}	1.3×10^{-2}	7.0×10^{-3}
5	0.03	6.3×10^{-3}	2.9×10^{-2}	2.9×10^{-2}

As expected, by increasing the bias b and decreasing the neuron threshold δ, increases the spike rate which reduces decoding error. Overall, the real-time decoding algorithm consistently outperforms Lazar's algorithm in terms of decoding accuracy at the mid-point of the spike interval.

Example 2. Multi-channel encoding and decoding

This example illustrates the performance of the decoding algorithm when the original signal $u(t)$ is encoded using a multi-channel TEM with $M = 10$ channels. All 10 integrate-and-fire neurons have identical thresholds and bias parameters but have different initial membrane potentials i.e., integral shifts $\Delta^{(i)}$ that are set according to (3.20).

Here the neuron threshold is $\delta = 0.1$ that is 10 times of that in *Example 1*. The threshold is set larger in order to illustrate that the decoded signal from the output of multi-channel TEM with larger threshold is close to that from the output of single-channel TEM with smaller threshold. Therefore, one can construct TEMs and TDMs without the need of neurons or TEMs with small threshold, which simplifies the construction. All other simulation parameters are those in Table 3.1. Given these settings, the total number of spikes in the combined spike train ($N = 194$) is the same as for the single-channel case.

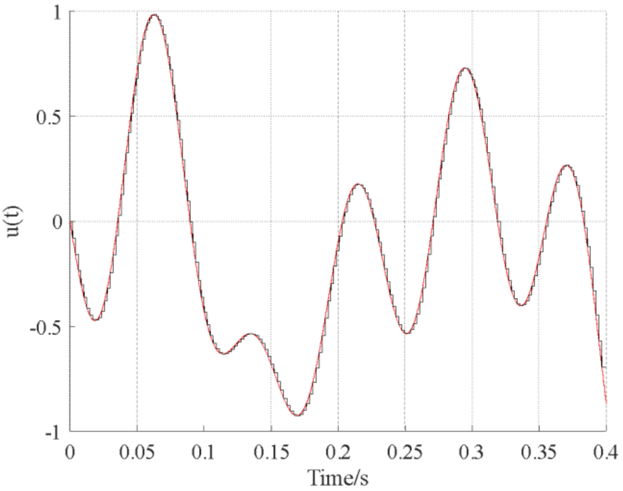


Fig. 3.9. Multi-channel ($M = 10$) decoding using the proposed fast algorithm. Original signal (red) superimposed on the decoded signal (black).

Compared with the single-channel case, this 10-channel decoding reaches similar results. The average spike interval is also 0.002 seconds, which indicates that the waiting time remains the same.

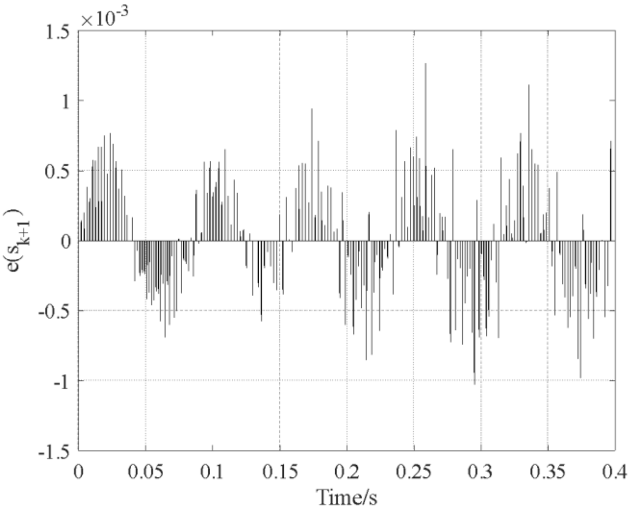


Fig. 3.10. Multi-channel ($M = 10$) decoding error $e(s_{k+1}),\ k = 1, \dots, 194$ for the proposed fast algorithm. Maximum error magnitude $|e(s_{k+1})|_{max} = 1.3 \times 10^{-3}$. The Mean Squared Error is 7.6×10^{-12}.

The Mean Squared Error is the same as that of the single-channel case, which shows that the estimation error of the single- and multi-channel cases are at the same level. The estimation error is also larger at the extreme points of the original signal, which is similar with the single-channel case.

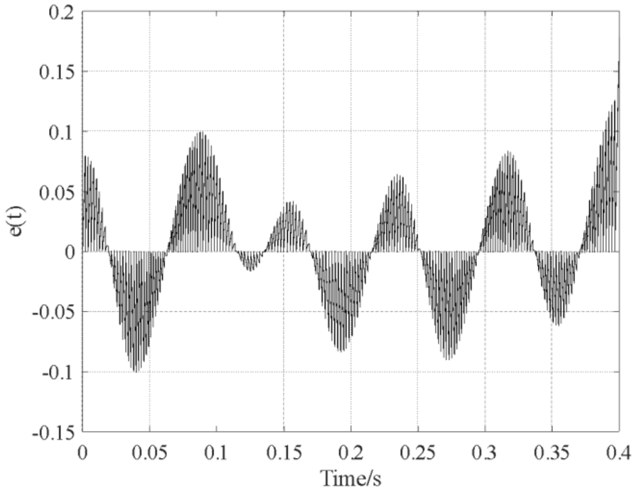


Fig. 3.11. Multi-channel ($M = 10$) decoding error $e(t) = \tilde{u}(t) - u(t)$ for the proposed fast algorithm. Maximum error magnitude $|e(t)|_{max} = 0.1261$. The Mean Squared Error is 0.0012.

The decoding error is almost the same as that of the single-channel case. However, the multi-channel time encoding does not require the neuron threshold to be small. The threshold of the multi-channel TEM is 10 times of the threshold in the single-channel case.

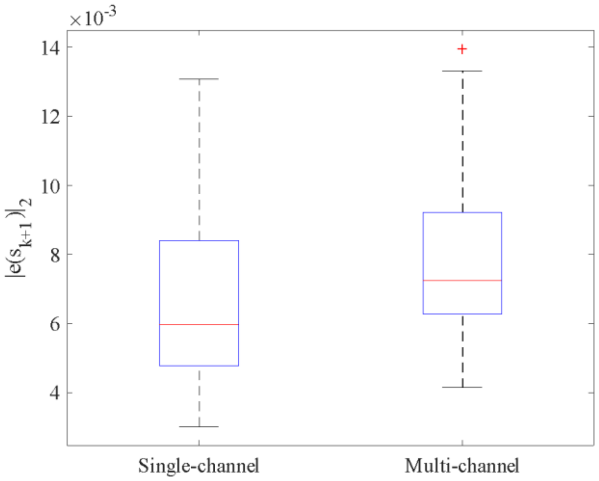


Fig. 3.12. Box plots of the l_2 error norms $|e(s_{k+1})|_2$ over 100 replications of the synthetic signal for the single- and multi-channel encoding/decoding.

It is shown that the estimation error at mid points of spikes of the multi-channel decoding is slightly larger than the single-channel decoding. However, as shown in Fig. 3.11, the overall reconstruction error is significantly close to that of the single-channel decoding. Therefore, one can utilize multi-channel TEMs and TDMs to replace single-channel TEMs and TDMs with simplified implementation that doesn't require small thresholds.

Here it is assumed that the 10th channel fails i.e. the corresponding integrate-and-fire neuron does not output any spikes. The decoded signal and decoding errors are shown in Fig. 3.13 and Fig. 3.14.

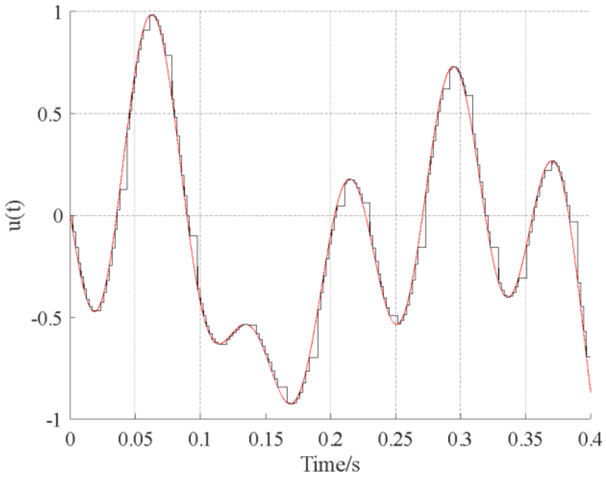


Fig. 3.13. Multi-channel ($M = 10$) decoding using the proposed fast algorithm, assuming the 10th channel fails. Original signal (red) superimposed on the decoded signal (black).

The figure shows that large gaps occur periodically. This is caused by the failure of the 10th channel. A failing channel does not output any spikes, therefore the reconstruction will not be carried out until the next working channel outputs a new spike. Nevertheless, the other estimation points remain unaffected since the estimation only requires two consecutive spikes.

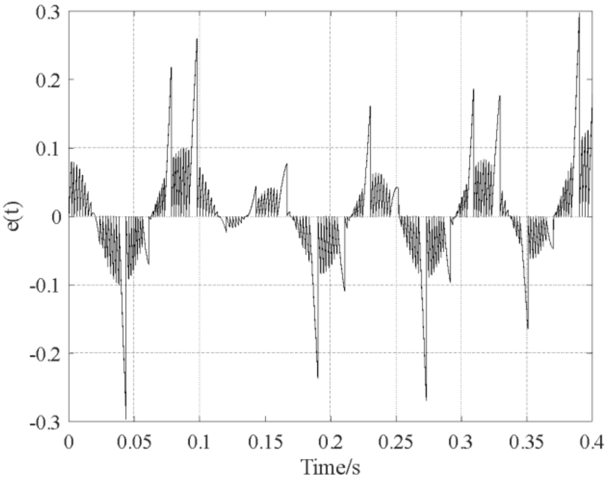


Fig. 3.14. Decoding error $e(t) = \tilde{u}(t) - u(t)$. Maximum error magnitude $|e(t)|_{max} = 0.3122$. The Mean Squared Error is 0.0038.

Compared with the decoding error illustrated in Fig. 3.11, the decoding error here is large when it comes to spikes that should have been output by the failing channel. However, the remaining decoding error is close to that in Fig. 3.11. Therefore, it can be concluded that the error caused by the failure of a channel is not accumulative and the multi-channel TEMs and TDMs are robust to the failure of channels.

3.6. Conclusions

This chapter introduced a new algorithm for decoding spike trains generated by integrate-and-fire neurons, in real-time and without the need to compute pseudo-inverse of matrices [25]. The algorithm used the two most recent spike times to recover the original bandlimited signal at the midpoint of the interval defined by the two spike times. Numerical simulations have shown that, for the proposed algorithm, the accuracy of signal reconstruction on the non-uniform grid was higher than that achieved by Lazar's original reconstruction algorithm [3]. An upper bound for the error was derived and shown through numerical simulations to be relatively sharp.

A multi-channel version of the algorithm was also introduced and demonstrated. It has been illustrated in this chapter that the decoded signal from the output of multi-channel TEM with larger threshold was close to that from the output of single-channel TEM with smaller threshold. This indicates that a multi-channel TEM with larger threshold could model a single-channel TEM with smaller threshold. Therefore, one can employ a multi-channel TEM to encode a signal without the need to construct a single-channel TEM with small threshold, which is challenging in the aspect of hardware realization.

The main benefit of this new algorithm is that it is fast, simple and can decode the spike train in real-time without the need to compute pseudo-inverse of matrices [25]. Furthermore, the accuracy of signal reconstruction on the non-uniform grid is higher than that achieved by Lazar's original reconstruction algorithm [3].

A disadvantage of the proposed algorithm is that the original signal is reconstructed on a non-uniform grid defined by the spike-time sequence. An additional step is required to reconstruct the original signal on a uniform grid – in this chapter a simple zero-order hold method is employed but other more accurate interpolation/quasi-interpolation approaches developed for real-time processing can be used [16, 19]. Ultimately, it is shown that the decoding error can easily be controlled by adjusting the bias and/or threshold parameters of the TEM.

Chapter 4
Spike-based PID control system design

4.1. Introduction

In order to take advantage of time encoding, novel devices that directly use information encoded in the form of spike trains to perform their function will need to be developed. A number of spike-based devices and systems have been developed in recent years [6~8]. Jimenez-Fernandez, et al [6] introduced spike-based PID controller with the concepts of some simple spike operators. The paper implemented a spike-based PID controller using Spikes Integrate & Generate (SI&G), Spike Temporal Derivative (STD) and Spikes Hold & Fire (SH&F) components. These components operate on the number of spikes instead of spike arriving time, which requires a global counter. Mie, et al [7] utilized this spike-based PID controller to implement a spike-based control system where all the internal signals are in the format of spikes. Paper [8] presented the analysis of spike firing rate using Laplace transformation and enabled the application of spike processing strategy in the field of spike-based silicon sensors and robotic actuators.

In general, there is a lack of a control architectures and design methodologies for spike-based controllers that directly use time-encoded signals to derive the control input signals and cancel the global counter [6].

This chapter proposes a spike-based PID control architecture that can implement a continuous-time PID control law derived using a conventional design method, taking advantage of the real-time decoding algorithm proposed in Chapter 3. The spike-based PID controller generates directly the analogue control input so that there is no need for digital-to analogue conversion.

The remainder of this chapter is organized as follows. Section 4.2 describes a closed-loop spike-based control system and introduces the data transmitting methodology in the system. Section 4.3 formulates the spike-based PID control law. Section 4.4 carries out an analysis of the approximation error associated with each of the PID control terms. Section 4.5 presents a numerical simulation study that demonstrates the performance of the spike-based PID control system in comparison with a standard continuous-time PID control implementation. Conclusions and Section 4.6 draws the conclusion.

4.2. Closed-loop spike-based control system

In this section, a remote closed-loop control system is presented. This system consists of two terminals: a command terminal which sends the error signal according to the reference, and an execute terminal which responds to the error signal and send the final system output signal back to the command terminal. Here, the remote communication between the command and execute terminals is in the format of spikes.

A block diagram of the architecture of a networked control system, incorporating TEMs and TDMs such that control and feedback signals are exchanged in the form of time-encoded signals is shown in Fig. 4.1.

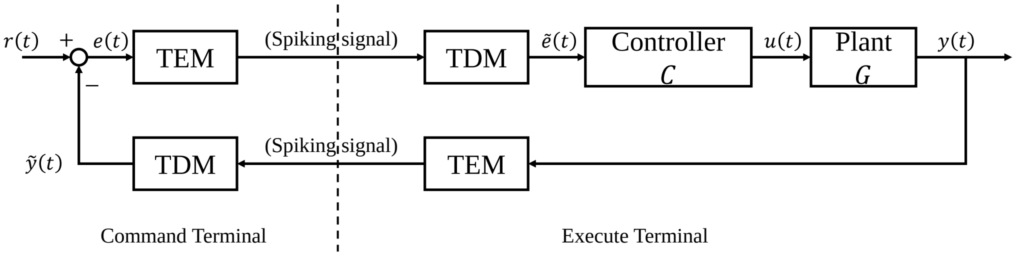


Fig. 4.1. Block diagram of a spike-based closed-loop control system. TEMs encode continuous signals such as $e(t)$ and $y(t)$. The encoded signals are transmitted in the format of spike trains. TDMs decode the spiking signals back to continuous signals. Controller $C(s)$ and plant $G(s)$ respond to these continuous signals.

The proposed networked control implementation has several advantages:

- Energy Efficiency [4].
- Simpler devices used for receiving and sending spiking signals [4].
- Sparse Data Representation.
- Robustness to noise [5].
- Event-Driven Processing.

4.3. Spike-based PID controller

Fig. 4.2. illustrates the block diagram of a spike-based network control system using a spike-based PID controller. The spike-based controller directly responds to the encoded error signal, i.e., the spike train $\{t_k^e\}$.

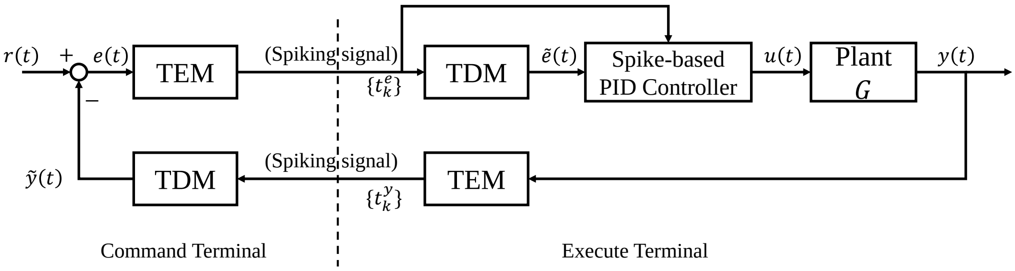


Fig. 4.2. Block diagram of the spike-based network control system with a spike-based PID controller.

The formulation of the proportional, integral and derivative terms of the spike-based PID controller is described below.

Proportional term

The proportional term uses the real-time error signal $\tilde{e}(t)$ generated by the fast decoding algorithm introduced in Chapter 3 as follows

$$P_s(t) = \tilde{e}(s_{k+1}) = \frac{C\delta}{t_{k+1}^e - t_k^e} - b \ \text{ for } t \in [t_k^e, t_{k+1}^e), k = 1,2,3,... \tag{4.1}$$

For the integral and derivative terms, the spike-based PID controller responds to the encoded spike train $\{t_k^e\}$.

Integral term

The IF-TEM encodes the error signal $e(t)$ by sampling the integral of $u(t)$ [3] at time instances t_k such that

$$\frac{1}{C}\int_0^t (u(t)+b)dt = k\cdot\delta, \quad k = 1,2,3,\dots \tag{4.2}$$

It follows that the integral of $e(t)$ at time $t = t_k^e$ is given by

$$I_s(t_k^e) = \int_0^{t_k^e} e(t)dt = k\cdot C\delta - b\cdot t_k^e. \tag{4.3}$$

The integral term is only updated when a new spike arrives and it is constant during the inter-spike interval that is $I_s(t) = I_s(t_k^e)$ for $t_k^e \le t < t_{k+1}^e$. When the next spike arrives, the value of the integral is updated.

Derivative term formulation

The derivative of the error signal $e(t)$ at $t = s_{k+1} = \frac{t_{k+1}^e + t_k^e}{2}$ is approximated as

$$\begin{aligned} e'(s_{k+1}) \approx D_s(s_{k+1}) &= \frac{e(s_{k+1}) - e(s_k)}{s_{k+1} - s_k} \\ &= \frac{\left(\frac{C\delta}{t_{k+1}^e - t_k^e} - b\right) - \left(\frac{C\delta}{t_k^e - t_{k-1}^e} - b\right)}{\frac{t_{k+1}^e + t_k^e}{2} - \frac{t_k^e + t_{k-1}^e}{2}} \\ &= \frac{2C\delta\cdot(-t_{k+1}^e + 2t_k^e - t_{k-1}^e)}{(t_{k+1}^e - t_k^e)(t_k^e - t_{k-1}^e)(t_{k+1}^e - t_{k-1}^e)}. \end{aligned} \tag{4.4}$$

The estimation of the derivative is held as constant between spike mid points so that $D_s(t) = D_s(s_{k+1})$ for $s_k \le t < s_{k+1}$. The estimation is updated when the next spike arrives.

The spike-based PID control law is therefore given by

$$\begin{aligned} u(t) &= k_{ps}\cdot P_s + k_{is}\cdot I_s + k_{ds}\cdot D_s \\ &= k_{ps}\cdot\left(\frac{C\delta}{t_{k+1}^e - t_k^e} - b\right) + k_{is}\cdot(k\cdot C\delta - b\cdot t_k^e) + k_{ds}\cdot\frac{2C\delta\cdot(-t_{k+1}^e + 2t_k^e - t_{k-1}^e)}{(t_{k+1}^e - t_k^e)(t_k^e - t_{k-1}^e)(t_{k+1}^e - t_{k-1}^e)}, \end{aligned} \tag{4.5}$$

where k_{ps}, k_{is} and k_{ds} are proportional, integral and derivative gain of the spike-based PID controller, respectively. P_s, I_s and D_s are updated at $t = t_k^e$.

4.4. Approximation error bound for PID terms

Approximation error bound for the proportional term

The approximation error of the proportional term P_s is equivalent to the reconstruction error of signal $e(t)$. As derived in Chapter 3 (3.29), the approximation error of the proportional term ε_p is bounded by

$$\varepsilon_p \le \frac{1}{6}\left(\frac{\pi\Omega C\delta}{b-c}\right)^2 c. \tag{4.6}$$

Approximation error bound for the integral term

The integral term is calculated every time a spike arrives and is kept constant in between spikes. For $t_k^e \le t < t_{k+1}^e$ it follows that

$$I(t) = \int_0^t e(s)ds = \int_0^{t_k^e} e(t)dt + \int_{t_k^e}^{t_{k+1}^e} e(t)dt = I_s + \int_{t_k^e}^{t_{k+1}^e} e(t)dt. \tag{4.7}$$

Therefore, the approximation error ε_i is bounded by

$$\varepsilon_i = I - I_s = \int_{t_k^e}^{t_{k+1}^e} e(t)dt. \tag{4.8}$$

Assuming $|e(t)| < c, \quad c > 0$ then ε_i is bounded as

$$|\varepsilon_i| = \left|\int_{t_k^e}^{t_{k+1}^e} e(t)dt\right| \leq \int_{t_k^e}^{t_{k+1}^e} |e(t)|dt \leq c(t_{k+1}^e - t_k^e). \tag{4.9}$$

Given that $t_{k+1}^e - t_k^e \leq \frac{C\delta}{b-c}$, it follows that

$$|\varepsilon_i| \leq \frac{c \cdot C\delta}{b-c}. \tag{4.10}$$

Approximation error bound for the derivative term

The derivative term at time s_{k+1} is given by

$$D_s(s_{k+1}) = \frac{e(s_{k+1}) - e(s_k)}{s_{k+1} - s_k}. \tag{4.11}$$

According to Lagrange Mean Value Theorem, there exists a point $\zeta \in [s_k, s_{k+1}]$ that

$$e'(\zeta) = \frac{e(s_{k+1}) - e(s_k)}{s_{k+1} - s_k}. \tag{4.12}$$

Assuming $|e(t)| \leq c$ and the bandwidth of $e(t)$ is Ω, it follows that

$$|e'(t)| \leq 2\pi\Omega c \tag{4.13}$$

such that for every interval $[s_k, s_{k+1}]$

$$\begin{aligned} |\varepsilon_d| &= |e'(t) - D_s(s_{k+1})| \\ &= |e'(t) - e'(\zeta)| \leq |e'(t)| + |e'(\zeta)| \leq 4\pi\Omega c. \end{aligned} \tag{4.14}$$

which means that $|\varepsilon_d| \leq 4\pi\Omega c$ for any t>0.

4.5. Numerical simulation

This section demonstrates the effectiveness of the spike-based network control system implemented using the proposed spike-based PID controller. As Fig. 4.2 shows, the error signal $e(t)$ and the feedback signal $y(t)$ are encoded and decoded.

Consider a second-order system described by the following transfer function

$$G(s) = \frac{Y(s)}{U(s)} = \frac{3}{s^2 + 3s - 2}. \tag{4.15}$$

The PID controller is given by

$$C(s) = k_p \cdot e(t) + k_i \cdot \int_0^t e(t)dt + k_d \cdot \frac{de(t)}{dt}. \tag{4.16}$$

The aim is to design a spike-based PID control system that achieves the following performance specifications:

- Overshoot: 0%.
- Settling time: 1.5 seconds.
- Steady state error: 0.

Here, the tuning approach for the PID controller is employed according to book [26]. k_p is firstly set low, with k_i and k_d set to zero. Then k_p is increased until there is overshoot. k_i and k_d are tuned in order to reach desired performance. The tuning procedure is shown in Fig. 4.3.

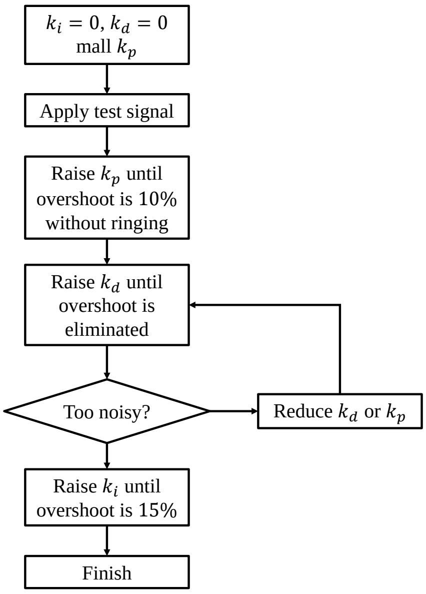

$k_i = 0, k_d = 0$
mall k_p
Apply test signal
Raise k_p until overshoot is 10% without ringing
Raise k_d until overshoot is eliminated
Too noisy?
Reduce k_d or k_p
Raise k_i until overshoot is 15%
Finish

Fig. 4.3 (redrawn from [26]). Tuning approach for the PID controller.

The simulation is carried using MATLAB. The closed-loop system is simulated for 100 seconds, with a simulation step length of 0.0001 second, i.e., there are 1,000,001 data points in the simulation.

The gains of the standard PID controller are tuned according to the approach proposed in [26], with $k_p = 7$, $k_i = 1.7$ and $k_d = 1.8$. The spike-based PID controller employs the same controller gains as the standard controller.

The simulation settings are shown in the following table.

Table 4.1. Simulation parameters.

Parameter	Explanation	Value
$\boldsymbol{T_{sim}}$	Overall simulation time	100 seconds
$\boldsymbol{dt}$	Sampling period	1×10^{-4} seconds
$\boldsymbol{\delta}$	IF neuron threshold	0.1
$\boldsymbol{b}$	IF neuron bias	2
$\boldsymbol{C}$	IF integration constant	1

The error signal $e(t)$ and system output signal $y(t)$ are encoded using single-channel IF-TEMs with same parameters. These encoded signals are decoded in real-time using the fast decoding algorithm proposed in Chapter 3.

4.5.1. Step response

Here is the simulation of the closed-loop system where the reference is a unit step signal with the step time of $t = 0s$.

The reference signal and final closed-loop system outputs are shown in Fig. 4.4.

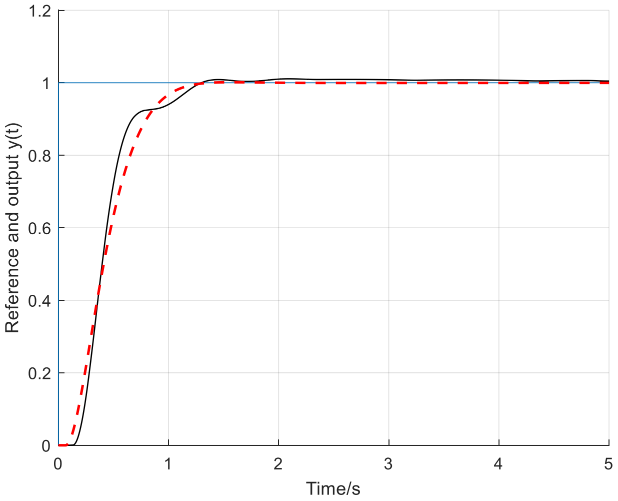


Fig. 4.4. Standard closed-loop system output (red) and spike-based closed-loop system output (black). The settling time of the standard system is 1.07 seconds and the settling time of the spike-based system is 1.19 seconds.

The figure shows that the spike-based system starts responding later than the standard system, since the TEM for error signal $e(t)$ has not output its second spike and the spike-based PID controller requires at least two spikes to propel. Also, the spike-based system has faster responding speed at the beginning, while the standard system's output is smoother after 0.6 second. The settling times of two systems are close and both systems do not have steady-state error, which indicates that the spike-based system has the same tracking speed as the standard system.

The outputs of the conventional and spike-based PID controllers are shown in Fig. 4.5.

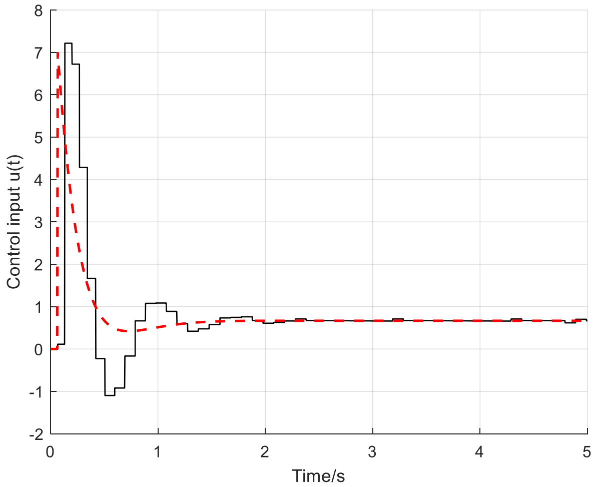


Fig. 4.5. The outputs of conventional (red) and spike-based (black) PID controllers.

The spike-based PID controller has a more oscillatory control input. This is because the spike-based control input is held between estimation points and is not updated continuously, which leads to additional control error.

Fig. 4.6 shows output errors of the conventional and spike-based PID control systems.

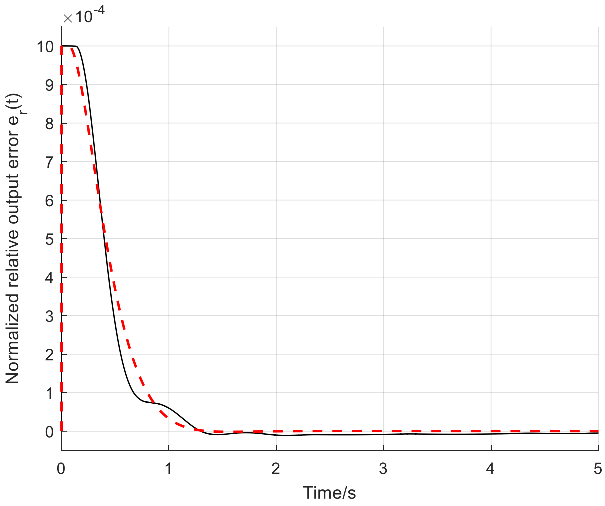


Fig. 4.6. Conventional closed-loop system output error (red) and spike-based closed-loop system output error (black). The normalized output error is given by $e_r(t) = \frac{r(t)-y(t)}{\|r(t)\|_2}$.

The errors of both spike-based and standard system converge to zero and the two systems have close convergency speed.

Here is the difference between the standard closed-loop system and the spike-based closed-loop system with same PID gains. The difference is shown in Fig. 4.7.

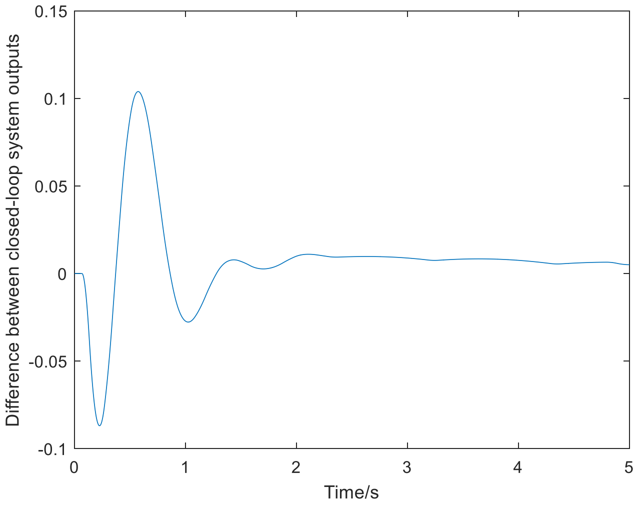


Fig. 4.7. Difference between the outputs of the conventional system and the spike-based control system.

These two systems have a larger difference from 0 to 1 second. The difference is small compared with the reference signal and decreases along the time.

4.5.2. Reference tracking

The spike-based network control was evaluated on a reference tracking control problem where the input reference $r(t)$ is a bandlimited signal that is a superposition of 10 sinusoidal signals with uniformly distributed amplitude and frequency in the range of $[-0.1,0.1]$ and $(0,2]$ (rad/s) respectively. The same controllers defined in Section 4.5.1 are employed here.

The reference tracking performance of the conventional and spike-based control systems are illustrated in Fig. 4.8.

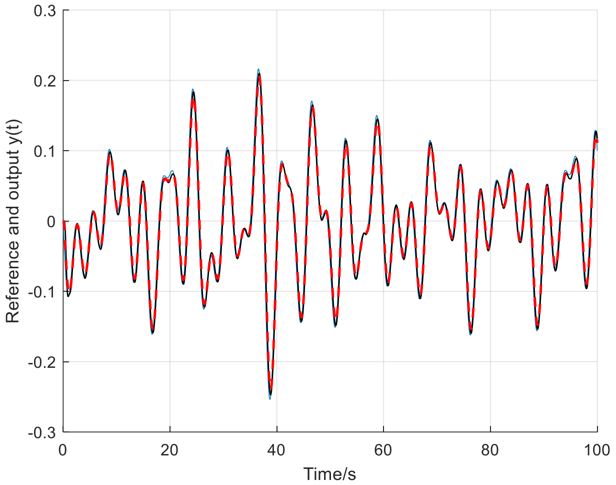


Fig. 4.8. Reference (blue), standard closed-loop system output (red) and spike-based closed-loop system output (black).

It is shown that both systems can track the reference very well and they have high responding speeds. To this respect, a spike-based closed-loop system can be applied to replace a standard system.

The standard PID control input and spike-based PID control inputs are shown in Fig. 4.9.

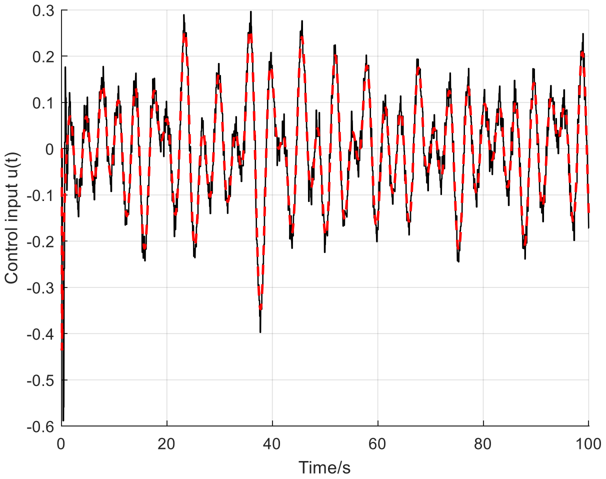


Fig. 4.9. Standard closed-loop system PID control input (red) and spike-based closed-loop system PID control input (black).

The control input of the spike-based PID controller is close to that of the standard controller. The spike-based control input has some jitter, yet it is still at the same level as the standard control input, which will not significantly increase the challenge of an actuator.

Fig. 4.10 shows the normalized output error of the conventional and spike-based PID control systems.

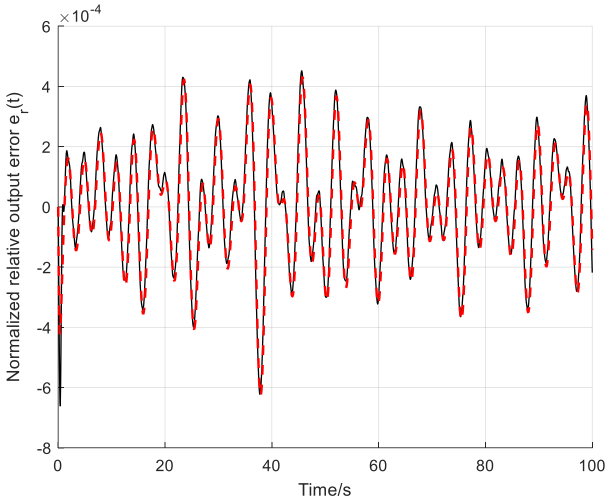


Fig. 4.10. Standard closed-loop system output error (red) and spike-based closed-loop system output error (black). The Mean Squared Error of the standard closed-loop system is 3.55×10^{-8}.

The Mean Squared Error of the spike-based closed-loop system is 3.66×10^{-8}, which is close to that of the standard system. This indicates that although the spike-based control input signal is slightly different from the standard control input signal, the spike-based process can track the reference well and model the standard system.

The difference between the outputs of the conventional and the spike-based closed-loop system with identical PID gains is shown in Fig. 4.11.

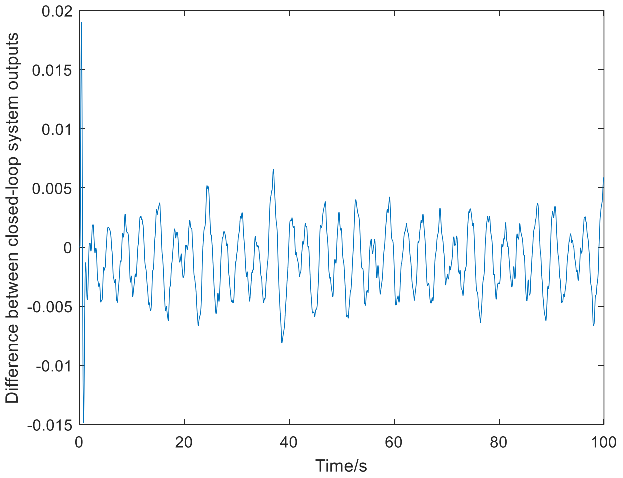


Fig. 4.11. Difference between the outputs of the conventional and the spike-based PID control systems.

The difference is large at the beginning, since the spike-based controller requires at least two spikes to be propelled. The result indicates that the spike-based system can model the standard system at a high accuracy.

4.5.3. Noise rejection

In order to assess the robustness of the spike-based PID control systems, simulations are conducted where both measurement noise affecting the output signal $y(t)$ and the spike times $\{t_k\}$ are taken into account.

Measurement noise added to system output

The measured output is given by

$$\hat{y}(t) = y(t) + \xi(t) \tag{4.17}$$

where ξ is a Gaussian white noise process with zero mean and standard deviation of 0.01. The Signal to Noise Ratio is approximately 20.4 dB. The resulting noisy output signal is shown in Fig. 4.12.

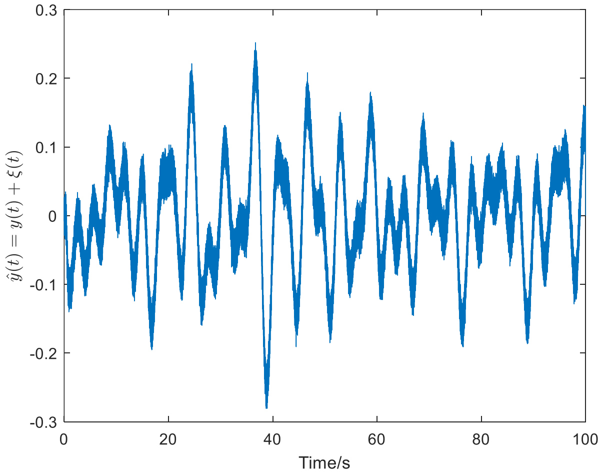


Fig. 4.12. Noised feedback signal $\hat{y}(t)$.

Fig. 4.13 provides a comparative illustration of the system outputs for both spike-based PID control systems, one with the presence of measurement noise and the other without it.

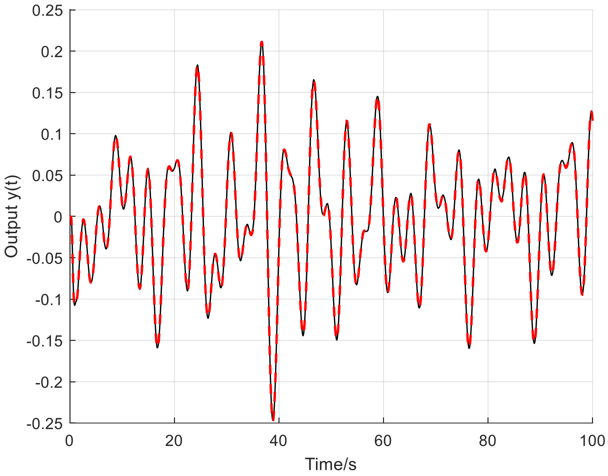


Fig. 4.13. Spike-based control system output (black) without senser noise, and the system output (red) with noise.

It is shown that the final output of the system with sensor noise is significantly close to that of the system without noise. Therefore, it can be concluded that the noise has very limited effect on the system's output.

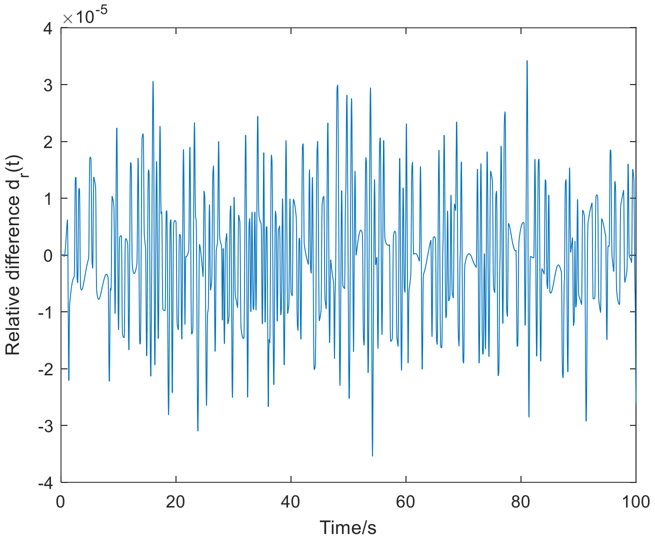


Fig. 4.14. Relative difference $d_r(t)$ between the outputs of the spike-based system and the system with sensor noise. The relative different $d_r(t)$ is defined as $d_r(t) = \frac{\hat{y}(t)-y(t)}{\|y(t)\|_2}$, where $\hat{y}(t)$ is the output of the spike-based system with sensor noise, $y(t)$ is the output of the spike-based system without sensor noise.

It is shown that the difference between the outputs of two systems is very small. This occurs because the integrate-and-fire time encoding acts as a low-pass filter, thereby endowing the encoding process with robustness against noise.

Measurement noise added to spike-time sequence

The transmitted sequence of spike-times is given by

$$\hat{t}_k = t_k + \zeta_k, \quad k = 1,2,3, \dots \tag{4.18}$$

where ζ_k are Gaussian distributed with zero mean and standard deviation of 0.001. The Signal to Noise Ratio is approximately 46 dB.

Fig. 4.15 provides a comparative illustration of the system outputs for both spike-based PID control systems, one with the presence of measurement noise and the other without it.

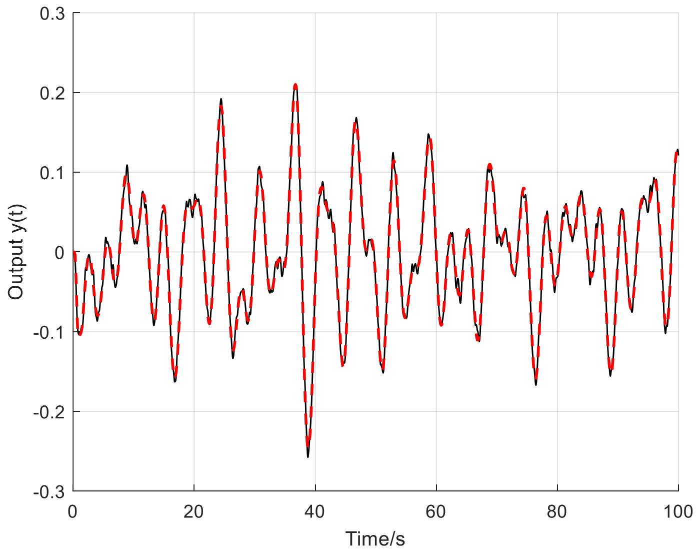


Fig. 4.15. Output of the spike-based PID in the case of noise-free (red) and noisy (black) spike timings.

The result shows that the system is affected by the noise. The spike timing noise causes the decoding error of the signals, which leads to the tracking error of the system. Nevertheless, the PID process is able to correct the tracking error and the noisy system can still track the reference.

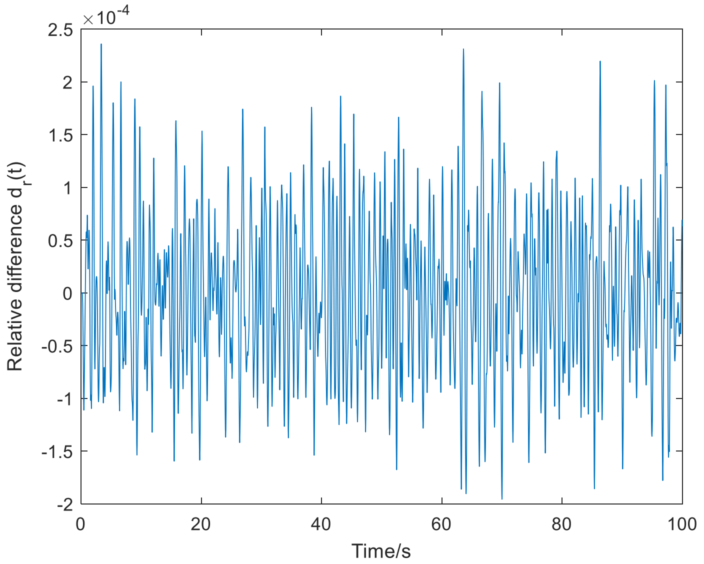


Fig. 4.16. Relative difference $d_r(t)$ between the outputs of the spike-based control system and the system with spike time jitter.

The figure shows that the difference between the noise-free and noisy systems are close to each other. Therefore, it can be concluded that the effect of the spike time jitter is limited.

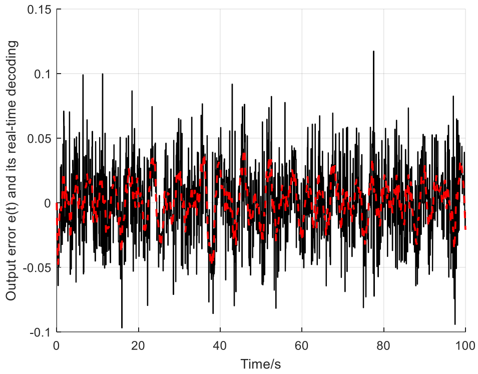


Fig. 4.17.a). System output error $e(t)$ and its decoding using the fast algorithm proposed in Chapter 2.

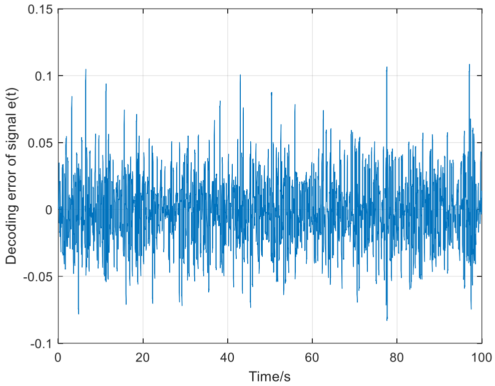


Fig. 4.17.b). Decoding error caused by the measurement noise of spike times.

It is shown that the spike timing noise has effect on the system output, which is caused by the decoding process. The noise of spike arriving times leads to the error of decoding. For example, in Fig. 4.16.a) and b), the system output error signal $e(t)$ is encoded using a TEM and decoded using the fast algorithm. The decoding algorithm refers to the spike times, which is noisy due to the measurement noise. The resulting decoding error furtherly leads to the error of data transmission. Nevertheless, the system is robust to the spike timing noise. Here, the ratio between the relative difference norm and the relative decoding error norm is calculated as follows:

The relative difference is defined as

$$d_r(t) = \frac{\hat{y}(t) - y(t)}{\|y(t)\|_2}, \tag{4.19}$$

where $\hat{y}(t)$ is the output of the spike-based system with spike timing noise, $y(t)$ is the output of the spike-based system without spike timing noise. The relative difference $d_r(t)$ indicates the effect of the spike timing noise on the system output.

The relative decoding error of is defined as

$$e_e = \frac{\hat{e}(t) - e(t)}{\|e(t)\|_2}, \quad k = 1,2,3, \dots \tag{4.20}$$

The relative decoding error e_e indicates the effect of the spike timing noise on the decoded error signal $e(t)$.

The ratio between the relative difference norm and the relative decoding error norm is calculated as

$$ratio_{d_r, e_e} = \frac{\|d_r(t)\|_2}{\|e_e\|_2}. \tag{4.21}$$

In this simulation, the ratio is $ratio_{d_r, e_e} = 0.035$, which shows that the effect of the spike timing noise on the system output is small compared with the effect of the spike timing noise on the decoded error signal $e(t)$. This indicates that although the spike timing noise leads to decoding error, the system is still robust to this noise. This because the PID control system tracks the reference and corrects the decoding error.

4.6. Conclusion

This chapter introduced a spike-based networked control system that uses time encoding to transmit the control system output and the control error that is used by a spike-based PID to compute the control actions. The spike-based network control system has several advantages compared with a conventional system including energy efficiency, robustness to noise, sparse-representation of signals and simpler devices for encoding and decoding of signals.

A key element of the network control system is a novel spike-based PID control law that relies on real-time decoding algorithm introduced in Chapter 3. The PID control terms were formulated in terms of the spike times corresponding to the encoded error signal and uniform error bounds were derived for each control term. A methodology for tuning the gains of the PID controller was introduced and demonstrated through numerical simulations. It has shown that the spike-based control system met the performance specification and that the error between the control system outputs corresponding to the conventional and spike-based implementation was very small.

The robustness of the spike-based control system to measurement errors, in respect to the system output and the timing of the transmitted spike code, was also investigated using numerical simulations. It has shown that the spike-based system exhibited very good measurement noise rejection properties due to the integrate-and-fire TEM acted as a low-pass filter and it minimized the effects of the system output measurement noise. Separately it has shown that the control system was also quite robust to spike time errors i.e., spike-time jitter.

Chapter 5
Spiking signal processing using identified models

5.1. Introduction

Time encoding of continuous-time signals using Time Encoding Machines, offers an alternative sampling paradigm that eliminates the need for a global clock to synchronise sampling the continuous signals and processing the resulting discrete-time signals. When the sequence of non-uniform time instants obtained using an IF-TEM corresponds to a spike train, the natural way to process the spike train is using Spiking Neural Networks and neuromorphic computing architectures [27].

However, the discrete sequence of inter-spike intervals $\{t_{k+1} - t_k\}_{k\in\mathbf{Z}^+}$ can be treated as an ordinary, non-uniformly sampled time series, which offers the opportunity to implement asynchronous signal processing of the non-uniform time series. A major advantage is that asynchronous circuits do not need to wait for clock signals and being event-driven. They do not consume energy unless there is data to process. Compared with conventional, non-uniform sampling, which generate both amplitude and timing information, time encoding using TEMs only generate time samples, with the inter-spike intervals carrying all the information needed to reconstruct the original signal.

Implementing signal processing operations on the time encoded signals requires designing suitable 'filters' that replicate the characteristics of conventional filters, that is the filtered time encoded signal after the reconstruction should approximate as closely as possible the continuous-time signal filtered using a conventional analogue filter.

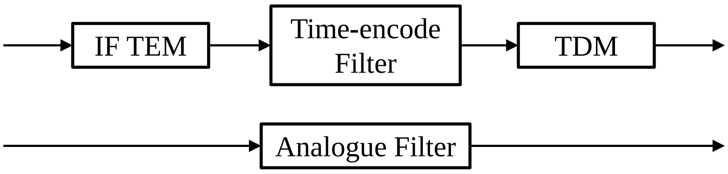


Fig. 5.1. Data transmission of the time-encode filter and the analogue filter.

This chapter introduces a system identification framework for designing asynchronous filters for processing time encoded signals generated using IF-TEMs.

The idea is to map the analog linear or nonlinear filter applied to a continuous-time signal onto equivalent time-encode filter that processes the discrete time series of spike-time intervals, generated by an IF-TEM, using system identification. Specifically, the time-code filters are identified based on input and output spike train intervals $\{t_k^u - t_{k-1}^u\}$ and $\{t_k^y - t_{k-1}^y\}$ respectively, obtained by encoding a predefined input $u(t)$ and the response $y(t)$ of the analogue filter to this input, using an IF-TEM.

The identified time-code filters can be used as building blocks, to implement signal processing operations of time encoded signals in the same way basic analog and digital signal processing blocks are used to process continuous- or uniformly-sampled discrete-time signals.

This chapter is organized as follows. Section 5.2 introduces an alternative formulation of the non-uniform time encoding of a signal $u(t)$ using an IF-TEM as the uniform sampling operation of an auxiliary function. Section 5.3 introduces the proposed system identification framework. Section 5.4 presents the case study of the identification and simulations of encoded-input-encoded-output models. Section 5.5 introduces the causality problem introduced by the model and provides the solution. Section

5.6 draws the conclusions.

5.2. IF-TEM encoding as a uniform sampling operation

According to Chapter 3, the time encoded signal can be recovered to a continuous signal in real-time using the fast algorithm. Therefore, the encoded signal $\{t_k\}_{k\in\mathbf{Z}^+}$ can be utilized to transmit data in a real-time system. The data transmission method using original continuous signal measures a two-dimensional signal: time t and signal value $u(t)$. While the method using decoded signal $\{t_k\}_{k\in\mathbf{Z}^+}$ measures a signal with only dimension: spike interval $\{t_{k+1} - t_k\}_{k\in\mathbf{Z}^+}$. Also, the method using decoded signal has a lower energy cost, which has been proved in the previous chapter.

The IF time encoding algorithm integrates the original signal $u(t)$ and outputs a spike each time the integral reaches the threshold [3]. This is a nonuniform sampling in time, since the TEM does not output spikes isochronally. However, according to [16], if the axis is reversed, the time encoding process can be treated as a uniform sampling of integral (see Fig. 5.2). Let a function $y = f(t)$ be:

$$y = f(t) = \frac{1}{C}\int_0^t (u(\tau) + b)d\tau, \tag{5.1}$$

then $y(t)$ is the integral of the signal $u(t)$. The IF time encoding algorithm samples $y(t)$ nonuniformly when $y(t) = k \cdot \delta,\ k = 1,2, \ldots.$ However, if the axis is reversed as Fig. 5.2 shows, it follows that the sampling of the inverse function is uniform.

The inverse function $t = f^{-1}(y)$ is sampled in value y, and the sampling step interval is threshold δ. At each sampling step, the TEM outputs a spike and the sampling value is the spike interval, i.e., the difference between the two consecutive spike arriving times.

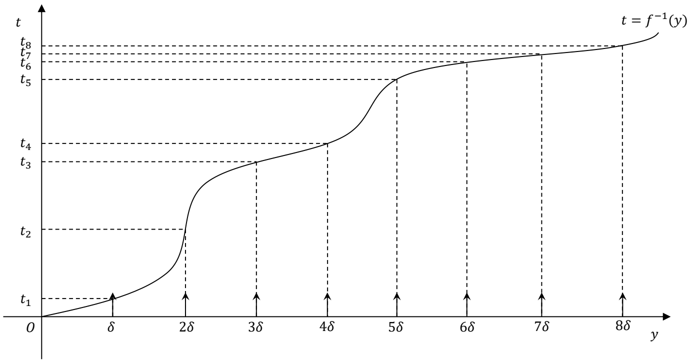


Fig. 5.2. Inverse uniform sampling of the integral.

According to Theorem 1 of [16]:

Let $y(t) \triangleq \int_0^t (u(\tau) + b)d\tau$. It follows that y has an inverse ψ such that

$$L_k^D \bar{\psi}' = \bar{q}_k, \forall k \in Z, \tag{5.2}$$

where $L_k^D \bar{\psi}' = \int_{k\bar{\delta}}^{(k+1)\bar{\delta}} (\bar{\psi}'(y))dy,\ \bar{\psi}(y) \triangleq \psi(y) - \frac{y}{b},\ D = \{k\bar{\delta}\}_{k\in Z},\ \bar{\delta} = C \cdot \delta$

and $\bar{q}_k \triangleq (t_{k+1} - t_k) - \frac{\bar{\delta}}{b}$.

Therefore, the encoded spike interval $t_{k+1} - t_k$ can be expressed as

$$\begin{aligned} t_{k+1} - t_k &= L_k^D \bar{\psi}' + \frac{C\delta}{b} \\ &= \int_{k\bar{\delta}}^{(k+1)\bar{\delta}} \psi'(y)dy \\ &= \psi\left((k+1)\bar{\delta}\right) - \psi(k\bar{\delta}), \qquad k = 1,2,3 \ldots \end{aligned} \tag{5.3}$$

By applying the fast reconstruction algorithm, the sampled signal, i.e., spike intervals $\{t_{k+1} - t_k\}_{k\in\mathbf{Z}^+}$ can be recovered to continuous signal in real-time. Therefore, the encoded spike train $\{t_k\}_{k\in\mathbf{Z}^+}$ can transmit data in the format of spike intervals $\{t_{k+1} - t_k\}_{k\in\mathbf{Z}^+}$. A real-time spike-based system can be realized if a filter which responds to the spike intervals and also has spiking outputs is designed.

5.2.1. Time encoding of Linear Time Invariant dynamical systems

A linear time invariant (LTI) system is a system that is both linear and time-invariant [28] where the response $y(t)$ to an arbitrary input $u(t)$ can be given directly using convolution [29].

Define $h(t)$ as the impulse response of this system. The impulse response of the system is the output of the system when respond to an impulse signal input $\delta(t)$. A unit impulse signal at $t = t_0$ is defined as

$$\delta(t - t_0) = \begin{cases} +\infty, & t = t_0 \\ 0, & t \neq t_0 \end{cases}$$

$$\text{and } \int_{-\infty}^{+\infty} \delta(t - t_0)dt = 1. \tag{5.4}$$

Then the response $y(t)$ to an arbitrary input $u(t)$ of a LTI system is given by

$$y(t) = (u * h)(t), \tag{5.5}$$

where $*$ is the convolution operation and $h(t)$ is the impulse response of this system.

The response $y(t)$ to input $u(t)$ is given by

$$y(t) = (u * h)(t) = \int_{-\infty}^{+\infty} u(\tau)h(t-\tau)d\tau = \int_{-\infty}^{+\infty} u(t-\tau)h(\tau)d\tau. \tag{5.6}$$

Here, the input signal $u(t)$ and output signal $y(t)$ of this LTI system are encoded using an IF-TEM into time sequences $\{t_k^u\}_{k\in\mathbf{Z}^+}$ and $\{t_k^y\}_{k\in\mathbf{Z}^+}$, respectively.

The biased integrals of $u(t)$ and $y(t)$ are defined as

$$v(t) \triangleq \int_0^t (u(\tau) + b)d\tau, \tag{5.7}$$

$$w(t) \triangleq \int_0^t (y(\tau) + b)d\tau. \tag{5.8}$$

According to (5.3), it follows that

$$t_{k+1}^u - t_k^u = \psi^u\left((k+1)\bar{\delta}\right) - \psi^u(k\bar{\delta}), \tag{5.9}$$

$$t_{k+1}^y - t_k^y = \psi^y\left((k+1)\bar{\delta}\right) - \psi^y(k\bar{\delta}), \tag{5.10}$$

where $k = 1,2,3 \ldots$, $\psi^u(v)$ is the inverse of $v(t)$ and $\psi^y(w)$ is the inverse of $w(t)$, which are the encoded input and output signal, respectively.

Since ψ^y is the inverse of $w(t)$, it gives that

$$\psi^y(k\bar{\delta}) = t_k^y. \tag{5.11}$$

Therefore, the encoded output is given by

$$\psi^y(k\bar{\delta}) = inv(w)(t_k^y)$$

$$
\begin{aligned}
&= inv\left(\int_0^{t_k^y}(y(t)+b)dt\right)\\
&= inv\left(\int_0^{t_k^y}\big((x*h)(t)+b\big)dt\right)\\
&= inv\left(\int_0^{t_k^y}\left(\int_{-\infty}^{+\infty}u(\tau)h(t-\tau)d\tau+b\right)dt\right).
\end{aligned}
\tag{5.12}
$$

Also, notice that ψ^u is the inverse of $v(t)$, it holds that

$$(\psi^u)' = \frac{1}{v'} = \frac{1}{u(t)+b}. \tag{5.13}$$

Therefore

$$u(t) = \frac{1}{(\psi^u(v(t)))'} - b. \tag{5.14}$$

It follows that

$$
\begin{aligned}
\psi^y(k\delta) &= inv\left(\int_0^{t_k^y}\left(\int_{-\infty}^{+\infty}u(\tau)h(t-\tau)d\tau+b\right)dt\right)\\
&= inv\left(\int_0^{t_k^y}\left(\int_{-\infty}^{+\infty}\left(\frac{1}{(\psi^u(v(\tau)))'}-b\right)h(t-\tau)d\tau+b\right)dt\right),
\end{aligned}
\tag{5.15}
$$

where $inv(f)$ is the inverse function of f. It also holds that

$$(\psi^y)' = 1/\left(\int_{-\infty}^{+\infty}\left(\frac{1}{(\psi^u(v(\tau)))'}-b\right)h(t-\tau)d\tau+b\right). \tag{5.16}$$

These indicate the function of the encoded input ψ^u and encoded output ψ^y.

5.2.2. Time encoding of Nonlinear Time Invariant systems

Define a general nonlinear system:

$$\frac{dy}{dt} = f(y,u), \tag{5.17}$$

where $f(y,u)$ is a nonlinear combination of $y(t)$ and $u(t)$.

According to [51], the nonlinear system can be described using a Volterra Series:

$$y(t) = \boldsymbol{N}u(t) = h_0 + \sum_{n=1}^{\infty}\int\cdots\int h_n(\tau_1,\dots,\tau_n)\cdot u(t-\tau_1)\cdots u(t-\tau_n)d\tau_1\cdots d\tau_n, \tag{5.18}$$

where

$$\int_0^{\infty}\cdots\int_0^{\infty}|h_n(\tau_1,\dots,\tau_n)|d\tau_1\cdots d\tau_n < \infty. \tag{5.19}$$

Here, the input signal $u(t)$ and output signal $y(t)$ of the nonlinear system are encoded using an IF-TEM into time sequences $\{t_k^u\}_{k\in\boldsymbol{Z}^+}$ and $\{t_k^y\}_{k\in\boldsymbol{Z}^+}$, respectively.

The biased integrals $v(t)$ and $w(t)$, as well as the inverse functions ψ^u and ψ^y are defined as in Section 5.2.1. Then the encoded output ψ^y is given by

$$
\begin{aligned}
\psi^y(k\delta) &= inv(w)(t_k^y)\\
&= inv\left(\int_0^{t_k^y}(y(t)+b)dt\right)
\end{aligned}
$$

$$= inv\left(\int_0^{t_k^y}(\boldsymbol{N}u(t)+b)dt\right)$$

$$= inv\left(\int_0^{t_k^y}\left(b+h_0 + \sum_{n=1}^{\infty}\int_0^{+\infty}\cdots\int_0^{+\infty}\int h_n(\tau_1,\dots,\tau_n)\cdot u(t-\tau_1)\cdots u(t-\tau_n)d\tau_1\cdots d\tau_n\right)dt\right). \tag{5.20}$$

Considering that $u(t)=\frac{1}{(\psi^u(v(t)))'}-b$, $\psi^y(k\delta)$ can be furtherly given by

$$\psi^y(k\delta) = inv\left(\int_0^{t_k^y}\left(b+h_0 + \sum_{n=1}^{\infty}\int_0^{+\infty}\cdots\int_0^{+\infty}\int h_n(\tau_1,\dots,\tau_n) \cdot\left(\frac{1}{(\psi^u(v(t-\tau_1)))'}-b\right)\cdots\left(\frac{1}{(\psi^u(v(t-\tau_n)))'}-b\right)d\tau_1\cdots d\tau_n\right)dt\right) \tag{5.21}$$

and

$$(\psi^y)' = 1/\left(b+h_0 + \sum_{n=1}^{\infty}\int_0^{+\infty}\cdots\int_0^{+\infty}\int h_n(\tau_1,\dots,\tau_n) \cdot\left(\frac{1}{(\psi^u(v(t-\tau_1)))'}-b\right)\cdots\left(\frac{1}{(\psi^u(v(t-\tau_n)))'}-b\right)d\tau_1\cdots d\tau_n\right). \tag{5.22}$$

These indicate the function of the encoded input ψ^u and encoded output ψ^y of the nonlinear system.

5.3. System identification methodology

The aim is to derive a model for the time encoded output of the dynamical system as a function of the time encoded input using system identification.

The general encoded-input-encoded-output (EIEO) model is given by

$$Y(k) = F\left[Y(k-1),Y(k-2),\dots,Y\left(k-n_y\right),U(k-d),U(k-d-1),\dots,U(k-d-n_u), e(k-1),e(k-2),\dots,e(k-n_e)\right]+e(k), \tag{5.23}$$

where $Y(k)=t_k^y-t_{k-1}^y$, $U(k)=t_k^u-t_{k-1}^u$ and $e(k)$ is the noise sequence [30].

The model is identified using encoded input and output spike times obtained by simulation of the dynamical system of interest.

Let $u(t)$ and $y(t)$ be the input and output of a continuous real-time system G, respectively. $u(t)$ and $y(t)$ are encoded using IF-TEMs (TEM settings for $u(t)$ and $y(t)$ are same), and the TEMs output two spike trains $\{t_k^u\}_{k\in\mathbf{Z}^+}$ and $\{t_k^y\}_{k\in\mathbf{Z}^+}$ in real-time. Let $U(k)_{k\in\mathbf{Z}^+}$ and $Y(k)_{k\in\mathbf{Z}^+}$ be the sampled spike time intervals of $\{t_k^u\}_{k\in\mathbf{Z}^+}$ and $\{t_k^y\}_{k\in\mathbf{Z}^+}$, respectively.

Fig. 5.3 shows the identification architecture.

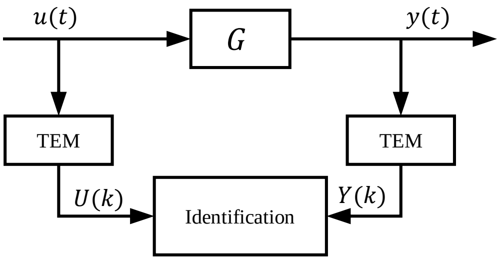


Fig. 5.3. Block diagram of the identification architecture.

As shown in Fig. 5.3, the encoded input and output $U(k)$ and $Y(k)$ are utilized to identify the model of (5.23).

The identified EIEO model can be used to implement filtering operations on the encoded signals.

As shown in Fig. 5.4, the original input signal $u(t)$ is firstly encoded to a spike train $\{t_k^u\}_{k\in\mathbf{Z}^+}$ that is processed using the identified EIEO. The filter output $\tilde{Y}(k)$ is decoded to a continuous signal $\tilde{y}(t)$ using the fast reconstruction algorithm in real-time. The continuous signal $\tilde{y}(t)$ should model the standard $y(t)$. Therefore, the encoded signal $Y(k)$ is utilized for the identification as shown in Fig. 5.3.

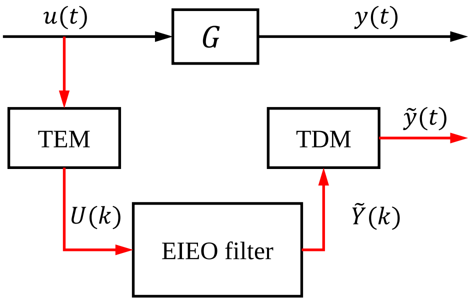


Fig. 5.4. Block diagram of the filtering architecture.

5.3.1. System identification using Hammerstein-Wiener models

A Hammerstein model has a nonlinear block preceding a linear block [31], while a Wiener model has a nonlinear block following a linear block [32]. A Hammerstein-Wiener model is a combination of a Hammerstein model and a Wiener model, which has nonlinear blocks preceding and following a linear model [33]. The block diagram of a Hammerstein-Wiener model is shown below.

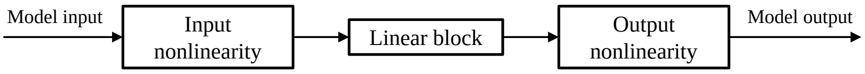


Fig. 5.5. Structure of a Hammerstein-Wiener model.

The transfer function [31] of the discrete linear block is given by

$$\frac{N(z)}{D(z)} = \frac{\alpha_0+\alpha_1 z^{-1}+\cdots+\alpha_{n-1} z^{-(n-1)}}{1+\beta_1 z^{-1}+\cdots+\beta_n z^{-n}}. \tag{5.24}$$

α_i and β_j are parameters to be approximated. n is selected manually.

The input and output nonlinear blocks can be estimated using one-dimensional polynomials [31]:

$$y(t) = \gamma_1 x(t) + \gamma_2 x^2(t) + \cdots + \gamma_m x^m(t), \tag{5.25}$$

where $x(t)$ and $y(t)$ are the input and output of the nonlinear block, respectively. The model is estimated by approximating the parameters $\gamma_1, \gamma_2, \dots, \gamma_m$. The order of the nonlinearity m is selected manually.

Also, the input and output nonlinearities can be represented using piece-wise linear functions [34]. A piece-wise linear function has break points $p_1, p_2, \dots, p_n$. The function is linear between break points, i.e., a piece-wise linear function is a connection of a series of linear functions. The model is estimated by approximating the locations of the break points [34].

5.3.2. Data generation and model estimation

In this chapter, a random band-limited input signal $u(t)$ is generated for the system. $u(t)$ is generated using a uniformly distributed white noise signal, which is filtered using a 10-order band-pass filter. The random input signal $u(t)$ and the system output $y(t)$ are encoded using IF-TEMs with same parameters. The resulting encoded input and output signals $U(k)$ and $Y(k)$ are utilized as the estimation data for identification.

The linear block of the Hammerstein-Wiener model is structured as (5.24). The input and output nonlinearities are selected as either one-polynomials or piece-wise linear functions.

The identification is carried out by approximating the parameters in the linear block (5.24) and the parameters of the nonlinearities. According to Section 5.3.1, the following parameters are to be approximated:

- Parameters of the linear block: $\alpha_0, \alpha_1, \dots, \alpha_{n-1}$ and $\beta_1, \beta_2, \dots, \beta_n$ of (5.24).
- Parameters of the input and output nonlinear functions: $\gamma_1, \gamma_2, \dots, \gamma_m$ of (5.25) or the location of break points of the piece-wise linear function(s) $(a_1, b_1), (a_2, b_2), \dots, (a_p, b_p)$.

The approximation methodology can be selected as Gauss-Newton method [24, 25] or Gradient Decent method [35].

Gauss-Newton method

The Gauss-Newton method minimizes the squared sum of the residuals $\sum_{i=1}^{N} r_i^2$ by iteratively optimize the parameters [36]. The residual $\boldsymbol{r} = [r_1, r_2, \dots, r_N]^{\mathrm{T}}$ is defined as

$$r_i(\boldsymbol{\theta}) = y_i - f(x_i, \boldsymbol{\theta}), \tag{5.26}$$

where N is the number of data points, $\boldsymbol{\theta} = [\theta_1, \theta_2, \dots, \theta_m]^{\mathrm{T}}$ is the parameter vector, x_i is the ith system input signal, y_i is the ith system output signal and $f(\cdot)$ is the estimated model with parameters $\boldsymbol{\theta}$.

Then Gauss-Newton method is given iteratively by

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} + (\boldsymbol{P}^{\mathrm{T}}\boldsymbol{P})^{-1}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{r}\big(\boldsymbol{\theta}^{(n)}\big), \tag{5.27}$$

with an initial estimation $\boldsymbol{\theta}^{(0)}$.

$\boldsymbol{P}$ is given by

$$(\boldsymbol{P})_{ij} = -\frac{\partial r_i(\boldsymbol{\theta}^{(n)})}{\partial \theta_j} \tag{5.28}$$

and $(\boldsymbol{P}^{\mathrm{T}}\boldsymbol{P})^{-1}\boldsymbol{P}^{\mathrm{T}}$ denotes the pseudo-inverse of $\boldsymbol{P}$.

Gradient Decent method

Gradient Decent method minimizes an objective function which is in the form of a squared sum, by the approach of updating the parameters in the opposite direction of the gradient of the objective function [35]. The definition of the objective function $\boldsymbol{J}(\boldsymbol{\theta})$ varies with the mode of the algorithm, where $\boldsymbol{\theta} = [\theta_1, \theta_2, \dots, \theta_m]^{\mathrm{T}}$ is the set of parameters.

- For the ***Batch Gradient Decent method***, the parameters $\boldsymbol{\theta}$ are updated using the gradients of all the dataset $[(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)]$.

  The objective function $\boldsymbol{J}(\boldsymbol{\theta})$ is defined as

  $$\boldsymbol{J}(\boldsymbol{\theta}) = \sum_{i=1}^{N}\big(y_i - f(x_i, \boldsymbol{\theta})\big)^2, \tag{5.29}$$

  where $f(\cdot)$ is the estimated model with parameters $\boldsymbol{\theta}$.

- For the ***Stochastic Gradient Descent method***, the parameters $\boldsymbol{\theta}$ are updated using the gradients of each training data (x_i, y_i).

  The objective function $\boldsymbol{J}(\boldsymbol{\theta})$ is defined as

  $$\boldsymbol{J}(\boldsymbol{\theta}) = \big(y_i - f(x_i, \boldsymbol{\theta})\big)^2. \tag{5.30}$$

- For the ***Mini-Batch Gradient Descent method***, the parameters $\boldsymbol{\theta}$ are updated using the gradients of a range of the training data $\big[(x_{j+1}, y_{j+1}), (x_{j+2}, y_{j+2}), \dots, (x_{j+K}, y_{j+K})\big]$.

  The objective function $\boldsymbol{J}(\boldsymbol{\theta})$ is defined as

  $$\boldsymbol{J}(\boldsymbol{\theta}) = \sum_{i=j}^{j+K}\big(y_i - f(x_i, \boldsymbol{\theta})\big)^2. \tag{5.31}$$

With an initial estimation $\boldsymbol{\theta}^{(0)}$, the parameter set $\boldsymbol{\theta}$ is iteratively given by

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - \eta\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{\theta}^{(n)}}, \tag{5.32}$$

where η is a constant step length of approximation.

5.4. Numerical study

In this section, the numerical study of different EIEO models of linear and nonlinear filters is carried out. The simulation is carried out using MATLAB. The outputs of the EIEO models are spiking signals $Y(k)$. For the validation of the identified models, the EIEO model outputs $Y(k)$ are decoded using the fast algorithm and the decoded model outputs are compared with standard system outputs. This section also addresses a causality problem introduced by the EIEO model and presents a solution to this problem.

Herer are the identification of four different linear systems:

- A first-order system with one zero;
- A second-order system without zero;

- A second-order system with one zero;
- A second-order system with two zeroes.

The identification is carried using the encoded input $U(k)$ and encoded output $Y(k)$. The identification is to model the behaviour the EIEO model of the systems. Here, the validation of the EIEO model is presented, and the final real-time decoded output signals to show the performance of the identified model.

The MATLAB simulation is carried out using 1000001 data points, with 100 seconds of total simulation time and 0.0001 second of step length of simulation. The sampling time of the system block is 0.001 second.

The random input is generated using a uniformly distributed white noise signal, which is filtered using a 10-order band-pass filter. The passband frequency of the filter is $0.05{\sim}1.5$ rad/s. The input and output signals of the systems are encoded using IF-TEMs with same parameters.

Parameter settings of the IF-TEMs are shown in the following table.

Table 5.1. IF-TEM parameters.

Parameter	Value	Description
δ	0.75	IF neuron threshold
b	2	IF neuron bias
C	1	IF integration constant

The simulation results are shown below.

Example 1. First order Linear System

The transfer function of the system is

$$G(z) = \frac{\hat{y}(z)}{\hat{u}(z)} = \frac{6z-5.994}{z-0.99}, T_s = 0.001s, \tag{5.33}$$

where $\hat{u}(z)$ and $\hat{y}(z)$ stand for the z-transform of the input signal $u(t)$ and output signal $y(t)$ of the system.

The encoded input spike interval $U(k)$ is utilized as the model input, while the encoded output spike interval $Y(k)$ is utilized as the model output. The data for estimation and validation are shown in Fig. 5.6.

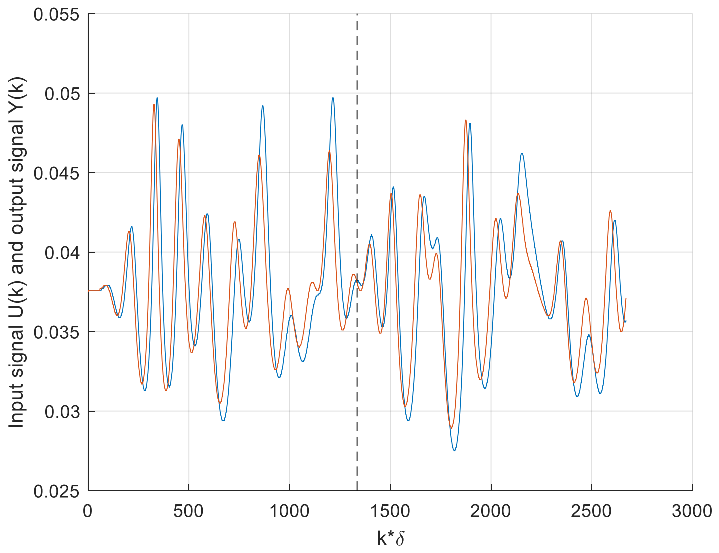


Fig. 5.6. Input signal $U(k)$ (blue) and output signal $Y(k)$ (red). The estimation data and validation data are divided by the dashed line.

The system is identified using Hammerstein-Wiener model.

The linear block of the model is

$$x_{out}(k) = \frac{B(z)}{F(z)} x_{in}(k) + e(k), \tag{5.34}$$

where

$$\begin{aligned} B(z) &= -1.161z^{-4} + z^{-5} + 1.155z^{-6} - 1.006z^{-7} \\ F(z) &= 1 - 1.064z^{-1} - 0.8479z^{-2} + 1.083z^{-3} - 0.1332z^{-4}, \end{aligned} \tag{5.35}$$

and $n_b = 7,\ n_f = 4,\ n_k = 1.$

The input nonlinearity is piecewise-linear with 8 break points and the output nonlinearity is piecewise-linear with 7 break points:

Break points of input nonlinearity	Break points of output nonlinearity
$(0.0317, -0.0071)$	$(-0.0120, 0.0765)$
$(0.0335, -0.0044)$	$(-0.0008, 0.0394)$
$(0.0354, -0.0021)$	$(0.0010, 0.0356)$
$(0.0380, 0.0004)$	$(0.0031, 0.0322)$
$(0.0404, 0.0022)$	$(0.0084, 0.0262)$
$(0.0426, 0.0036)$	$(0.0145, 0.0284)$
$(0.0447, 0.0047)$	$(0.0206, 0.0354)$
$(0.0479, 0.006)$	

The identified model of the system is a spike-based filter, which has spiking input and output. The input to the model is the encoded signal $\{t_k^u\}$ and the output of the model is $\{t_k^y\}$.

The identification results are shown in Fig. 5.7. The relative error is the error divided by the 2-norm of $Y(k)$.

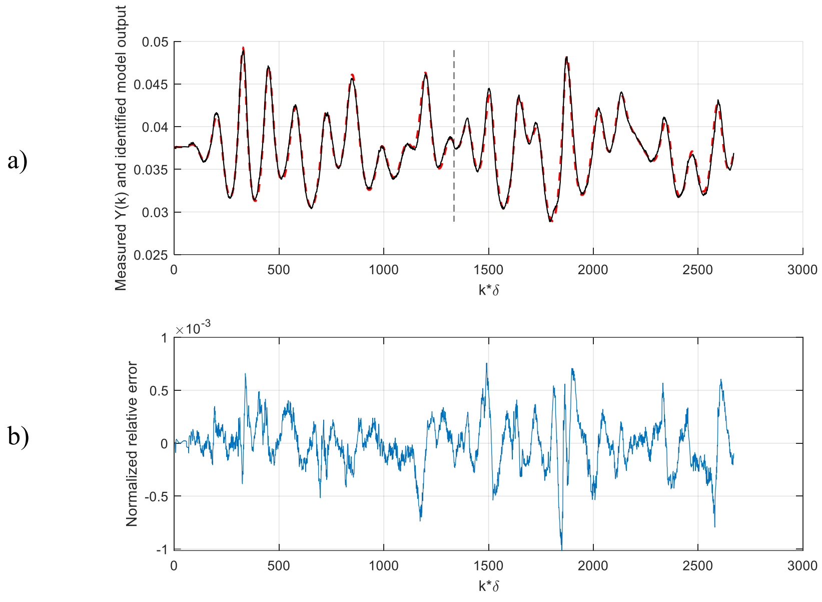


Fig. 5.7.a). Measured $Y(k)$ (red) and identified model output (black), where $\tilde{Y}(k)$ is the model output of the identified model. The estimation data and validation data are divided by the dashed line.

b). Relative error ($e_r = \frac{\tilde{Y}(k)-Y(k)}{\|Y(k)\|_2}$).

The identified model's output fits the validation data very well with a small relative error. The error is small compared with the level of $Y(k)$ and has a zero mean. Therefore, the identified model can be applied as an EIEO model.

Fig. 5.8 shows the decoded signals and error.

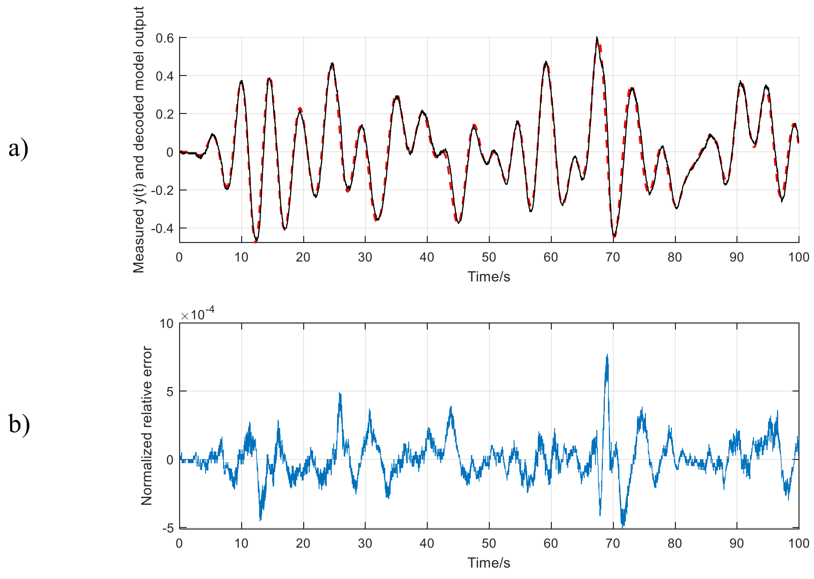


Fig. 5.8.a) Measured output $y(t)$ (red) and decoded system output (black).

b). Relative error. The relative error is defined as $e_r = \frac{\tilde{y}(t)-y(t)}{\|y(t)\|_2}$, where $\tilde{y}(t)$ is the decoded model output using the real-time fast algorithm proposed in Chapter 3.

Fig. 5.8 shows that the decoded signal is close to the measured signal. The error is small compared with the norm of the output. Therefore, the identification is effective and the identified model can be employed to replace the analogue filter.

Example 2: Second-Order System

The transfer function of the system is

$$G(z) = \frac{\hat{y}(z)}{\hat{u}(z)} = \frac{0.00003}{z^2 - 1.9405z + 0.9405}, T_s = 0.001s. \tag{5.36}$$

The encoded input spike interval $U(k)$ is utilized as the model input, while the encoded output spike interval $Y(k)$ is utilized as the model output. The data for estimation and validation are shown below.

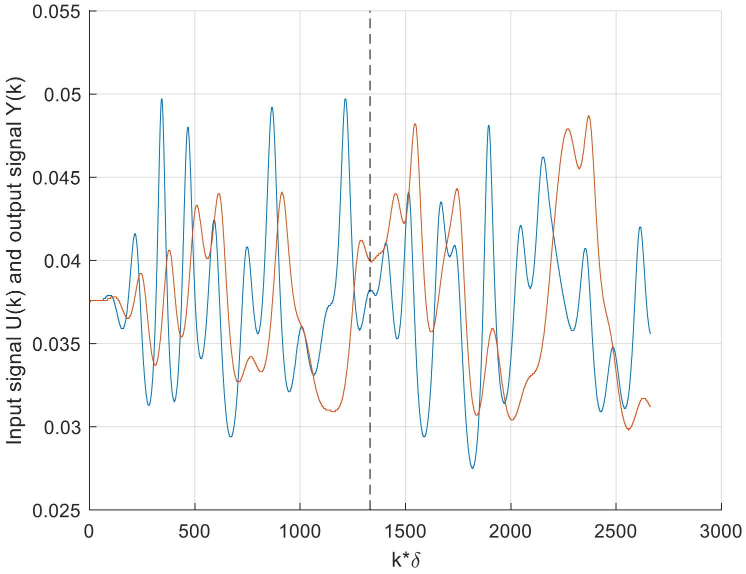


Fig. 5.9. Input signal $U(k)$ (blue) and output signal $Y(k)$ (red). The estimation data and validation data are divided by the dashed line.

The system is identified using Hammerstein-Wiener model.

The linear block of the model is

$$x_{out}(k) = \frac{B(z)}{F(z)} x_{in}(k) + e(k), \tag{5.37}$$

where

$$B(z) = z^{-2}$$
$$F(z) = 1 - z^{-1}, \tag{5.38}$$

and $n_b = 2,\ n_f = 1,\ n_k = 1$.

The input nonlinearity is one-dimensional polynomial of degree 1 and the output nonlinearity is piecewise-linear with 4 break points.

Input nonlinearity:

$$x_{out}(k) = 1.4878 x_{in}(k) - 0.0559 \tag{5.39}$$

Break points of output nonlinearity piecewise-linear:

Break points of output nonlinearity
$(-0.8742, 0.0297)$
$(-0.4746, 0.0326)$
$(-0.1612, 0.0353)$
$(0.2253, 0.0406)$

The identified model of the system is a spike-based filter, which has spiking input and output. The input to the model is the encoded signal $\{t_k^u\}$ and the output of the model is $\{t_k^y\}$.

The identification results are shown below. The relative error is the error divided by the 2-norm of $Y(k)$.

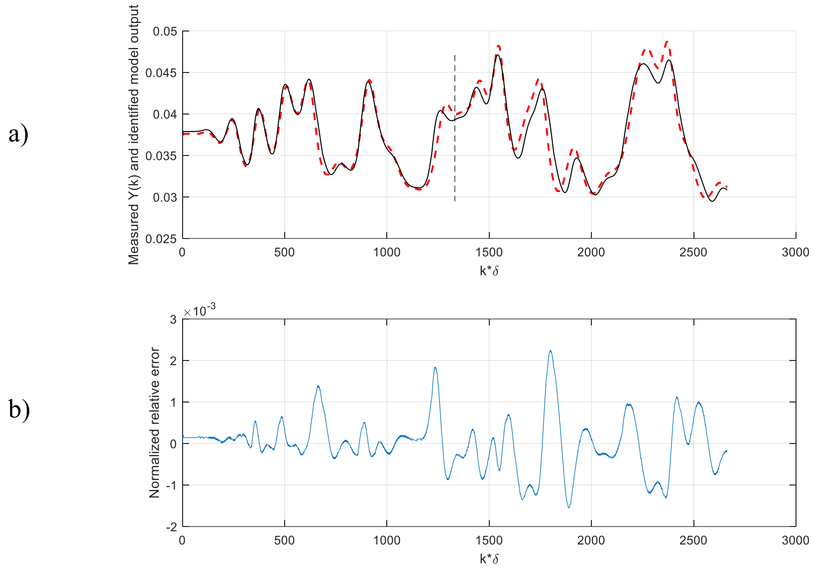


Fig. 5.10.a). Measured $Y(k)$ (red) and identified model output (black). The estimation data and validation data are divided by the dashed line.

b). Relative error of the identified model.

The identification has larger error than the previous first-order system case. The identified model fits the estimation data with a smaller error, but the error increases for validation. Notice that the overall error is still small compared with $Y(k)$.

Fig. 5.11 shows the decoded signals and error.

a)

b)

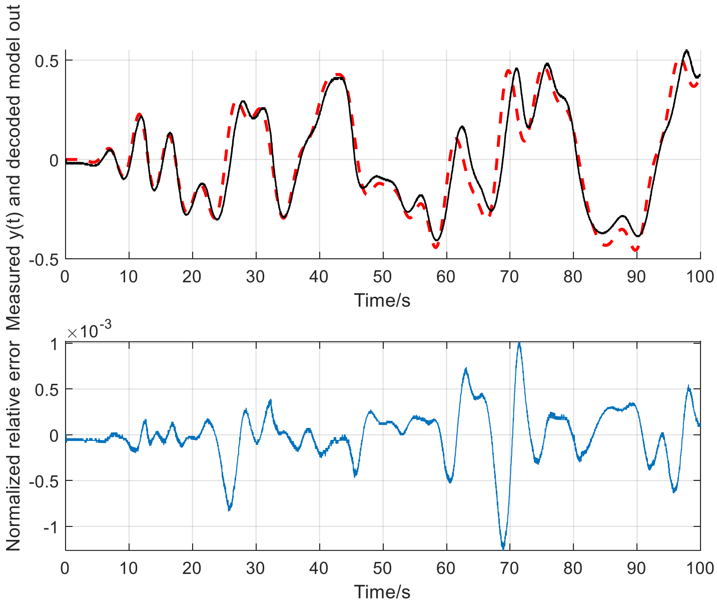


Fig. 5.11.a). Measured output $y(t)$ (red) and decoded system output (black).

b). Relative error.

It is shown in the figure that the decoded signal can model the measured system output. The error is large from 65 to 75 second. Despite this, the error is relatively small and has a zero mean.

Example 3: Second-order system with one zero

The transfer function of the system is

$$G(z)=\frac{\hat{y}(z)}{\hat{u}(z)}=\frac{3600z-3599.7}{z^2-1.872z+0.886}, T_s=0.001s, \tag{5.40}$$

The encoded input spike interval $U(k)$ is utilized as the model input, while the encoded output spike interval $Y(k)$ is utilized as the model output. The data for estimation and validation are shown below.

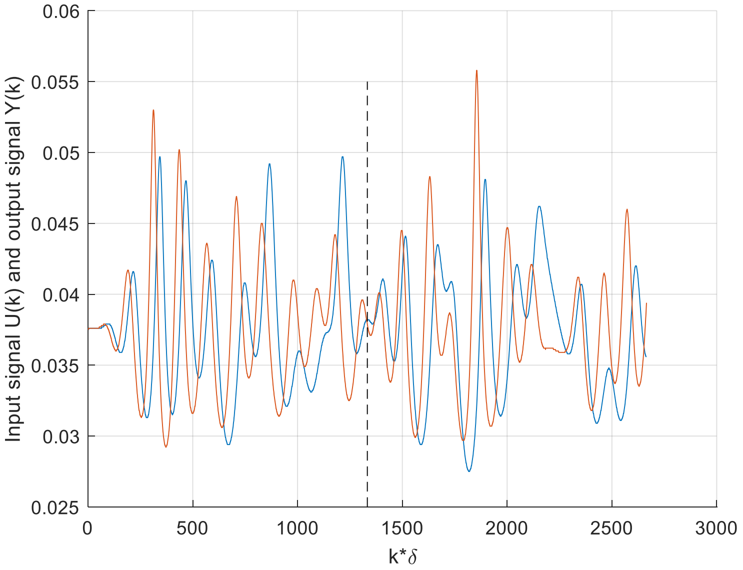


Fig. 5.12. Input signal $U(k)$ (blue) and output signal $Y(k)$ (red). The estimation data and validation data are

divided by the dashed line.

The system is identified using Hammerstein-Wiener model.

The linear block of the model is

$$x_{out}(k) = \frac{B(z)}{F(z)} x_{in}(k) + e(k), \tag{5.41}$$

where

$$\begin{aligned} B(z) &= 2.295z^{-1} - 5.111z^{-2} - 1.645z^{-3} + 8.629z^{-4} + z^{-5} - 7.17z^{-6} - 2.285z^{-7} + 5.158z^{-8} \\ &\quad + 0.5144z^{-9} - 1.706z^{-10} + 0.3204z^{-11} \\ F(z) &= 1 - 2.301z^{-1} - 0.2371z^{-2} + 3.411z^{-3} - 0.4691z^{-4} - 2.426z^{-5} - 0.2363z^{-6} + 1.95z^{-7} \\ &\quad - 0.2226z^{-8} - 0.6838z^{-9} + 0.04912z^{-10} + 0.1456z^{-11} + 0.1557z^{-12} \\ &\quad - 0.1792z^{-13} + 0.04452z^{-14}, \end{aligned} \tag{5.42}$$

and $n_b = 11,\ n_f = 14,\ n_k = 1$.

The input nonlinearity is piecewise-linear with 1 break point and the output nonlinearity is piecewise-linear with 19 break points:

Break points of input nonlinearity	Break points of output nonlinearity
(0.0396,0.0003)	$(-0.0163, -0.0830)$
	$(-0.0141, -0.0746)$
	$(-0.0118, -0.0636)$
	$(-0.0096, -0.0484)$
	$(-0.0073, -0.0271)$
	$(-0.0051, -0.0082)$
	$(-0.0028, 0.0166)$
	$(-0.0002, 0.0318)$
	$(0.0016, 0.0833)$
	$(0.004, -1.5448)$
	$(0.0062, -2.3383)$
	$(0.0084, -2.5396)$
	$(0.0107, -2.3017)$
	$(0.0129, -1.7773)$
	$(0.0152, -1.119)$
	$(0.0174, -0.4797)$
	$(0.0196, -0.0121)$
	$(0.0219, 0.1318)$
	$(0.0241, 0.0490)$

The identified model of the system is a spike-based filter, which has spiking input and output. The input to the model is the encoded signal $\{t_k^u\}$ and the output of the model is $\{t_k^y\}$.

The identification results are shown below. The relative error is the error divided by the 2-norm of $Y(k)$.

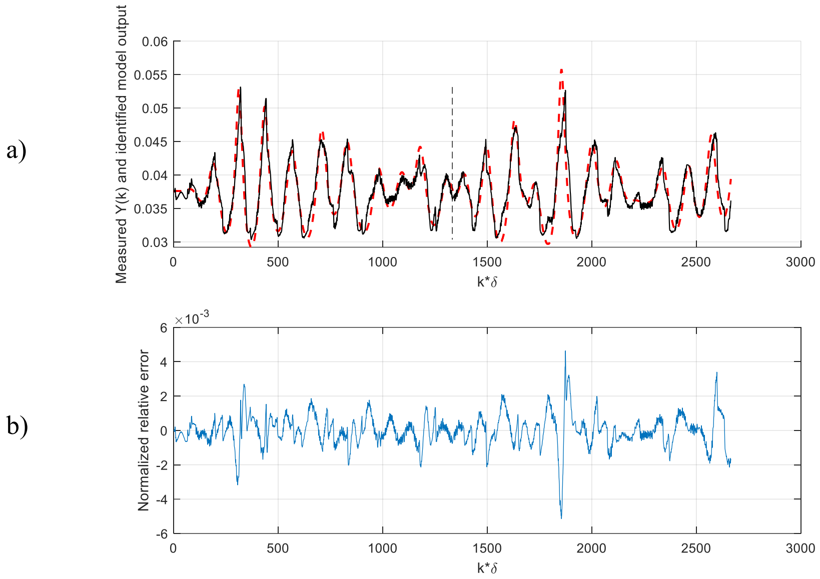


Fig. 5.13.a) Measured $Y(k)$ (red) and identified model output (black). The estimation data and validation data are divided by the dashed line.

b). Relative error of the identified model.

The identified model is shown to be able to model the behaviour of an EIEO model. It is shown that the identified model output has some jitters. This is because the linear block is selected with a relatively high order.

Fig. 5.14 shows the decoded signals and error.

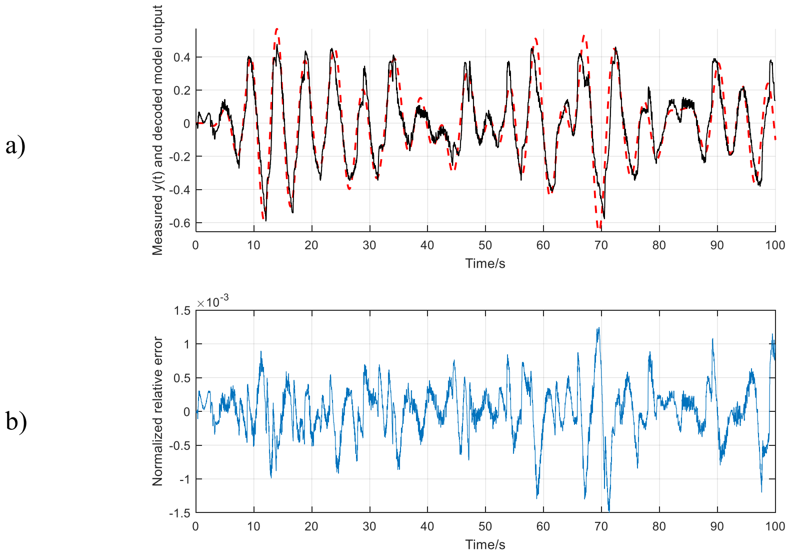


Fig. 5.14.a). Measured output $y(t)$ (red) and decoded system output (black).

b). Relative error.

The figure shows that the decoded output can model the measured output, although there are some jitters. The decoded output has some error compared with the measured output, but the error is relatively small and has a zero mean.

Example 4: Second-order system with two zeros

The transfer function of the system is

$$G(z) = \frac{\hat{y}(z)}{\hat{u}(z)} = \frac{z^2 - 1.99999z + 0.99999}{z^2 - 1.9997z + 0.9997}, T_s = 0.001s, \tag{5.43}$$

The encoded input spike interval $U(k)$ is utilized as the model input, while the encoded output spike interval $Y(k)$ is utilized as the model output. The data for estimation and validation are shown below.

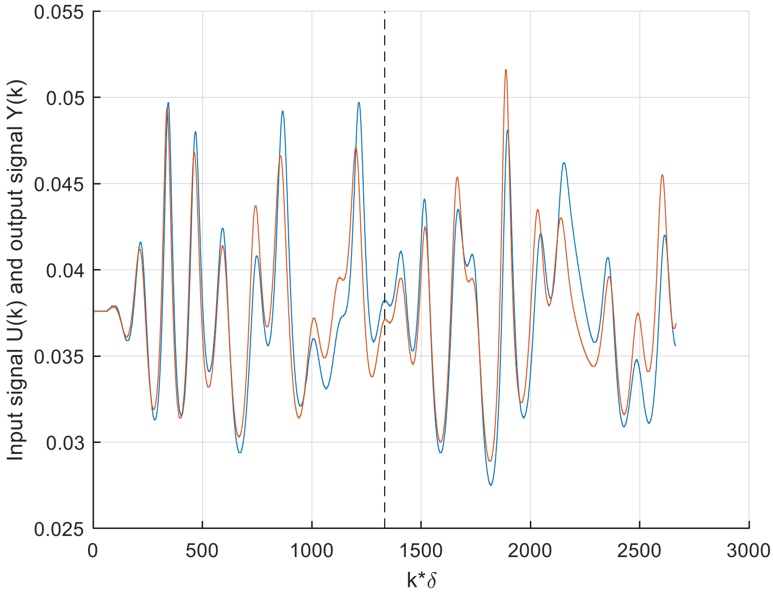


Fig. 5.15. Input signal $U(k)$ (blue) and output signal $Y(k)$ (red). The estimation data and validation data are divided by the dashed line.

The system is identified using Hammerstein-Wiener model.

The linear block of the model is

$$x_{out}(k) = \frac{B(z)}{F(z)} x_{in}(k) + e(k), \tag{5.44}$$

where

$$\begin{aligned} B(z) &= -0.5109 + z^{-1} - 0.995z^{-2} + 0.9676z^{-3} - 0.4619z^{-4} \\ F(z) &= 1 - 1.714z^{-1} + 1.671z^{-2} - 1.847z^{-3} + 0.831z^{-4} - 0.1658z^{-5} + 0.1461z^{-6} \\ &\quad + 0.01988z^{-7} - 0.01891z^{-8} + 0.102z^{-9} - 0.1155z^{-10} + 0.2702z^{-11} \\ &\quad - 0.239z^{-12} + 0.2453z^{-13} - 0.1413z^{-14} + 0.01616z^{-15} - 0.03849z^{-16} \\ &\quad - 0.008281z^{-17} + 0.1436z^{-18} - 0.1072z^{-19} + 0.2266z^{-20} - 0.5897z^{-21} \\ &\quad + 0.4329z^{-22} - 0.458z^{-23} + 0.3408z^{-24}, \end{aligned} \tag{5.45}$$

and $n_b = 5,\ n_f = 24,\ n_k = 1$.

The input nonlinearity is piecewise-linear with 26 break points and the output nonlinearity is one-dimensional polynomial of degree 5.

Break points of input nonlinearity piecewise-linear:

Break points of output nonlinearity
$(0.0302, -0.0075)$
$(0.0310, -0.0062)$
$(0.0317, -0.0061)$
$(0.0325, -0.0045)$
$(0.0332, -0.0042)$
$(0.0338, -0.0035)$
$(0.0348, -0.0025)$
$(0.0351, -0.002)$
$(0.0361, -0.001)$
$(0.0372, 0.0001)$
$(0.0377, 0.0005)$
$(0.0385, 0.0015)$
$(0.0393, 0.0021)$
$(0.0399, 0.0026)$
$(0.0430, 0.0050)$
$(0.0433, 0.0052)$
$(0.0434, 0.0054)$
$(0.0446, 0.0063)$
$(0.0471, 0.0076)$
$(0.0472, 0.0076)$
$(0.0472, 0.0077)$
$(0.0473, 0.0077)$
$(0.0490, 0.0085)$
$(0.0495, 0.0084)$
$(0.0510, 0.0078)$
$(0.0512, 0.0075)$

Output nonlinearity:

$$x_{out}(k) = 2.0016e07x_{in}^5(k) - 1.0681e06x_{in}^4(k) - 1.6375e04x_{in}^3(k) + 102.3508x_{in}^2(k) \\ - 2.1180x_{in}(k) + 0.0372 \tag{5.46}$$

The identified model of the system is a spike-based filter, which has spiking input and output. The input to the model is the encoded signal $\{t_k^u\}$ and the output of the model is $\{t_k^y\}$.

The identification results are shown below. The relative error is the error divided by the 2-norm of $Y(k)$.

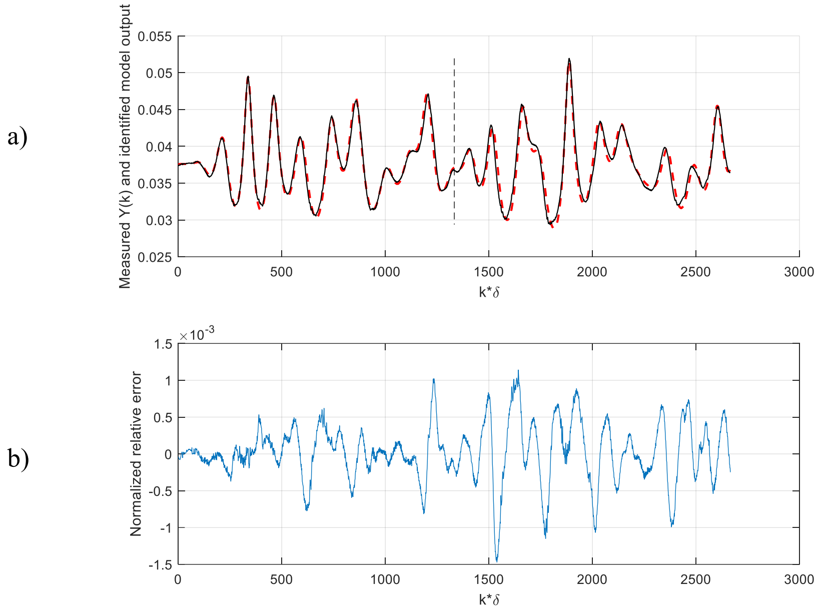


Fig. 5.16.a). Measured $Y(k)$ (red) and identified model output (black). The estimation data and validation data are divided by the dashed line.

b). Relative error of the identified model.

The identified model fit the estimation data very well with a small error. For the validation, the model has also very small error compared with the measured output and there is no shift between the model output and the estimation/validation data.

Fig. 5.17 shows the decoded signals and error.

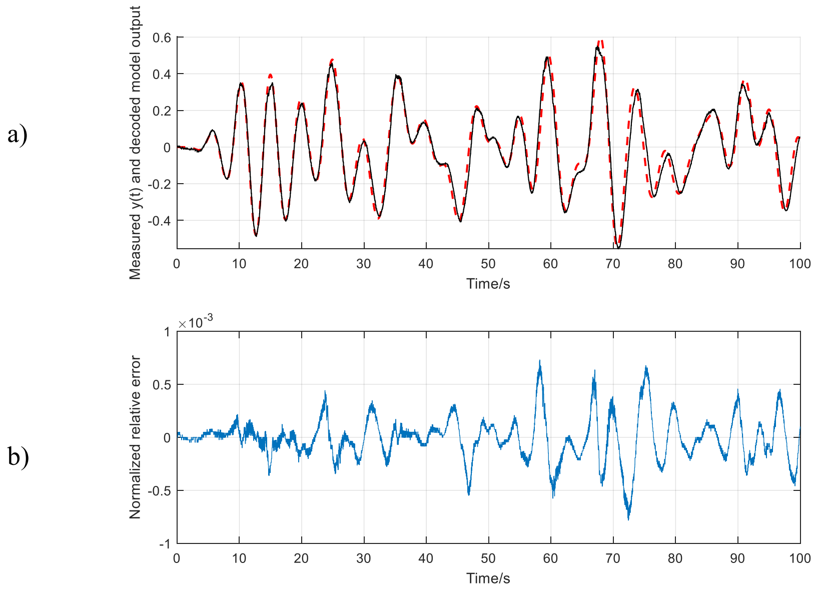


Fig. 5.18.a). Measured output $y(t)$ (red) and decoded system output (black).

b). Relative error.

It is shown that the decoded model output has very little difference between the measured continuous output. This indicates that the identified model can be utilized to replace an analogue filter in a system.

Example 5: Identification of a Low-pass Filter

Here the identification results of EIEO models of a low-pass filter are presented. The low-pass filter is a Butterworth filter [37]. The frequency response of the filter is

$$G(\omega) = \frac{\omega_N^n}{\sqrt{\omega_N^{2n}+\omega^{2n}}}, \tag{5.47}$$

where ω_N is the pass band frequency, n is the order of the filter, or the number of reactive elements in the filter.

The identification is carried using the encoded input $U(k)$ and encoded output $Y(k)$.

The MATLAB simulation is carried out using 200,001 data points, with 20 seconds of total simulation time and 0.0001 second of step length of simulation.

The random input is generated using a uniformly distributed white noise. The noise is fed to a low pass filter with passband edge frequency of 200 rad/s. Therefore, the input signal is a random bandlimited signal.

The original input signal $u(t)$ is fed to the designated filter and the order of the filter is 10. The output of the is $y(t)$. Parameter settings are shown in the following table.

Table 5.2. Simulation parameters.

Parameter	Value	Description
δ	0.025	IF neuron threshold
b	0.5	IF neuron bias
C	1	IF integration constant
Ω	200	Signal bandwidth
ω_N	100	Passband edge frequency of the low-pass filter

The input signal $u(t)$ and output signal $y(t)$ of a low-pass filter are encoded into spike trains $\{t_k^u\}$ and $\{t_k^y\}$, with corresponding spike intervals $U(k)$ and $Y(k)$.

The encoded input spike interval $U(k)$ is utilized as the model input, while the encoded output spike interval $Y(k)$ is utilized as the model output.

The input signal $u(t)$ and output signal $y(t)$ of the filter is shown below.

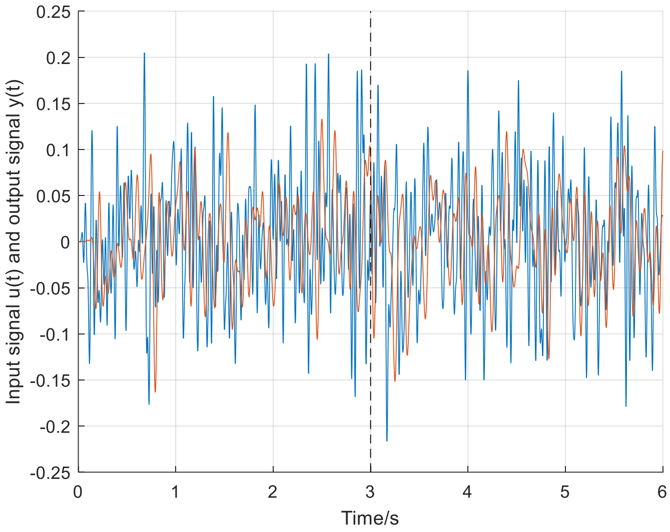


Fig. 5.19. Input signal (blue) and output signal (red) of the low-pass filter.

The data for estimation and validation are shown in Fig. 5.20.

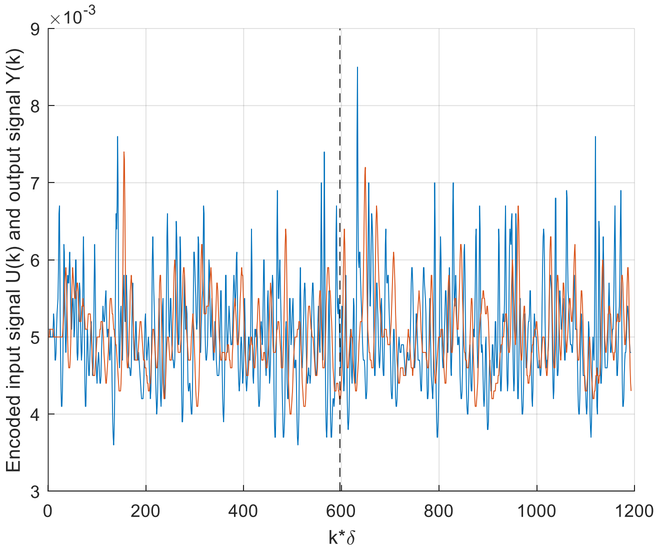


Fig. 5.20. Input signal $U(k)$ (blue) and output signal $Y(k)$ (red). The estimation data and validation data are divided by the dashed line.

The model is identified using Hammerstein-Wiener model.

The linear block of the model is

$$x_{out}(k) = \frac{B(z)}{F(z)} x_{in}(k) + e(k), \tag{5.48}$$

where

$$B(z) = 0.1064z^{-7} - 0.5399z^{-8} + z^{-9} - 0.6054z^{-10} - 0.5078z^{-11} + 1.021z^{-12} - 0.6027z^{-13} + 0.1282z^{-14}$$

$$F(z) = 1 - 3.587z^{-1} + 3.262z^{-2} + 2.31z^{-3} - 4.79z^{-4} + 0.7482z^{-5} + 1.5z^{-6} + 0.2178z^{-7} - 0.9704z^{-8} + 0.3101z^{-9}, \tag{5.49}$$

and $n_b = 14,\ n_f = 9,\ n_k = 1$.

The input nonlinearity is piecewise-linear with 2 break points and the output nonlinearity is piecewise-linear with 2 break points:

Break points of input nonlinearity	Break points of output nonlinearity
$(0.0088, 0.00158)$	$(-0.0007, 0.0033)$
$(0.0144, 0.0029)$	$(0.0003, 0.0059)$

The identification results are shown below. The relative error is the error divided by the 2-norm of $Y(k)$.

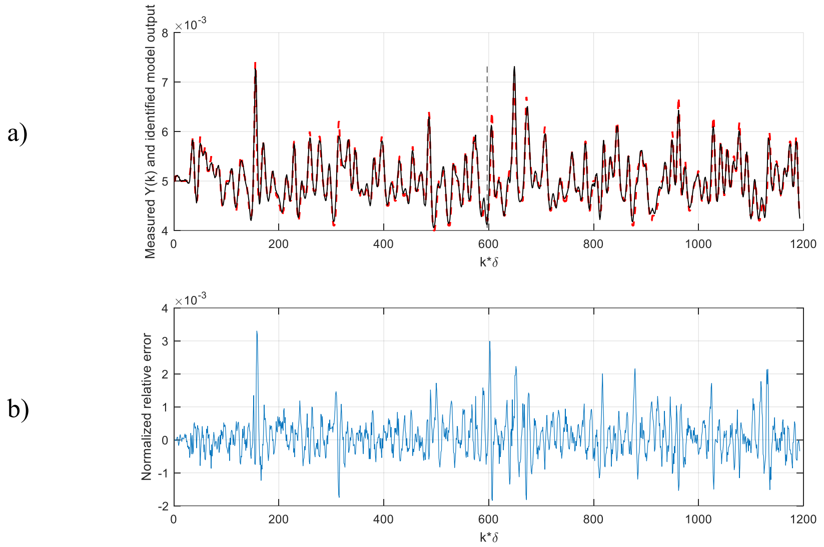


Fig. 5.21.a). Measured $Y(k)$ (red) and identified model output (black). The estimation data and validation data are divided by the dashed line.

b). Relative error of the identified model.

It is shown in the figure that the identified model fits the estimation data well and the error is not increasing for data validation. The error is relatively small compared with $Y(k)$ and does not accumulate over samples.

Fig. 5.22 shows the decoded signal and error.

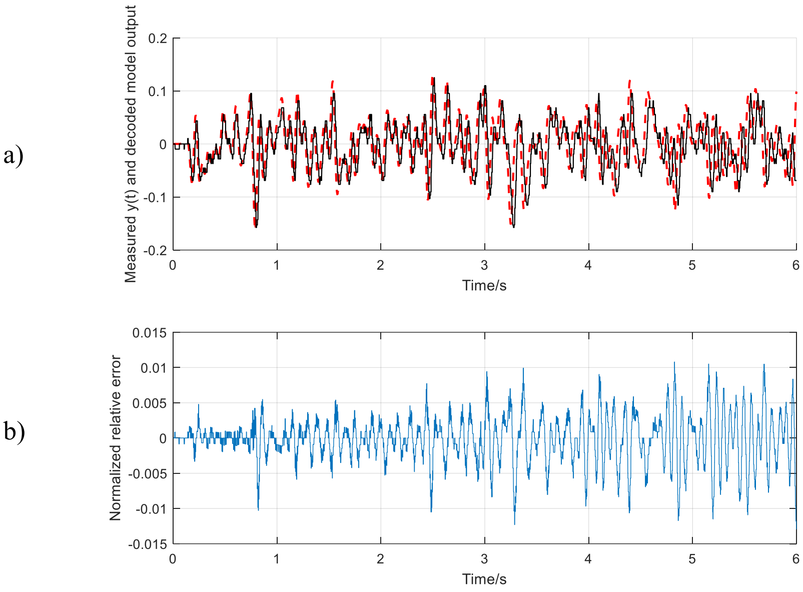


Fig. 5.22.a). Measured output $y(t)$ (red) and decoded system output (black)

b). Relative error.

It is shown in the figure that the decoded model output has a delay and a difference to the measured continuous output. However, the function of the filter is to filter the input frequencies. The identified model is suitable for this task since the frequency spectrum of $y(t)$ and the decoded model output is significantly close, which is shown below.

Fig. 5.23 shows the single-sided spectrum of the filter output and the decoded model output.

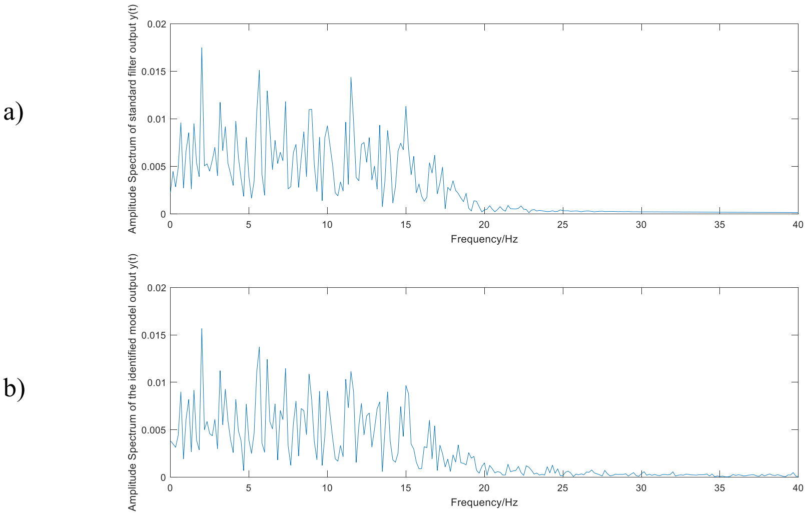


Fig. 5.23.a). Spectrum of the filter output.

b). Spectrum of the decoded model output.

The figure shows that the two spectrums are close to each other. The spectrum of the decoded model output has very similar extreme points as the filter output. The amplitude of both spectrums converge to zero after 20 Hz.

Example 6: Identification of a nonlinear system: Duffing equation

Here the encoded input and output signals of a Duffing Equation is identified. The Duffing Equation is given by

$$y''(t) = u(t) - ky'(t) - y^3(t) \tag{5.50}$$

where k is set to 1.

the standard input and output signal of the Duffing Equation are shown below.

a)

b)

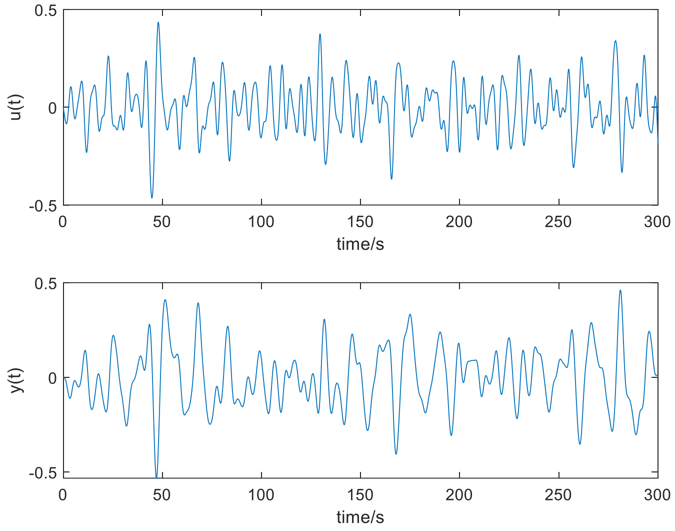


Fig. 5.24.a). Input signal of a standard Duffing Equation system.

b). Output signal of a standard Duffing Equation system.

The encoded input spike interval $U(k)$ is utilized as the model input, while the encoded output spike interval $Y(k)$ is utilized as the model output. The data for estimation and validation are shown below.

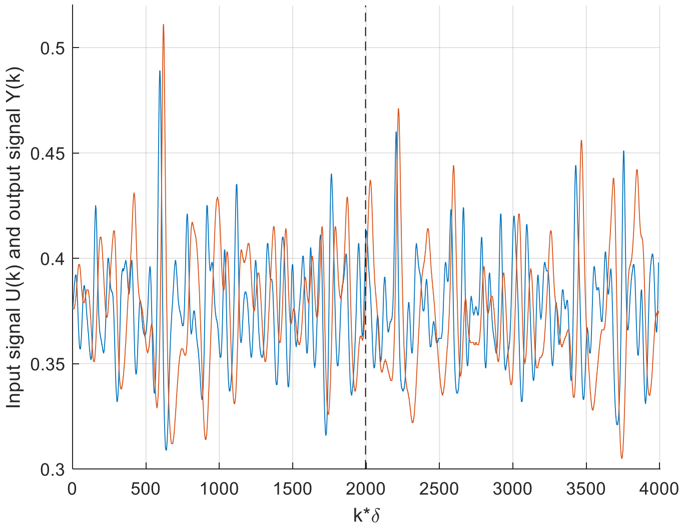


Fig. 5.25. Input signal $U(k)$ (blue) and output signal $Y(k)$ (red). The estimation data and validation data are divided by the dashed line.

The system is identified using Hammerstein-Wiener model.

The linear block of the model is

$$x_{out}(k) = \frac{B(z)}{F(z)} x_{in}(k) + e(k), \tag{5.51}$$

where

$$\begin{aligned} B(z) &= 0.6711z^{-1} - 0.9106z^{-2} - 0.8204z^{-3} + 0.9169z^{-4} + z^{-5} - 0.4519z^{-6} - 0.8554z^{-7} \\ &\quad + 0.4507z^{-8} \\ F(z) &= 1 - 1.158z^{-1} - 1.336z^{-2} + 1.081z^{-3} + 1.404z^{-4} - 0.4179z^{-5} - 1.067z^{-6} + 0.4942z^{-7}, \end{aligned} \tag{5.52}$$

and $n_b = 8,\ n_f = 7,\ n_k = 1$.

The input nonlinearity is piecewise-linear with 5 break points and the output nonlinearity is piecewise-linear with 8 break points:

Break points of input nonlinearity	Break points of output nonlinearity
$(0.3155, -0.0033)$	$(-0.1914, -1.0059)$
$(0.3808, 0.0004)$	$(-0.1380, -0.9903)$
$(0.4101, 0.0019)$	$(-0.1342, -0.9349)$
$(0.4181, 0.0019)$	$(-0.0774, -0.5198)$
$(0.4402, 0.0040)$	$(-0.0425, -0.1406)$
	$(-0.0016, 0.3455)$
	$(0.0620, 1.3520)$
	$(0.0674, 1.4730)$

The identified model of the system is a spike-based filter, which has spiking input and output. The input to the model is the encoded signal $\{t_k^u\}$ and the output of the model is $\{t_k^y\}$.

The identification results are shown below. The relative error is the error divided by the 2-norm of $Y(k)$.

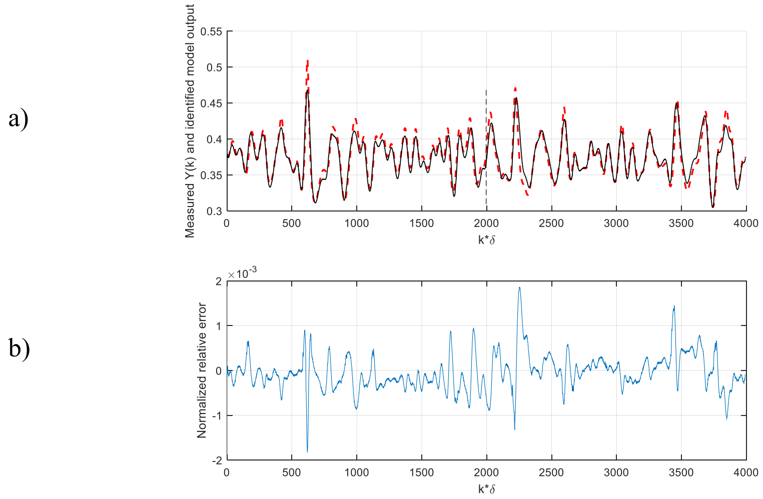


Fig. 5.26.a). Measured $Y(k)$ (red) and identified model output (black). The estimation data and validation

data are divided by the dashed line.

b). Relative error of the identified model.

The figure shows that the error is relatively large at some peak points of $Y(k)$. Nevertheless, the error is small and the identified model is able to mimic the behaviour of the EIEO model.

In Fig. 5.27 and Fig. 5.28, the plot of input $u(t)$ and output $y(t)$ of both the standard system and identified model are presented.

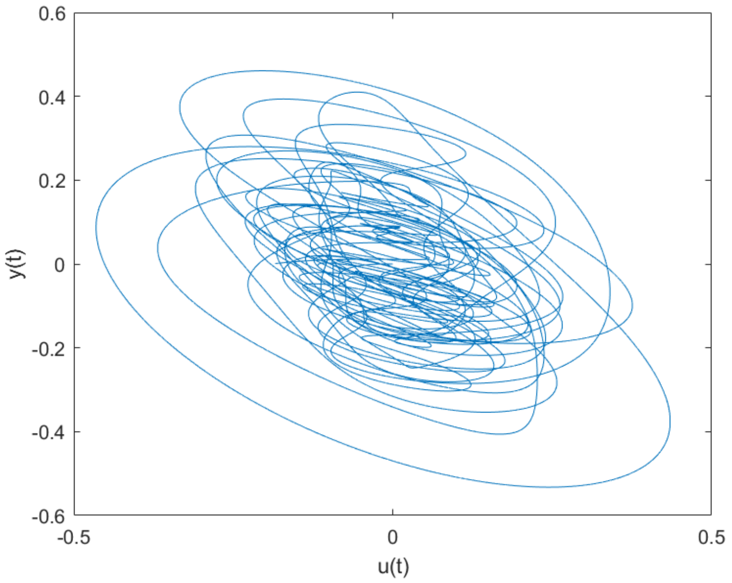


Fig. 5.27. Input $u(t)$ and output $y(t)$ of the standard Duffing Equation.

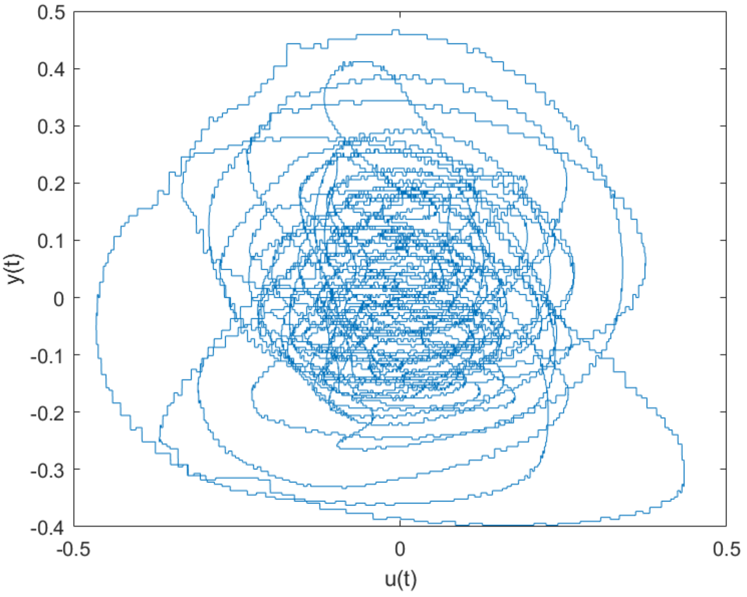


Fig. 5.28. Input $u(t)$ and output $y(t)$ of the identified model.

Fig. 5.23 and fig. 5.30 shows the output $y(t)$ and delayed output $y(t-1)$ of the standard system and the identified model. Notice that the sampling time of the simulation is 0.0001 second, there are 10000 steps delay of $y(t-1)$ to $y(t)$.

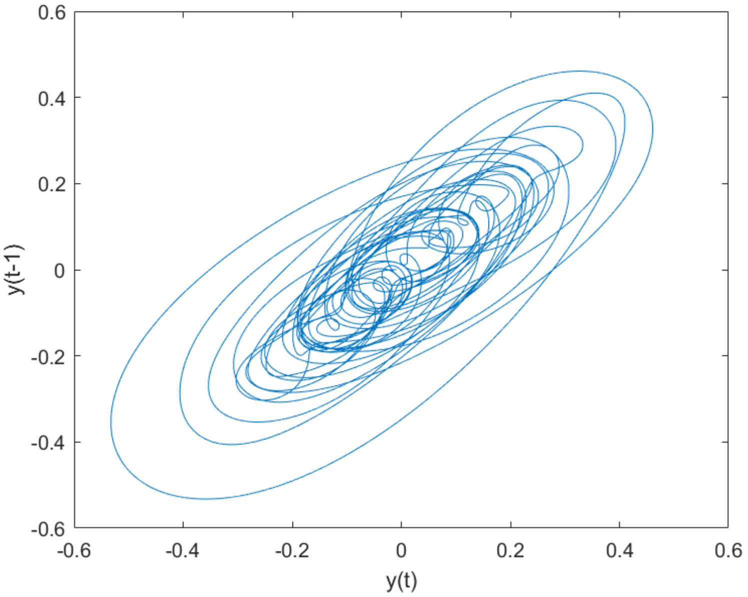


Fig. 5.29. Output $y(t)$ and delayed output $y(t-1)$ of the standard Duffing Equation.

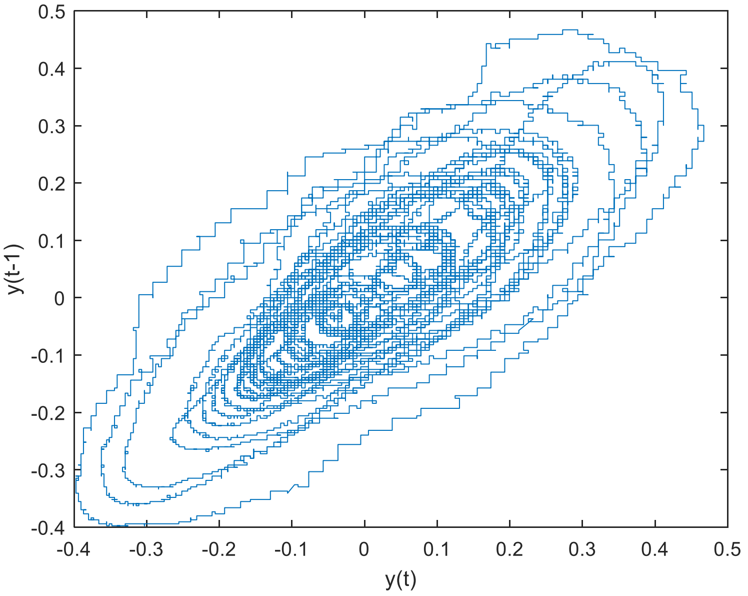


Fig. 5.30. Output $y(t)$ and delayed output $y(t-1)$ of the identified model.

From Fig. 5.27 and 5.29, it is shown that the Duffing Equation system has some chaos behaviours. Fig. 5.28 and 5.30 show that the decoded model output can model the chaos behaviour to some extent. Therefore, one can utilize an identified model to model a Duffing Equation system.

5.5. Causality problem of the EIEO model

5.5.1. Problem description

For an EIEO model of a spike filter, the input and output signals are in the format of spikes. Let $\{t_k^u\}_{k\in\mathbf{Z}^+}$ and $\{t_k^y\}_{k\in\mathbf{Z}^+}$ be the input and output spike train of a discrete spike filter, respectively. In a practice signal processing scenario, the spikes are input to the filter, and the filter outputs spikes in real-time. Actually, the filter reacts to the input spike arriving time, and outputs another spike arriving time. It is possible that the output spike arrives before the input spike, i.e., $t_k^y < t_k^u$. Fig. 5.32 illustrates the scenario.

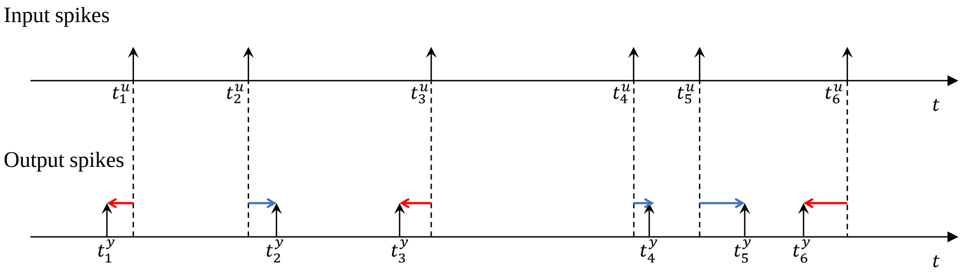


Fig. 5.31. Possible input and output spike trains of the filter.

Here, $t_k^y < t_k^u$ for $k = 1, 3, 6$. It yields that the filter must output corresponding spikes before the 1st, 3rd and 6th spikes arrive. It is an anti-causal process, which cannot be implemented in practice.

5.5.2. Solution to the causality problem

Step1. Output spike shifting

An approach to the solution to the causality problem is adding a shift ΔT to the output spikes. The shifted spikes are $\hat{t}_k^y = t_k^y + \Delta T$. The shift ΔT is selected properly so that $\hat{t}_k^y > t_k^u, \forall k$. The output spike shifting is shown in Fig. 5.31.

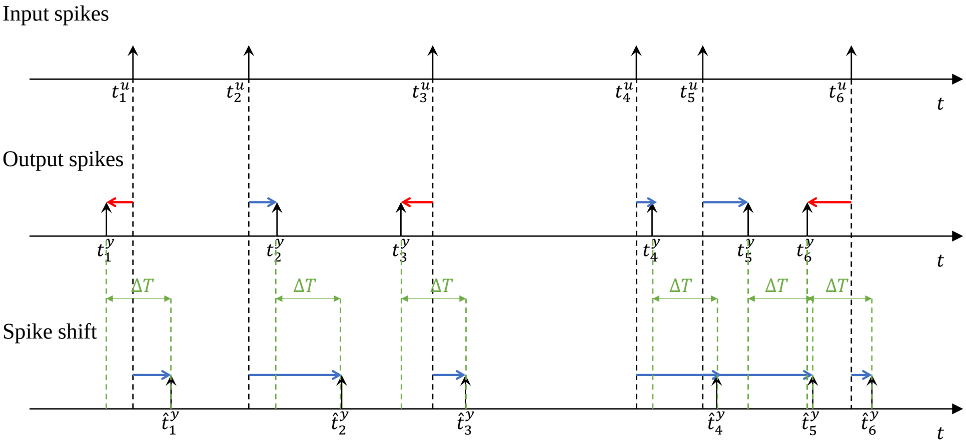


Fig. 5.32. Shift the output spikes.

As shown in the figure, the shifted output spikes $\{\hat{t}_k^y\}_{k\in\mathbf{Z}^+}$ always comes after the input spikes. This ensures the causality of the process. The spike shift ΔT is required to be larger than or equal to the maximum difference of t_k^u and t_k^y, i.e., $\Delta T \geq \max(t_k^u - t_k^y)$.

Step 2. Decoding the shifted output spike train

According to the previous step, the model outputs a spike train which is shifted for ΔT. Therefore, when the kth spike $\{t_k^y\}$ is output at time instant t, it actually transmits the information at $t - \Delta T$. Here, the fast decoding algorithm is applied to reconstruct the output signal.

The output signal $\tilde{y}(t)$ is firstly estimated at the mid points $\hat{s}_{k+1}$ of spikes. The mid points are shifted because the spikes are shifted, i.e., $\hat{s}_{k+1} = s_{k+1} + \Delta T$, where $s(k) = \frac{t_{k+1}^y + t_k^y}{2}$ and $\hat{s}(k) = \frac{\hat{t}_{k+1}^y + \hat{t}_k^y}{2}$.

At $t = \hat{s}_{k+1}$, the output signal is estimated as

$$\begin{aligned}\tilde{y}(\hat{s}_{k+1}) &= \frac{C\delta}{t_{k+1}^y - t_k^y} - b \\ &= \frac{C\delta}{(\hat{t}_{k+1}^y - \Delta T) - (\hat{t}_k^y - \Delta T)} - b \\ &= \frac{C\delta}{\hat{t}_{k+1}^y - \hat{t}_k^y} - b, \end{aligned} \tag{5.53}$$

where C is the integral constant, δ is the neuron threshold and b is the bias.

Notice the mid points $\hat{s}_{k+1}$ are shifted, $\tilde{y}(\hat{s}_{k+1})$ actually estimates the output at $t = \hat{s}_{k+1} - \Delta T$. Therefore, the estimation will have a delay of ΔT. At time instant $t = t_{k+1}$, the decoded signal indicates the amplitude of the original signal $y(t)$ at $t = \hat{s}_{k+1} - \Delta T$. The real-time system can be implemented by estimating $y(t)$ at $t = \hat{s}_{k+1}$ using interpolation/extrapolation methodologies.

There are two methodologies to reach a continuous signal from the estimation at $t = \hat{s}_{k+1}$.

Methodology 1. Zero-order hold:

By simply adding a zero-order hold, a continuous signal is reached. However, there is a delay of ΔT.

Methodology 2. First-order hold:

In order to eliminate the delay caused by spike shift, one can apply a first-order hold to predict the signal at $t = \hat{s}_{k+1}$, using the estimation at $t = \hat{s}_{k+1} - \Delta T$.

The estimation of the output signal has been given for time $t = \hat{s}_{k+1}$. However, this estimation actually gives the value of the output at time $t = \hat{s}_{k+1} - \Delta T = s_{k+1}$. Here, a first-order hold is applied to predict the output signal at $t = \hat{s}_{k+1}$.

The output signal is given by

$$\begin{aligned}\tilde{y}(t) &= \tilde{y}(s_{k+1}) + K \cdot (t - s_{k+1}) \\ &= \tilde{y}(s_{k+1}) + K \cdot \big(t - (\hat{s}_{k+1} - \Delta T)\big) \\ &= \frac{C\delta}{\hat{t}_{k+1}^y - \hat{t}_k^y} - b + K \cdot \left(t - \frac{\hat{t}_{k+1}^y + \hat{t}_k^y}{2} + \Delta T\right), \end{aligned} \tag{5.54}$$

where

$$K = \frac{\tilde{y}(s_{k+1}) - \tilde{y}(s_k)}{s_{k+1} - s_k}$$

$$\begin{aligned} &= \frac{\tilde{y}(s_{k+1}) - \tilde{y}(s_k)}{\hat{s}_{k+1} - \hat{s}_k} \\ &= \frac{\left(\frac{C\delta}{\hat{t}^y_{k+1} - \hat{t}^y_k} - b\right) - \left(\frac{C\delta}{\hat{t}^y_k - \hat{t}^y_{k-1}} - b\right)}{\frac{\hat{t}^y_{k+1} + \hat{t}^y_k}{2} - \frac{\hat{t}^y_k + \hat{t}^y_{k-1}}{2}} \\ &= \frac{2C\delta \cdot \left(\frac{1}{\hat{t}^y_{k+1} - \hat{t}^y_k} - \frac{1}{\hat{t}^y_k - \hat{t}^y_{k-1}}\right)}{\hat{t}^y_{k+1} - \hat{t}^y_{k-1}}, \end{aligned} \tag{5.55}$$

for $k = 2,3,4, \ldots$

Compared with methodology 1, this methodology using first-order hold eliminates the output delay, but may cause larger modelling error.

5.5.3. Simulation results

The simulation is carried out using the same settings as Section 5.4.1. The system models (a first order system and second order systems) are also the same. Compare the results with those in the file *Linear Identification*.

Here the reconstructed output signal using shifted spikes is shown, through two different methodologies.

First order system

$\max(t^u_k - t^y_k) = 0.1926$, ΔT is select as $0.2s$.

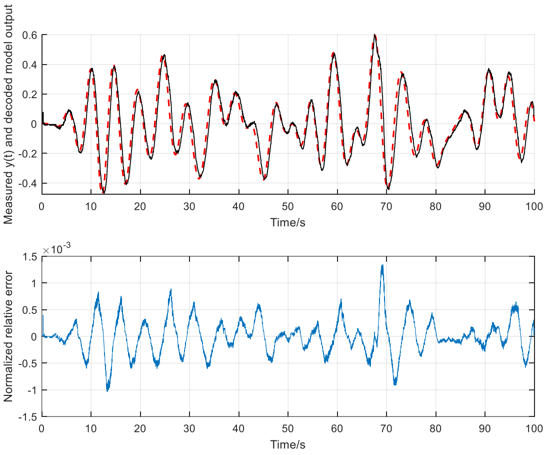


Fig. 5.33.a). Measured $y(t)$ (red) and decoded model output (black) using ZOH, $\Delta T = 0.2s$.

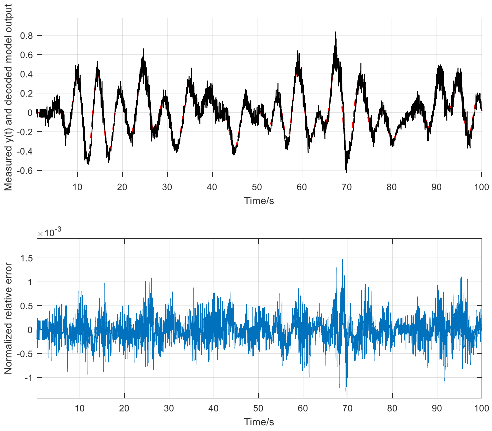


Fig. 5.33.b). Measured $y(t)$ (red) and decoded model output (black) using FOH, $\Delta T = 0.2s$.

Second order system

$\max(t_k^u - t_k^y) = 1.1746$, ΔT is select as $1.2s$.

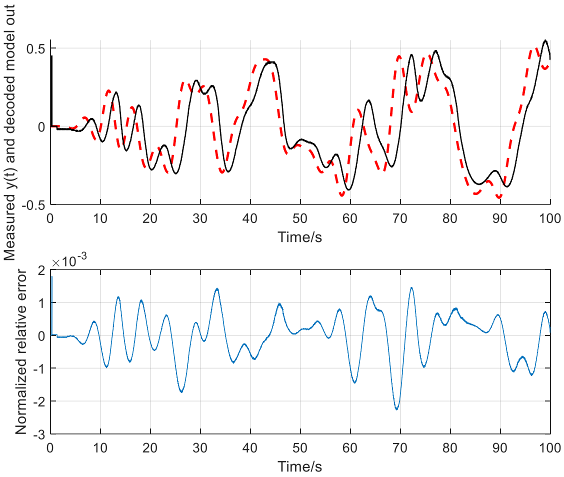


Fig. 5.34.a). Measured $y(t)$ (red) and decoded model output (black) using ZOH, $\Delta T = 1.2s$.

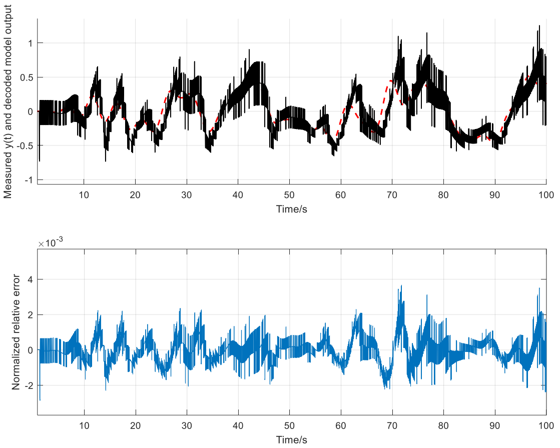


Fig. 5.34.b). Measured $y(t)$ (red) and decoded model output (black) using FOH, $\Delta T = 1.2s$.

Second order system with one zero

$\max(t_k^u - t_k^y) = 0.4258$, ΔT is select as $0.5s$.

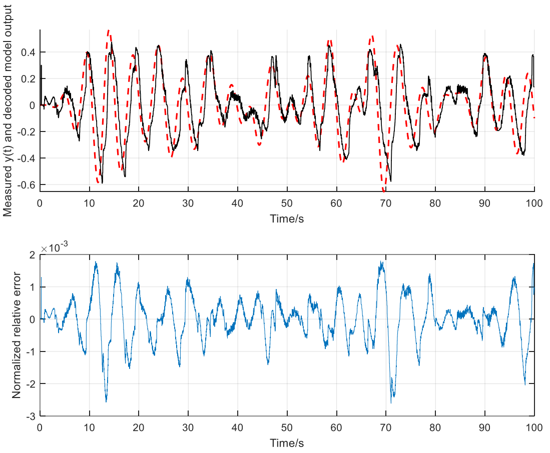


Fig. 5.35.a). Measured $y(t)$ (red) and decoded model output (black) using ZOH, $\Delta T = 0.5s$.

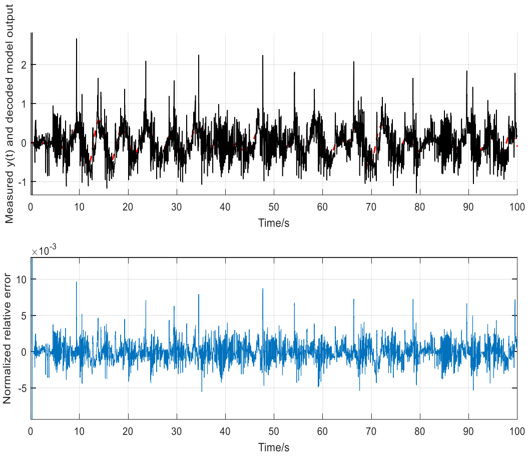


Fig. 5.35.b). Measured $y(t)$ (red) and decoded model output (black) using FOH, $\Delta T = 0.5s$.

Second order system with two zeroes

$\max(t_k^u - t_k^y) = 0.3259$, ΔT is select as $0.5s$.

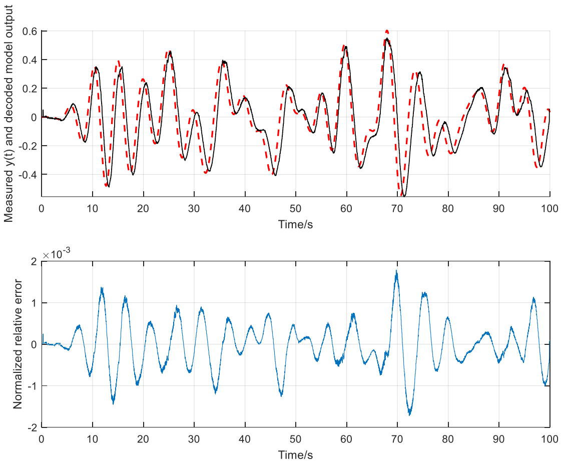


Fig. 5.36.a). Measured $y(t)$ (red) and decoded model output (black) using ZOH, $\Delta T = 0.5s$.

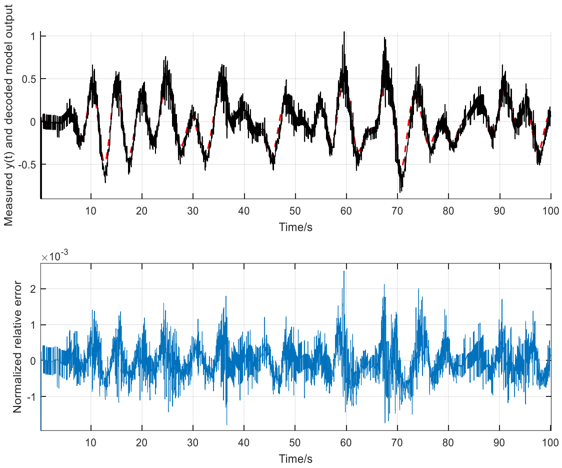


Fig. 5.36.b). Measured $y(t)$ (red) and decoded model output (black) using FOH, $\Delta T = 0.5s$.

From Fig. 5.33 to 5.36, it is shown that the models using ZOHs has much fewer jitters than that using FOHs. However, the models using ZOHs introduce a time delay. On the other hand, although the ZOH eliminates the delay, it leads to a much more jittering output, which is challenge to the hardware. It can also be concluded that both the delay and jitter are caused by the time shift ΔT. A larger ΔT leads to a larger delay and more jitters. Therefore, one can reduce the delay and jitter by decreasing ΔT, which can be realized by decreasing threshold δ, increasing bias b or employing multi-channel time encoding and decoding as discussed in Chapter 3.

5.6. Conclusion

In this chapter, a practical framework for identification of EIEO models of dynamical systems based on the encoded spiking input and output signals of a system was introduced. It has shown through numerical simulation using both linear and nonlinear systems that the identified EIEO models can approximate well the encoded outputs. The identified models fitted the estimation/validation data well. The decoded model outputs could also model the measured outputs at a high accuracy. This indicates that once an identified EIEO model starts responding to a spiking input signal, its spiking output can be immediately utilized in a system or decoded into a continuous signal. Therefore, the identified EIEO models can be used to process directly time encoded signals providing the basis for implementing signal processing operations directly on time encoded signals. The time encoded signal can be decoded after processing in real-time using the fast decoding algorithm introduced in Chapter 3.

This chapter utilized Hammerstein-Wiener model to identify the model. The models were designed to fit the following systems: four linear systems, a low-pass filter and a nonlinear Duffing Equation system.

For the linear systems, it has shown that the identified model outputs fitted the measured spike intervals very well and the decoded outputs could model the measured continuous system output signals with relatively small error. For the low-pass filter, the decoded identified model output could also model the measured filtered signal with a significantly similar spectrum. For the nonlinear Duffing Equation system, the identified model could mimic the chaos behaviour of the Duffing Equation. Therefore, it can be concluded that the identified model can be utilized to respond to and transmit spiking signals and replace analogue linear/nonlinear filters in a real-time system.

A causality problem was also discussed in this chapter. The problem was caused by the difference between the input and output spiking signals. As illustrated in this chapter, the EIEO models responded to input spike timings and output also spike timings. Under certain circumstances, the output spike timings are required to be generated before the input spike timings are given, which breaks the causality of the system. Therefore, a solution to this causality problem is required. To this end, this chapter proposed a solution that shifts the output spike train. Via simulation, it has shown that the shift of the output spike train solved the causality problem. However, one must select proper extrapolation methodologies: Zero-Order hold method has fewer jitters while First-Order hold method eliminates the delay.

According to the simulation results, the order of the identified EIEO models were relatively high. This could be challenging because it increases the complexity of realization and the cost of hardware. One can utilize alternative models for identification such as NAMAX models. Also, under certain conditions, some high-order systems with dominant poles can be approximated as second-order systems, which simplifies the implementations at the cost of accuracy.

Chapter 6
Conclusion and future work

6.1. Conclusion

In 2003, Lazar et al [13]. introduced the Integrate-and-Fire Time Encoding Machine (IF-TEMs) as an alternative sampling approach that mimic the behaviour of neurons in the brain and convert continuous signals into discrete spike timings.

IF-TEMs are particularly valuable in scenarios where spike-based communication is advantageous. Specifically, time encoding allows representing and processing information efficiently which reduces energy consumption, making it suitable for low-power applications like sensor networks and mobile devices as well as for bio-inspired computing and communication systems.

Lazar also proposed an information-loss-free reconstruction algorithm for retrieving the original signal post-transmission [3]. Nonetheless, the information-loss-free reconstruction algorithm necessitates transmitting the entire spike train, a process that can span a significant duration. This limitation of Lazar's information-loss-free reconstruction algorithm restricts its suitability for real-time control systems, hindering the application of spiking data transmission in such contexts.

To enable the practical application of Integrate-and-Fire Time Encoding/Decoding Machines in real-time systems, this thesis introduced a novel and efficient real-time decoding algorithm tailored for IF-TEMs.

The proposed real-time single-channel decoding algorithm provides a piece-wise constant approximation of the original signal over the spike time intervals, which is suitable in the implementation of real-time control systems.

Analytical bounds are derived for the approximation error at the mid-point of the spike-time interval and it is shown though numerical simulations that proposed real-time decoding algorithm outperforms Lazar's algorithm at these points. The analytical error bound show explicitly the dependence of the error on the key parameters of IF-TEM i.e., the neuron threshold and the additive bias.

The fast decoding algorithm, facilitates the utilization of time encoding within a real-time control framework. To that end, Chapter 4 introduces a spike-based network control system, a key component of which is a spike-based PID control law. Specifically, approximations for the proportional, integral, and derivative terms of the PID controller are formulated in terms of the spike times alongside analytical bounds for the errors introduced by the approximations. A methodology for tuning PID controller gains such that the performance of the spike-based PID controller closely matches that of the continuous time one is also introduced. Numerical simulation studies are used to demonstrate the performance of the spike-based PID control system's performance in noise-free as well as noisy conditions.

Specifically, the robustness of the spike-based PID controller is demonstrated by considering both output measurement noise as well as spike time jitter (noise).

Compared with a standard PID control system, a spike-based PID control system has the following advantages as described before:

- Energy Efficiency on the aspect of the entire system.
- Simpler devices used for receiving and sending spiking signals.
- Sparse Data Representation.
- Robustness to noise.

- Event-Driven Processing.

In an ideal scenario, time encoding should replace conventional amplitude-sampling used not just in control but for general signal processing. To that end, Chapter 5 introduces a system identification approach for developing Encoded-Input-Encoded-Output models of linear and nonlinear dynamical systems, which can be used to implement basic linear and nonlinear filtering operations by directly processing spike time sequences. The proposed approach involves encoding both the simulated input and output of a dynamical system of interests and applying system identification methodology to derive the EIEO model based on the input and output spike time interval sequences. This methodology enables the creation of spike-based linear and nonlinear operators as well as filters. Given that spike-based models deal with inputs and outputs based on spike arrival times, there is a potential risk of introducing anti-causal issues into a system. To address this concern, this thesis introduces a solution that guarantees causality. By designing spike-based components in this manner, it becomes possible to construct a system founded on spiking signal processing.

6.2. Future work

The research presented in this thesis can be extended further in a number of directions:

1. The rapid real-time decoding algorithm for TEMs initially approximates the original signal at the midpoint of spikes, after which the continuous output signal is produced utilizing either a Zero-Order Hold or a First-Order Hold. Alternative signal extrapolation techniques may also be devised or utilized to create the continuous output signal, potentially leading to a reduction in the overall decoding error associated with this algorithm.
2. Through the simulation, it was shown that the true error bound at the spike mid-points are much smaller than the theoretical bound suggesting that the analytical error bound could be refined further.
3. This thesis focuses mainly on Ideal Integrate-and-Fire-TEMs. The fast decoding algorithm could also be extended to other types of neurons, such as IF neurons with refectory periods and Leaky IF neurons.
4. The spike-based PID controller developed in this thesis is tuned based on the gains derived for conventional continuous-time PID controller. A new methodology for the tunning directly the gains of the spike-based PID controllers could also be developed.
5. This thesis proposes the designing of spike-based components that respond to spikes and output also spikes. It is possible to design other types of components using the same identification-based methodology. For example, one could design components that respond to spikes and output continuous signals.
6. All the works and results in this thesis are developed in order to enable the real-time spiking data transmission. It is simple for the designer to employ the methodologies and algorithms to construct a real-time spike-based control system. Also, these results could be applied in the field of Spiking Neuron Networks, where the spiking neurons communicate with each other using spikes. The proposed fast real-time decoding algorithm could be employed to decode the neuron

outputs, while the spike-based components could be applied to model the behaviour of some neurons or an entire SNN.

References

[1] Lathi, Bhagwandas P. "Modern digital and analog communication systems." Oxford University Press, Inc., 1990.

[2] Wang, Alice, and Anantha Chandrakasan. "Energy-efficient DSPs for wireless sensor networks." IEEE signal processing magazine 19.4 (2002): 68-78.

[3] Lazar, Aurel A., and László T. Tóth. "Perfect recovery and sensitivity analysis of time encoded bandlimited signals." IEEE Transactions on Circuits and Systems I: Regular Papers 51.10 (2004): 2060-2073.

[4] Han, Joon-Kyu, et al. "A review of artificial spiking neuron devices for neural processing and sensing." Advanced Functional Materials 32.33 (2022): 2204102.

[5] Pei, Xing, Lon Wilkens, and Frank Moss. "Noise-mediated spike timing precision from aperiodic stimuli in an array of Hodgekin-Huxley-type neurons." Physical review letters 77.22 (1996): 4679.

[6] Jimenez-Fernandez, Angel, et al. "A neuro-inspired spike-based PID motor controller for multi-motor robots with low cost FPGAs." Sensors 12.4 (2012): 3831-3856.

[7] Mie, Shunsuke, Yuichi Okuyama, and Hiroaki Saito. "Simplified quadcopter simulation model for spike-based hardware PID controller using SystemC-AMS." 2018 IEEE 12th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC). IEEE, 2018.

[8] Perez-Peña, Fernando, et al. "Neuro-inspired spike-based motion: From dynamic vision sensor to robot motor open-loop control through spike-VITE." Sensors 13.11 (2013): 15805-15832.

[9] Henry, Gary T. Practical sampling. Vol. 21. Sage, 1990.

[10] Malmirchegini, Mehrzad, et al. "Non-uniform sampling based on an adaptive level-crossing scheme." IET Signal Processing 9.6 (2015): 484-490.

[11] Hernandez, Luis, and Enrique Prefasi. "Analog-to-digital conversion using noise shaping and time encoding." IEEE Transactions on Circuits and Systems I: Regular Papers 55.7 (2008): 2026-2037.

[12] Gontier, David, and Martin Vetterli. "Sampling based on timing: Time encoding machines on shift-invariant subspaces." Applied and Computational Harmonic Analysis 36.1 (2014): 63-78.

[13] Lazar, Aurel A., and László T. Tóth. "Time encoding and perfect recovery of bandlimited signals." 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).. Vol. 6. IEEE, 2003.

[14] Feichtinger, Hans G., and Karlheinz Gröchenig. "Theory and practice of irregular sampling." Wavelets. CRC Press, 2021. 305-363.

[15] Lazar, Aurel A. "Time encoding with an integrate-and-fire neuron with a refractory period." Neurocomputing 58 (2004): 53-58.

[16] Florescu, Dorian, and Daniel Coca. "A novel reconstruction framework for time-encoded signals with integrate-and-fire neurons." Neural computation 27.9 (2015): 1872-1898.

[17] Lazar, Aurel A., Ernő K. Simonyi, and László T. Tóth. "A real-time algorithm for time decoding machines." 2006 14th European Signal Processing Conference. IEEE, 2006.

[18] Lazar, Aurel A., and Eftychios A. Pnevmatikakis. "Faithful representation of stimuli with a population of integrate-and-fire neurons." Neural computation 20.11 (2008): 2715-2744.

[19] Maass, Wolfgang. "Networks of spiking neurons: the third generation of neural network models." Neural networks 10.9 (1997): 1659-1671.

[20] Adam, Karen, Adam Scholefield, and Martin Vetterli. "Encoding and decoding mixed bandlimited signals using spiking integrate-and-fire neurons." ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020.

[21] Lazar, Aurel A., Ernö K. Simonyi, and László T. Tóth. "An overcomplete stitching algorithm for time decoding machines." IEEE Transactions on Circuits and Systems I: Regular Papers 55.9 (2008): 2619-2630.

[22] Adam, Karen, Adam Scholefield, and Martin Vetterli. "Sampling and reconstruction of bandlimited signals with multi-channel time encoding." IEEE Transactions on Signal Processing 68 (2020): 1105-1119.

[23] Levy, Doron. "Introduction to numerical analysis." Department of Mathematics and Center for Scientific Computation and Mathematical Modeling (CSCAMM) University of Maryland (2010): 2-2.

[24] Averbuch, Amir, et al. "Local cubic splines on non-uniform grids and real-time computation of wavelet transform." Advances in Computational Mathematics 43.4 (2017): 733-758.

[25] Thao, Nguyen T., and Dominik Rzepka. "Time encoding of bandlimited signals: reconstruction by pseudo-inversion and time-varying multiplierless FIR filtering." IEEE Transactions on Signal Processing 69 (2020): 341-356.

[26] Ellis, George. Control system design guide: using your computer to understand and diagnose feedback controllers. Butterworth-Heinemann, 2012.

[27] Mohemmed, Ammar, et al. "Training spiking neural networks to associate spatio-temporal input–output spike patterns." Neurocomputing 107 (2013): 3-10.

[28] Willems, Jan C. "From time series to linear system—Part I. Finite dimensional linear time invariant systems." Automatica 22.5 (1986): 561-580.

[29] Phillips, Charles L., et al. Signals, systems, and transforms. Upper Saddle River: Prentice Hall, 2003.

[30] Billings, S. A. "Identification of nonlinear systems using parameter estimation techniques." Institute of Electrical Engineers Conference. 1981.

[31] Narendra, K., and P. Gallman. "An iterative method for the identification of nonlinear systems using a Hammerstein model." IEEE Transactions on Automatic control 11.3 (1966): 546-550.

[32] Wiener, N. (1942). Response of a nonlinear system to noise. Technical report. Radiation Lab MIT 1942. restricted, report V-16. no. 129. (p. 112). Declassified Jul 1946. Published as rep. no. PB-1-58087. US Dept. Commerce.

[33] Wills, Adrian, et al. "Identification of hammerstein–wiener models." Automatica 49.1 (2013): 70-81.

[34] Imai, Hiroshi, and Masao Iri. "An optimal algorithm for approximating a piecewise linear function." Journal of information processing 9.3 (1986): 159-162.

[35] Ruder, Sebastian. "An overview of gradient descent optimization algorithms." arXiv preprint arXiv:1609.04747 (2016).

[36] Björck, Åke. Numerical methods for least squares problems. Society for Industrial and Applied Mathematics, 1996.

[37] Butterworth, Stephen. "On the theory of filter amplifiers." Wireless Engineer 7.6 (1930): 536-541.

[38] Papoulis, Athanasios. "A new algorithm in spectral analysis and band-limited extrapolation." IEEE Transactions on Circuits and systems 22.9 (1975): 735-742.

[39] Xu, Yang, et al. "Spike trains encoding and threshold rescaling method for deep spiking neural networks." 2017 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE, 2017.

[40] Aldroubi, Akram, and Hans Feichtinger. "Exact iterative reconstruction algorithm for multivariate irregularly sampled functions in spline-like spaces: the L^p -theory." Proceedings of the American Mathematical Society 126.9 (1998): 2677-2686.

[41] Florescu, Dorian. Reconstruction, identification and implementation methods for spiking neural circuits. Springer, 2017.

[42] Bing, Zhenshan, et al. "A survey of robotics control based on learning-inspired spiking neural networks." Frontiers in neurorobotics 12 (2018): 35.

[43] Glatz, Sebastian, et al. "Adaptive motor control and learning in a spiking neural network realised on a mixed-signal neuromorphic processor." 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019.

[44] Lazar, Aurel A., and E. A. Pnevmatikakis, "Consistent recovery of sensory stimuli encoded with MIMO neural circuits", Computational intelligence and neuroscience, 2010.

[45] Liu, Junxiu, et al. "Case study: Bio-inspired self-adaptive strategy for spike-based PID controller." 2015 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2015.

[46] Ghosh-Dastidar, Samanwoy, and Hojjat Adeli. "Spiking neural networks." International journal of neural systems 19.04 (2009): 295-308.

[47] Yudin, Nikita Evgen'evich. "Adaptive Gauss–Newton method for solving systems of nonlinear equations." Doklady Mathematics. Vol. 104. No. 2. Moscow: Pleiades Publishing, 2021.

[48] Aldroubi, Akram, and Karlheinz Gröchenig. "Nonuniform sampling and reconstruction in shift-invariant spaces." SIAM review 43.4 (2001): 585-620.

[49] Dominguez-Morales, Juan P., et al. "Deep spiking neural network model for time-variant signals

classification: a real-time speech recognition approach." 2018 International Joint Conference on Neural Networks (IJCNN). IEEE, 2018.

[50] Gerstner, Wulfram, and Werner M. Kistler. Spiking neuron models: Single neurons, populations, plasticity. Cambridge university press, 2002.

[51] Boyd, Stephen, and Leon Chua. "Fading memory and the problem of approximating nonlinear operators with Volterra series." IEEE Transactions on circuits and systems 32.11 (1985): 1150-1161.