



Selection and Optimization of Temporal Spike Encoding Methods for Spiking Neural Networks

Balint, P., Kasabov, N., & Kiss, R. (2020). Selection and Optimization of Temporal Spike Encoding Methods for Spiking Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 31(2), 358-370. Article 8689349. <https://doi.org/10.1109/TNNLS.2019.2906158>

[Link to publication record in Ulster University Research Portal](#)

Published in:
IEEE Transactions on Neural Networks and Learning Systems

Publication Status:
Published (in print/issue): 05/02/2020

DOI:
[10.1109/TNNLS.2019.2906158](https://doi.org/10.1109/TNNLS.2019.2906158)

Document Version
Author Accepted version

General rights

The copyright and moral rights to the output are retained by the output author(s), unless otherwise stated by the document licence.

Unless otherwise stated, users are permitted to download a copy of the output for personal study or non-commercial research and are permitted to freely distribute the URL of the output. They are not permitted to alter, reproduce, distribute or make any commercial use of the output without obtaining the permission of the author(s).

If the document is licenced under Creative Commons, the rights of users of the documents can be found at <https://creativecommons.org/share-your-work/licenses/>.

Take down policy

The Research Portal is Ulster University's institutional repository that provides access to Ulster's research outputs. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact pure-support@ulster.ac.uk

Selection and Optimization of Temporal Spike Encoding Methods for Spiking Neural Networks

Balint Petro^{ID}, Nikola Kasabov^{ID}, *Fellow, IEEE*, and Rita M. Kiss

Abstract—Spiking neural networks (SNNs) receive trains of spiking events as inputs. In order to design efficient SNN systems, real-valued signals must be optimally encoded into spike trains so that the task-relevant information is retained. This paper provides a systematic quantitative and qualitative analysis and guidelines for optimal temporal encoding. It proposes a methodology of a three-step encoding workflow: method selection by signal characteristics, parameter optimization by error metrics between original and reconstructed signals, and validation by comparison of the original signal and the encoded spike train. Four encoding methods are analyzed: one stimulus estimation [Ben’s Spiker algorithm (BSA)] and three temporal contrast [threshold-based, step-forward (SW), and moving-window (MW)] encodings. A short theoretical analysis is provided, and the extended quantitative analysis is carried out applying four types of test signals: step-wise signal, smooth (sinusoid) signal with added noise, trended smooth signal, and event-like smooth signal. Various time-domain and frequency spectrum properties are explored, and a comparison is provided. BSA, the only method providing unipolar spikes, was shown to be ineffective for step-wise signals, but it can follow smoothly changing signals if filter coefficients are scaled appropriately. Producing bipolar (positive and negative) spike trains, SW encoding was most effective for all types of signals as it proved to be robust and easy to optimize. Signal-to-noise ratio (SNR) can be recommended as the error metric for parameter optimization. Currently, only a visual check is available for final validation.

Index Terms—Signal processing, spike encoding, spiking neural networks (SNNs), stimulus estimation, temporal contrast.

I. INTRODUCTION

IN a spiking neural network (SNN), information travels between the processing units in the form of binary spiking events. SNN systems are thus inspired by the information processing solutions of the biological brain. Real world measurements provide analog (continuous or discrete) real-value temporal signals; therefore, it is necessary to implement an encoding method to convert the analog values to spike events to provide input to such systems. This analog-to-spike encoding can compress the data size of the signal considerably [1]

since the spike train is a binary value series. In turn, this binary signal provides fast processing, especially using purpose-built hardware, e.g., SpiNNaker [2]. Ultimately, a correct spike encoding could lead to better information preservation along with input data reduction and compression [1], [3]–[6].

It is extremely important to generate the spike train input to the SNN such that the task-relevant information content of the signal is preserved. The issues here are what information is lost and what is preserved and thus how effective was the encoding. One approach is not to evaluate the effectiveness of the encoding separately but for the whole SNN application, e.g., the chosen encoding is deemed effective if the output of the whole SNN system yields good results, such as good classification accuracy [1]. Another approach is to try and optimize the encoding step by itself. However, the comparison of original and encoded signals is nontrivial: how to compare a binary event series with a continuous signal or calculate some error metric of the differences? At best, one can apply the corresponding decoding algorithm and compare the reconstructed signal with the original.

Each encoding method has a different way of extracting information from the input signal. Selecting the specific encoding method depends on signal characteristics, such as the presence of relevant information in the time or frequency domain, the presence of noise in the data, can the data be shifted or scaled. It is also necessary to understand how the encoding changes the signal characteristics, e.g., does it cut into the frequency spectrum, and can it suppress noise. Furthermore, the type of SNN to be utilized also has to be considered, i.e., what type of input it can accept. After choosing the encoding algorithm, the optimization of the encoding parameters also has to be done to ensure that meaningful spike trains are generated and consequentially, and this meaningfulness needs to be validated. This paper provides a quantitative and qualitative analysis into different temporal spike encoding methods and aims at providing guidelines to the process of selecting and optimizing the spike encoding method. The analysis covers both time and frequency domains as well.

A. Overview of Temporal Spike Encoding Methods

Spike encoding can be based on firing rate [instantaneous average firing rate (AFR)] [7], population rank coding (relative firing time of a population of neurons) [8], or temporal coding (exact timing of individual spikes) [9]. Firing rate encoding resembles biological systems in that cortical electric activity

Manuscript received June 19, 2018; revised October 26, 2018; accepted March 6, 2019. This work was supported by the Ministry of Human Capacities through the Higher Education Excellence Program in the frame of biotechnology research area of Budapest University of Technology and Economics (BME FIKP-BIO). The work of B. Petro was supported by the Erasmus Mundus Action 2—PANTHER Project. (Corresponding author: Balint Petro.)

B. Petro and R. M. Kiss are with the Department of Mechatronics, Optics and Mechanical Engineering Informatics, Budapest University of Technology and Economics, Budapest, Hungary (e-mail: petro@mogi.bme.hu).

N. Kasabov is with the Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, Auckland, New Zealand.

Digital Object Identifier 10.1109/TNNLS.2019.2906158

typically has an oscillatory nature. In population rank coding, neurons each have corresponding receptive fields, and they fire in the order of to what extent an input value belongs to their respective receptive field. Temporal encoding methods utilize the exact timing of each spike which marks a change in the signal value. It is thought that biological neurons apply spike timing as information encoding [3]–[6], [10]. At the same time, temporal encoding is well suited for streaming data encoding and fast processing such as magneto/electroencephalography and electrocardiogram. In this paper, only temporal spike encoding methods are further considered.

As a brief overview, existing temporal encoding methods belong to two groups that take two distinct approaches: temporal contrast [3]–[6], [11] and stimulus estimation [12]. Temporal contrast algorithms track the temporal changes in the signal, and the exact timing of spikes represents these changes, similar to the artificial retina [3], [11]. In principle, the next consecutive signal value is compared to an interval and if the value is outside of the interval, a positive or negative spike is generated accordingly. Temporal contrast encoding was first developed as a hardware implementation to allow for the fast visual information processing and was inspired by the human retina [3]. Temporal contrast methods implemented in the NeuCube SNN framework [13] are threshold-based representation (TBR), step-forward (SF) encoding, and moving-window (MW) encoding. These three methods are analyzed in this paper. Another such encoding method that resembles momentum-like algorithms has been presented recently in [14]; however, the corresponding decoding algorithm has not been demonstrated yet. Temporal contrast algorithms produce a bipolar spike sequence (positive and negative spikes, plus zero).

Stimulus estimation encoding is inspired by the response function of the biological neuron and employs a linear filter to find a unipolar (only positive one and zero) spike train that represents the signal [15]. An analog signal can be constructed from a spike train by convolution with a finite-impulse response (FIR) filter, an idea called spike interval information coding. The analog-to-spike encoding is, therefore, an inverse problem, finding the spike train that when reconstructed to analog values gives a close approximation of the original signal. This inverse process is a “deconvolution” by the same filter; a prominent early implementation of which is the Hough Spiker algorithm (HSA) [15]. To overcome some of the disadvantages of HSA, a modified HSA (mHSA) algorithm had been implemented in the CAM-Brain Machine as presented in [12]. A further improved solution of the inverse problem was implemented as Ben’s Spiker algorithm (BSA) [12]. BSA calculates two error terms that would result from emitting or not emitting a spike at a time point and makes the decision to spike comparing these errors to a threshold. Of these algorithms, this paper considers only BSA, since it was already shown to perform better than HSA and mHSA [12] and BSA encoding had been implemented in SNNs and successfully applied to EEG data [16], [17]. Stimulus estimation encoding produces a unipolar spike sequence (positive spikes and zero).

Temporal contrast and stimulus estimation encoding differ not only in their mechanisms but also in the polarity of

the spike sequence produced (bipolar and unipolar, respectively). Unipolar SNN architectures support only unipolar spike sequences, i.e., the presence or the absence of a firing event at a time point which is transmitted through an excitatory (positive) or inhibitory (negative) connection. Bipolar SNN architectures can support bipolar spike sequences, i.e., positive spikes, negative spikes, and no firing [13]. For example, this can be implemented as changing the state (membrane potential) of the input neuron(s) according to the spike sign. Another approach is that the positive spikes are fed to one input neuron while the negative spikes are fed to another and these neurons are then connected to the rest of the network through positive or negative connections.

B. Related Works on Temporal Spike Encoding Optimization

General studies on SNNs start with the assumption that the encoded spike train is already available as the input to the system. Little has been published on the specific effects of encoding methods on the spike train information content and the reconstructed signal, or in fact on the rationale behind the selection of a particular encoding algorithm. In many cases of application-related studies, the efforts to optimize the encoding are tied to the performance of the whole SNN. For example, the parameters for the chosen encoding method were included in the grid search performed on all other SNN parameters and were evaluated based on the total system performance, e.g., classification accuracy [18]. In this way, the extracted information content was determined by the machine learning process itself without the influence of expert knowledge on the data generation process.

Few are the studies that considered optimizing the encoding step separately. In the seminal paper that introduced BSA [12], the applied FIR filter was a “cleaned-up,” normalized, quantized version of a filter that was found through genetic search algorithms in [19]. The threshold was then optimized for the signal-to-noise ratio (SNR) between the original signal and the noise (i.e., the encoding error between original and reconstructed signals). In [20], BSA was applied to normalized EEG signals; filter design parameters were sought by trial and error, while the applied threshold was found through minimizing the normalized absolute error between original and reconstructed signals. As validation, the signals were compared visually.

Another framework for spiking data encoding related to stimulus estimation was formulated in [21] which aims at maximizing information content while minimizing spike density, i.e., AFR to achieve better data compression. The idea is that if existing knowledge is available about how the data was generated in the examined system, this should be taken into account (via “knowledge injection”) when the optimal encoding method is sought. For stimulus response encoding, the applied convolution function is created based on a model of the signal generation [21]. For example, in fMRI data encoding, a hemodynamic response function can be used to biologically model how neural activity generates fMRI signals [1]. This response function can be modeled as a gamma function that can be employed for stimulus estimation

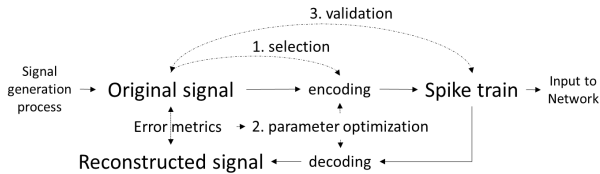


Fig. 1. Proposed encoding workflow.

198 encoding and the filter itself is learned from the data through
 199 genetic algorithms (GAGamma). Parameters are optimized
 200 simultaneously while imposing practical limits on them and on
 201 spike density as well; the optimization was evaluated based on
 202 the root-mean-square error (RMSE) between the signals [21].
 203 As the literature above show, the SNN community has not
 204 settled on any encoding method selection and optimization
 205 methodology with regards to temporal data.

206 The rest of this paper is organized as follows. Section II
 207 introduces the proposed encoding methodology, and
 208 Section III introduces the encoding methods (note the
 209 algorithms in the Appendix) that are employed in this
 210 investigation. Section IV details the investigative methods,
 211 and Section V presents the results and offers discussions.
 212 Finally, Section VI offers concluding remarks.

213 **II. PROPOSED ENCODING METHODOLOGY**
 214 **AND AIM OF STUDY**

215 This paper sets out to provide an analytical background for
 216 encoding method selection and optimization in and of itself,
 217 utilizing a signal reconstruction approach as generated by
 218 the corresponding decoding algorithm. Comparing the original
 219 and reconstructed signals can employ different error metrics,
 220 based on which the encoding parameter(s) can be tuned.
 221 A comparison in time or frequency domain can verify to
 222 what extent the encoding method preserved the information
 223 in the original signal. This is important for classification tasks
 224 where we may have prior knowledge about the nature of useful
 225 information in the signal, e.g., task-specific response signals
 226 or a frequency band of interest. For prediction tasks, decoding
 227 the spike train to real-value signals is crucial in interpreting
 228 the output of the SNN.

229 Based on these remarks, a three-step encoding workflow is
 230 proposed (Fig. 1). First, the encoding method is selected based
 231 on signal characteristics. Second, the encoding parameters
 232 are optimized based on error metrics between reconstructed
 233 and original signals (verification in the time domain). Third,
 234 the encoding is validated by comparing the spike train to the
 235 original signal in the time/frequency domain.

236 The aim of this paper is to provide guidelines for these
 237 three steps, i.e., selection, optimization, and validation of
 238 encoding methods. This warrants an investigation utilizing
 239 well-defined and characteristic test signals as inputs to the
 240 encoding algorithms in question. Specific properties and the
 241 possible optimization of the encoding methods are explored on
 242 test signals, and the implications are discussed. An overview
 243 is provided to aid in the selection of method according to the
 244 signal characteristics.

245 **III. ENCODING AND DECODING**
 246 **ALGORITHMS INVESTIGATED**
 247 **IN THIS RESEARCH**

248 The following encoding and corresponding decoding algo-
 249 rithms are studied here for the selection and optimization of a
 250 suitable spike encoding method and its optimized parameters
 251 for a given task: TBR, SF, MW, and BSA. The algorithms are
 252 described in the following, and pseudocodes are also presented
 253 in the Appendix.

254 *A. Threshold-Based Representation*

255 The simplest implementation of temporal contrast encoding,
 256 TBR [3], [22], is based on tracking temporal changes in the
 257 signal as demonstrated in the artificial retina [3], [11]. The
 258 absolute value change between consecutive signal values is
 259 compared to a threshold; if large enough, a positive/negative
 260 spike is emitted (based on the sign of change). To calculate
 261 this threshold, the whole sample length is taken into account.
 262 The first derivative is calculated; then, the standard deviation
 263 of this derivative is multiplied by a factor to obtain the
 264 encoding threshold (see Algorithm 1 in the Appendix). The
 265 only parameter of this encoding is this factor which is inde-
 266 pendent of the signal amplitude but is determined by the signal
 267 characteristics. Decoding of the signal is straightforward: the
 268 reconstructed signal is given by a summation of positive
 269 and negative spikes multiplied by the encoding threshold
 270 (see Algorithm 5 in the Appendix). The initial reconstruction
 271 value should match the initial signal value.

272 *B. Step-Forward Encoding*

273 The SF encoding [13] utilizes an interval around a mov-
 274 ing baseline with a set threshold (see Algorithm 2 in the
 275 Appendix). The initial baseline equals the initial signal value.
 276 If the next signal value is above/below baseline \pm threshold
 277 value, a positive/negative spike is registered and the baseline is
 278 moved to the upper/lower limit of the threshold interval. The
 279 set threshold is signal amplitude dependent and is the only
 280 parameter of this encoding. The decoding process is essentially
 281 the reconstruction of this moving baseline, similar to TBR (see
 282 Algorithm 5 in the Appendix).

283 *C. Moving Window*

284 The MW encoding [13] uses a moving baseline with a set
 285 threshold value, where the baseline always equals the mean of
 286 the preceding signal values in a time window (see Algorithm 3
 287 in the Appendix). Thus, the moving baseline is essentially the
 288 application of a moving average filter. If the signal value is
 289 above/below baseline \pm threshold value, a positive/negative
 290 spike is registered. MW thus has two parameters: the threshold
 291 and the window size. Decoding is essentially the same as
 292 for TBR or SF (see Algorithm 5 in the Appendix). At this
 293 point, an additional moving average filter could be applied
 294 to make the reconstructed signal smoother, in which case the
 295 encoding–decoding corresponds to a two-pass (twice applied)
 296 moving average filter.

297 D. Ben's Spiker Algorithm

298 An analog signal can be constructed from a spike train
 299 by convolution with an FIR filter. BSA is an algorithm for
 300 producing the spike train from which the original signal can
 301 be reconstructed well [12] (see Algorithm 4 in the Appendix).
 302 BSA works only for positive-valued signals. First, an FIR
 303 filter is created. Then, two error terms are calculated at
 304 each time point: one that results from subtracting the filter
 305 coefficients from the subsequent signal values, and one that
 306 results from not changing the signal (no subtraction). If the
 307 subtraction error is smaller than the unchanged signal error
 308 term minus a threshold, a positive spike is generated and the
 309 filter coefficients are subtracted from the signal. Decoding is
 310 straightforward since it was kept in mind during the encoding:
 311 a convolution of the spike train with the filter coefficients gives
 312 the reconstructed signal (see Algorithm 6 in the Appendix).
 313 BSA encoding results in a unipolar (only positive) spike train.

314 The original BSA encoding requires input with [0, 1]
 315 limits. However, BSA can be applied to any positive-valued
 316 signal if the filter coefficients are scaled up such that they
 317 appropriately match the signal boundaries. Therefore, a simple
 318 signal shift above zero is sufficient. The recommended scaling
 319 of coefficients is discussed in Section V. (part E/2).

320 IV. METHODS OF INVESTIGATION

321 A. Test Signals Used for Quantitative Analysis

322 A number of signal types were considered as test signals,
 323 such as purely step-wise signals, smooth (sinusoidal) signals,
 324 signals with trends, and event-like signals. These signal types
 325 can model a wide variety of important behaviors such as
 326 sudden or smooth changes, slopes and plateaus, trend effects,
 327 different amplitude events, and important frequency spectra.
 328 The rationale behind using sinusoidal signals is that from a
 329 modeling standpoint, measured EEG signals are comprised
 330 a mixture of sinusoidal waves plus multisource Gaussian
 331 noise [23]. The signal parameters were randomly generated,
 332 and to analyze noise effects, random white noise was added.

333 The following test signals were constructed with the length
 334 of 1000 samples to test the properties of encoding methods:

- 335 1) step-wise signal with increasing step size without noise
336 [Fig. 2(a)];
- 337 2) smooth signal with sine components continuously rang-
338 ing from 2 to 20 Hz with random power, combined with
339 random phase lags plus white noise [Fig. 2(b)];
- 340 3) trended signal, the same smooth signal as before multi-
341 plied by an exponential saturation trend plus white noise
342 [Fig. 2(c)];
- 343 4) event-like signal, resembling EEG signals during
344 perturbation-evoked potential events [24] plus white noise
345 [Fig. 2(d)].

346 B. Properties to Investigate

347 In order for the analysis to allow a comparison between
 348 the encoding methods, the following behaviors/properties were
 349 investigated for each encoding method (where applicable):

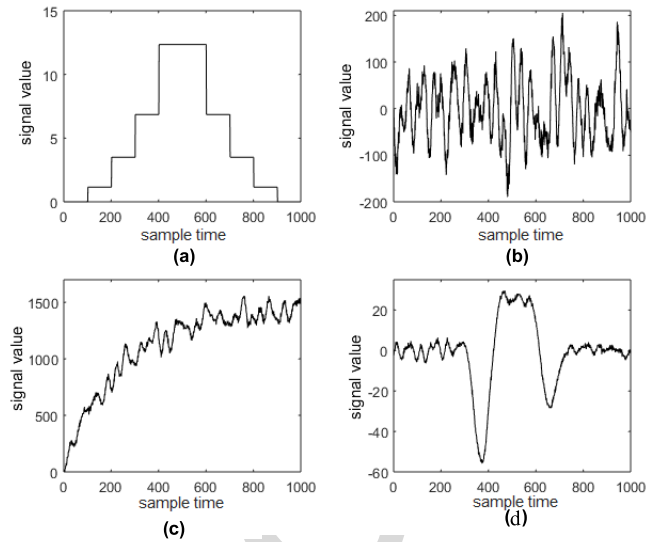


Fig. 2. Test signals used for analysis. (a) Step-wise signal. (b) Smooth signal. (c) Trended smooth signal. (d) Event-like signal.

- 350 1) ability to follow various test signals, and offset
351 and scaling error in time domain (qualitatively and
352 quantitatively);
- 353 2) false encoding at start and/or end of signal;
- 354 3) frequency characteristics such as noise suppression, arti-
355 facts, white noise, and spink noise introduced during
356 signal reconstruction;
- 357 4) parameter dependencies;
- 358 5) optimization curves, robustness, and methods.

359 It is important to note that both time and frequency domain
 360 effects were included in the analysis because in many applica-
 361 tions although the signal is in the time domain, the important
 362 information is in the frequency spectrum, even if it is not the
 363 spectrum that is to be encoded per se.

364 C. Error and Indicator Metrics

365 The optimization criterion considered here is the accurate
 366 recovery of the signal; minimizing the difference between
 367 original and reconstructed signals serves as the objective
 368 function. There are multiple candidate error metrics since
 369 there is no consensus on which one to use. In this paper,
 370 the following optimization criteria were used:

- 371 1) SNR;
- 372 2) RMSE;
- 373 3) coefficient of regression (*R*-squared).

374 SNR is defined here as the signal-to-noise ratio where the
 375 difference between the original (*s*) and reconstructed (*r*) signal
 376 is considered as “noise” [12]. SNR is to be maximized and is
 377 calculated as

$$378 \text{SNR} = 20 \cdot \log \frac{\text{Power}(s)}{\text{Power}(s-r)} [\text{dB}]. \quad (1)$$

379 A negative SNR means that the error introduced through
 380 the encoding is more substantial than the information content
 381 itself; SNR of 0 dB means equality between the two.

RMSE is to be minimized and is defined classically as

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^N (r_t - s_t)^2}{N}} \quad (2)$$

where s is the original and r is the reconstructed signal and gives a summation of the modeling errors.

The R -squared, although classically used for measuring regression fit, in a broader sense is a measure of modeling fit in general and it is employed here in this capacity. The R -squared gives a relation between the variance unexplained by the modeling (SS_{res}) and the variance of the original input (SS_{tot}) data

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} \quad (3)$$

$$SS_{\text{res}} = \sum_{t=1}^N (r_t - s_t)^2 \quad (4)$$

$$SS_{\text{tot}} = \sum_{t=1}^N (s_t - \bar{s})^2. \quad (5)$$

R -squared is to be maximized. Note that in this case, R -squared can assume negative values as well; a negative R -squared means that the fit is worse than substituting the mean of the signal (in which case R -squared equals 0).

As an indicator metric, AFR shows how saturated the spike train is and is calculated as the quotient of all spikes (sp) in a given spike train and the number of total time points

$$\text{AFR} = \frac{\sum_{t=1}^N |sp_t|}{N}. \quad (6)$$

V. QUANTITATIVE, QUALITATIVE, AND COMPARATIVE ANALYSIS OF DIFFERENT ENCODING METHODS AND OPTIMIZATION OF THEIR PARAMETERS

A. Numerical Performance of Encoding Methods

To present numerical performance for future reference, best error metric values that could be achieved by the optimization of each encoding method are presented in Table I for the test signals. In the following sections, results from the individual encoding methods are analyzed.

B. Threshold-Based Representation Encoding

TBR registers large enough signal changes only; thus, it is expected that small, gradual changes are not represented. In addition, tracking only signal change causes that sudden, step-wise changes will be poorly represented (Table I) since the step size of the reconstruction is uniform as it equals the threshold. Dynamics of smoothly changing signals are followed (Table I); however, the uniform steps introduce a scaling error which is prominent in the case of trended signals [Fig. 3(a)]. For small and large amplitude events, the selected encoding threshold determines the captured event type [Fig. 3(b)]; there is clearly a tradeoff between representing small and large events. Since the whole sample length is considered for threshold value calculation, this encoding can be disadvantageous for long samples where there may be amplitude differences between events of different parts of the sample. As an advantage, there are no falsely registered

TABLE I

NUMERICAL PERFORMANCE EVALUATION OF THE STUDIED ENCODING METHODS WITH OPTIMIZED PARAMETERS FOR THE TEST SIGNALS

signal	metric	TBR	SF	MW	BSA
step-wise	SNR	7.77	21.79	22.57	11.09
	RMSE	2.67	0.53	0.49	1.82
	R-sq.	0.64	0.99	0.99	0.83
smooth	SNR	2.74	13.47	0.03	10.23
	RMSE	49.89	14.55	68.40	61.42
	R-sq.	0.47	0.96	0.01	0.20
trended smooth	SNR	9.64	38.22	27.40	12.40
	RMSE	368.61	14.94	50.00	281.19
	R-sq.	-0.14	1.00	0.98	0.40
event-like	SNR	5.88	26.24	9.43	10.44
	RMSE	8.66	0.83	5.75	17.31
	R-sq.	0.74	1.00	0.89	-0.03

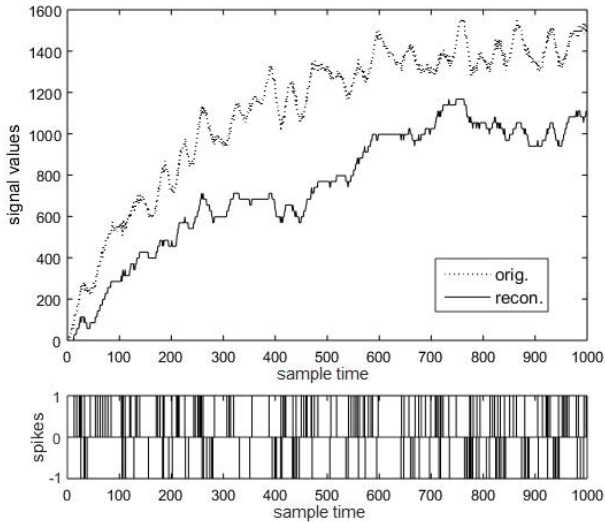
values at the start or end of neither the spike train nor the reconstructed signal.

TBR reduces white noise to a certain extent by applying the threshold to small perturbations in the signal. At the same time, it is sensitive to the presence of strong white noise in the signal, since the spurious signal changes due to white noise cover the more gradual changes. As observed, the presence of white noise causes TBR to introduce strong $1/f$ noise (“pink noise”) during reconstruction; strong low-frequency artifact components appear. For longer signals, this may cause the reconstructed signal to drift away. Parameter optimization is not straightforward for TBR because typically, multiple minima and wider plateaus can be observed due to different amplitude events, even in the case of a continuous, not event-specific signal (Fig. 4). Selecting a certain threshold results in an encoding that better represents events having amplitudes that correspond with the threshold. As observed, all three metrics give similar optimization curves (remember that SNR and R -squared are to be maximized) (Fig. 4).

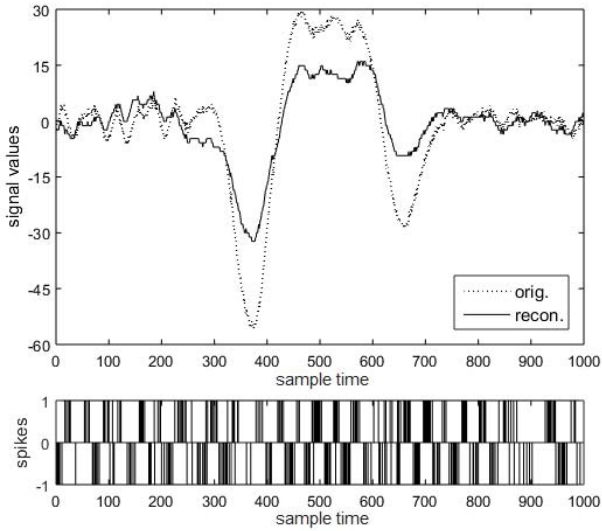
In summary, TBR encoding had been developed to quickly process streaming, online data; the computation is simple and fast. In contrast, the threshold parameter determines the amplitude of the events that are represented correctly by the spike sequence. Therefore, it is encouraged that for each application, the possible events that can appear in the signal are considered and the threshold is chosen such that the relevant events are captured. This knowledge should guide the parameter optimization since multiple peaks can usually be observed, e.g., selecting a higher threshold if higher peaks are of interest in the given application.

C. Step-Forward Encoding

For SF encoding, even though the reconstructed signal is step-wise, it follows most types of continuous signals exceptionally well (Table I) both in time and frequency domains since multiple steps are allowed to account for a single change in the original signal. Step-wise, smooth [Fig. 5(a)], and trended signals [Fig. 5(b)] are all followed well (Table I).

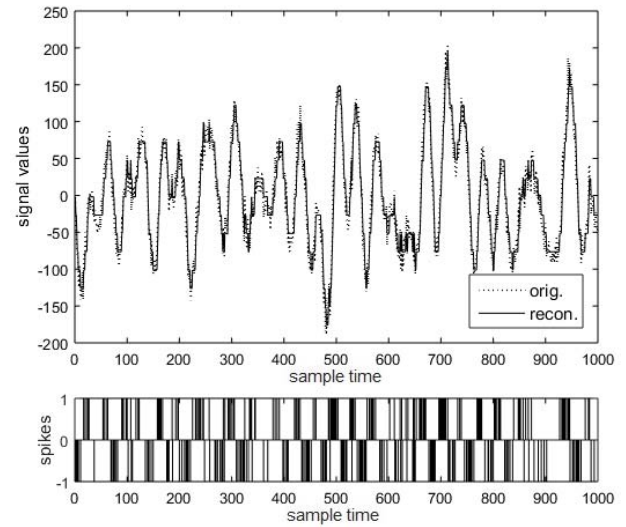


(a)

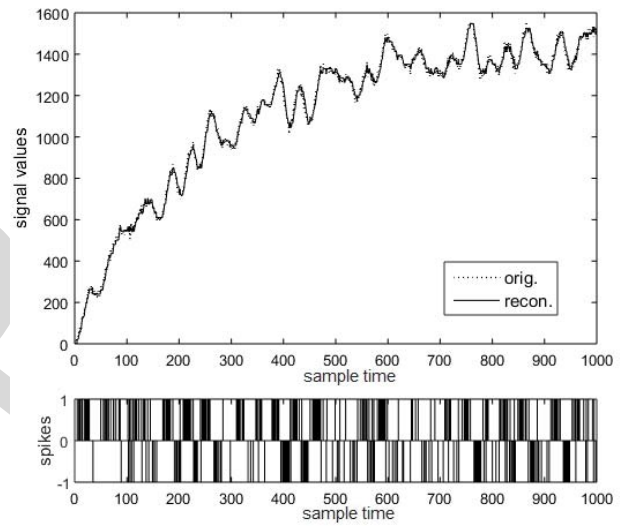


(b)

Fig. 3. (a) Scaling error of a trending signal with TBR encoding. (b) TBR-generated spikes show that TBR cannot represent different amplitude events well simultaneously.



(a)



(b)

Fig. 5. (a) Smooth signal is followed well by using SF encoding. (b) Spike sequence generated by SF represents a trending signal well.

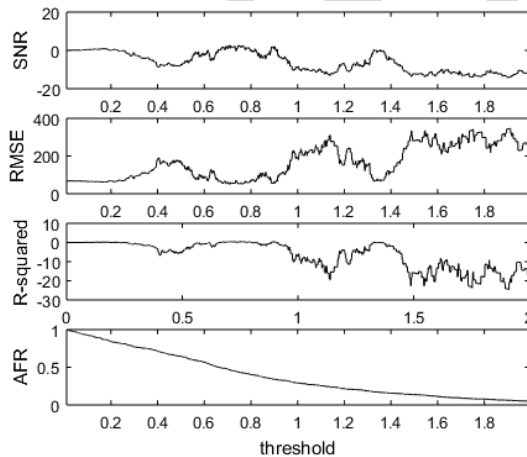


Fig. 4. Threshold optimization curves for TBR (smooth signal).

466 The threshold can be chosen such that small and large
 467 amplitude events are both well represented. Similar to TBR,
 468 SF encoding represents only signal changes and thus offset

error is expected to be present in the reconstructed signal
 unless the initial values are matched. Noise in the signal is
 minimally reduced by the threshold.

SF encoding reconstructs the frequency spectrum as is;
 in addition, it introduces no artifact frequency components
 and only minimal noise that is related to quantization. For
 noisy input signals, the noise is minimally reduced but mostly
 encoded; however, this does not lead to $1/f$ noise as was the
 case with TBR, which is favorable.

Due to the moving baseline, the reconstructed signal does
 not drift away even for longer signals. Overshoot does not
 occur with SF since the moving baseline is adjusted only
 by the threshold value; thus, the change is equal or less
 than the signal change. The optimization curves match for
 all three metrics and show a wide, high fit plateau (Fig. 6).
 The wide plateau means that the threshold can be increased,
 and thus spike density can be lowered without a significant
 loss of encoding accuracy, which can be favorable in certain

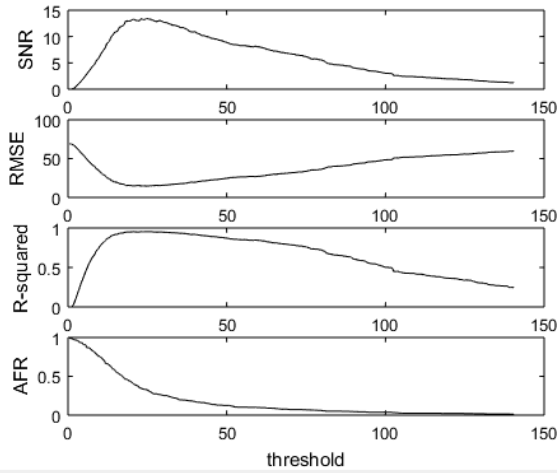


Fig. 6. Threshold optimization curves for SF with a high good fit plateau (smooth signal with noise).

487 applications or cases. Furthermore, a lower AFR further
 488 enhances data compression and helps to prevent the saturation
 489 of SNN. Lowering the AFR also improves noise suppression
 490 but magnifies the quantization noise, especially in the lower
 491 frequency ranges.

492 In summary, SF encoding performed well for all types of
 493 test signals (Table I), both in time and frequency domains with
 494 straightforward and robust optimization. SF encoding can thus
 495 be recommended for any applications, especially when there
 496 is little information on the nature of the original signal.

497 D. Moving-Window Encoding

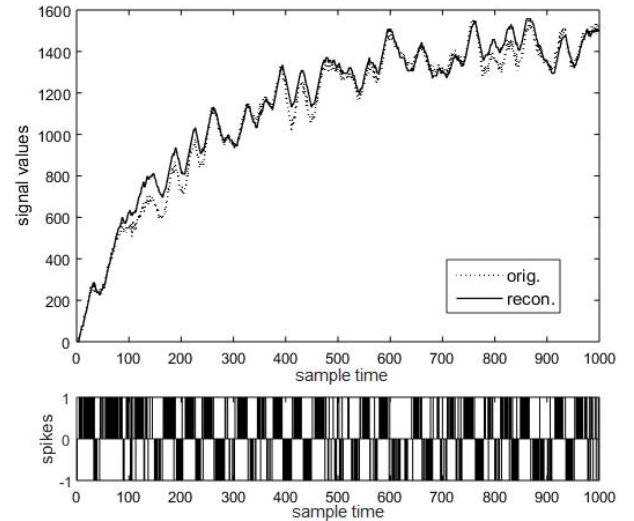
498 1) *Theoretical Considerations of MW Encoding*: The MW
 499 encoding was suggested to be robust against noise [13]. The
 500 signal values are compared to a moving average, which acts
 501 as a moving average filter. Moving average is an optimal
 502 smoothing filter in the time domain against white noise,
 503 while a poor low-pass filter in the frequency domain [25].
 504 The frequency response function of moving average filters of
 505 windows size M is the sinc function

$$506 \quad H(f) = \frac{\sin(\pi f M)}{M \sin(\pi f)}. \quad (7)$$

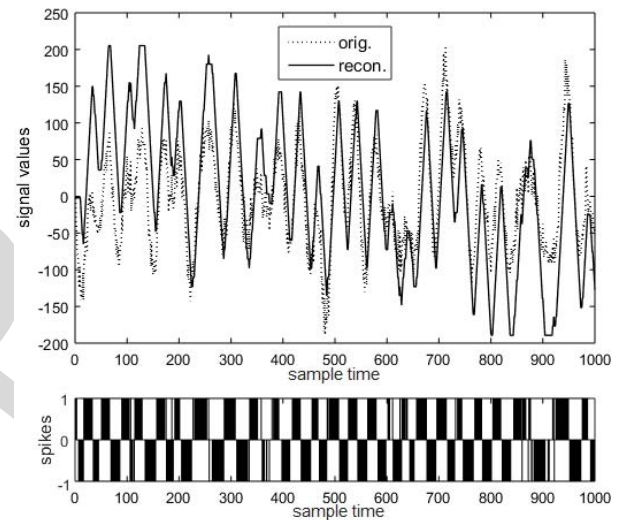
507 The window size influences the cutoff frequency, the slope
 508 of the frequency response, and the noise reduction. The cutoff
 509 frequency at -3 dB is approximately given by

$$510 \quad f_{co} = 0.8859 \cdot \sqrt{M^2 - 1} \left[\frac{\pi \cdot \text{rad}}{\text{sample}} \right]. \quad (8)$$

511 The noise suppression is proportional (equal in amplitude)
 512 to the square root of the window size. This means a tradeoff
 513 between noise reduction and the spectrum width retained dur-
 514 ing the encoding. Another implication is that the MW encod-
 515 ing in principle could be used to filter out a strong, artifact
 516 frequency component, e.g., power line noise in EEG signals.
 517 However, as it was pointed out, the band-stop characteristics
 518 of this filter are disadvantageous as the attenuation is rather
 519 shallow. Thus, it is recommended to apply a separate digital



(a)



(b)

Fig. 7. (a) MW encoding follows a trended, smooth signal well. (b) Noise and sharp peaks are reduced using MW encoding.

520 band-stop filter to remove line noise before encoding the signal
 521 altogether. An important point is that the signal beginning is
 522 not encoded well until the window size is reached. This can
 523 be managed with a slight modification of the algorithm: for
 524 the first M points, the baseline can be set to the mean of these
 525 M points.

526 2) *Quantitative Results of MW Encoding*: Similar to SF,
 527 MW follows sharp steps well (Table I). Trended [Fig. 7(a)],
 528 smooth [Fig. 7(b)], and event-like signals are represented well
 529 (Table I), but overshoot-type errors often appear. Interestingly,
 530 error metrics indicate a poor match for the smooth signal
 531 despite that visually, the signal dynamics appear to be well
 532 captured.

533 As stated before, MW encoding reduces white noise. How-
 534 ever, the $1/f$ (pink) noise also appears during signal recon-
 535 struction, the amount of which appears to be proportional
 536 to the suppressed components in the spectrum above cutoff
 537 frequency. For longer signals, this may cause the reconstructed
 538 signal to drift away.
 539

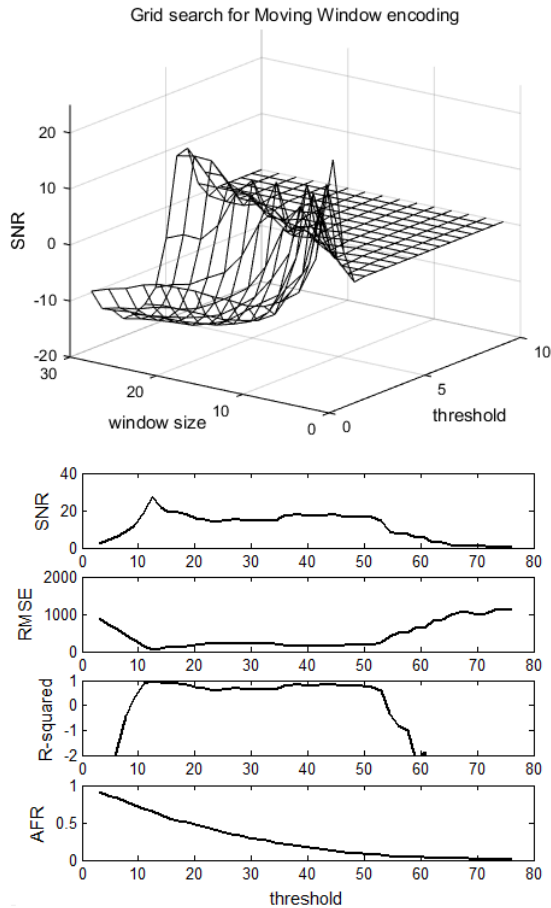


Fig. 8. Grid search for MW parameters and optimization curves (noisy trended smooth signal).

MW encoding has two parameters: window size and threshold. The window size affects the encoded spectrum width and noise suppression and thus should be chosen first. Then, the threshold can be optimized according to any of the error metrics (Fig. 8). SNR is favorable here since negative values clearly indicate prohibited areas. However, at a low threshold (close to continuous firing), the peak SNR can lead to a false threshold selection.

In sum, MW encoding was conceived to provide the robustness against noise by what is essentially a moving average filter. However, this noise suppression can only increase with the window size which in turn lowers the cutoff frequency. This may not be permissible in some applications, e.g., for EEG data that is not sufficiently oversampled. In such cases, it is recommended to use a separate digital filter as a preprocessing step instead and applying a spike encoding other than MW.

E. Ben's Spiker Algorithm Encoding

1) *Theoretical Considerations of BSA Encoding*: In the original BSA implementation, the error threshold is an exact value and is subtracted [12]. However, this can effectively be replaced by a multiplication with a factor smaller than one. This method was implemented for this investigation. Such a threshold value is scale invariant; thus, the encoding becomes

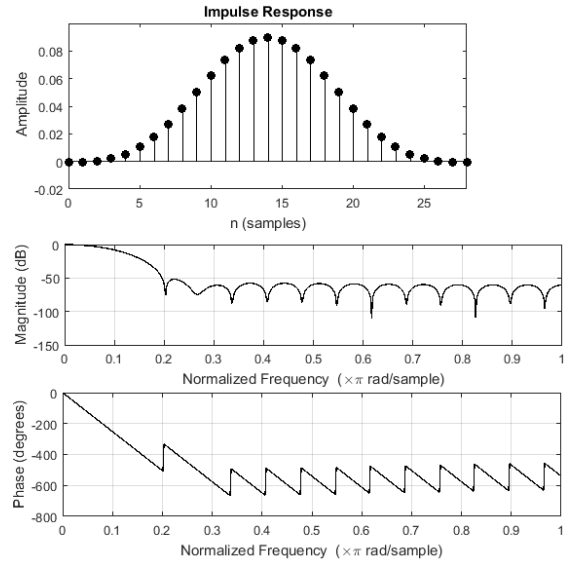


Fig. 9. Exemplar filter coefficients and frequency response.

more robust. For BSA, scaling of the filter coefficients provides a reconstruction boundary: the maximum value that can be reached (by constant firing) is the sum of coefficients, while the minimum value equals 0 (if all coefficients are nonnegative) or a small negative value.

Filter design is based on at least two parameters, e.g., a low-pass filter with cutoff frequency (f_{co}) and filter size (Fig. 9). Design of the filter is crucial since the signal components that can be represented are determined by the applied filter [12]. If a low-pass filter was used with cutoff frequency $f_{co} = 0.05$, the SNR of the reconstructed signal will sharply fall to 0 dB above this frequency. This means that the cutoff frequency is to be chosen based on the task-relevant frequency spectrum. The number of coefficients gives the width of the filter and determines the sharpness of the frequency response function. With regards to scaling, it should be sufficient in principle that the coefficients are scaled such that the reconstruction boundary equals the signal boundary. However, this limits the dynamic characteristics which the reconstructed signal can take. During the implementation steps of this investigation, several values were experimented with. It was found that if the sum of coefficients (the upper boundary of the reconstructed signal) is scaled up to twice the original signal upper boundary, the spike train is less saturated and the encoding is more flexible.

2) *Quantitative Results for BSA Encoding*: For BSA, the deconvolution algorithm works only for positive-valued signals [12]; thus, the signals need to be shifted to the positive range. BSA encoding favors a continuously changing signal; these are represented (e.g., smooth signal, Fig. 10) and reconstructed well (Table I), even with trends in the signal. At the same time, BSA has a weakness for plateaus since firing with a constant rate is required to represent a nonzero constant value; without spikes, the reconstructed signal does not hold but quickly returns to 0. This weakness for constant values can lead to critical, conceptual failures for sharp step

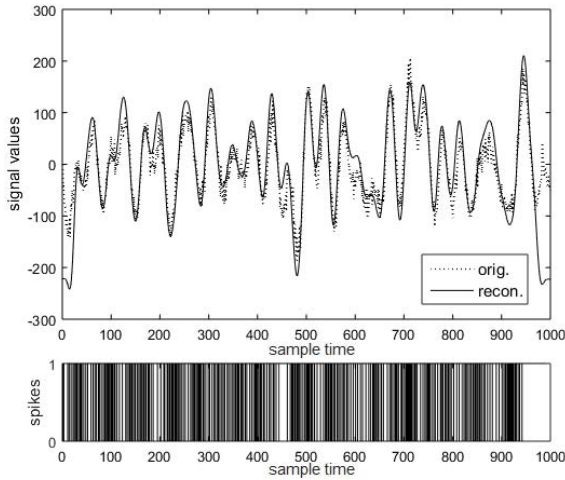


Fig. 10. Reconstruction of a smooth signal with well-scaled BSA filter coefficients with the corresponding spike sequence.

598 signals and signals with plateaus in general. Higher plateaus
 599 (in the upper half of the signal range) can be particularly
 600 problematic, especially if the filter coefficients are not scaled
 601 up appropriately [Fig. 11(a)]. Here, this effect is demonstrated
 602 through scaling the coefficients such that their sum equals the
 603 maximum amplitude of the signal (“poorly scaled” case) and
 604 to equal twice the maximum (“well-scaled” case). In addition,
 605 BSA in general tends to show offset and scaling error in the
 606 reconstructed signal, depending on the optimization.

607 Another implication is that there are false spikes and false
 608 reconstructed values at the start of the signal since the value
 609 has to catch up from 0. This means that the data should be
 610 shifted to have a minimum of 0 to improve the BSA encoding.
 611 The end of the signal is also falsely encoded due to the
 612 convolution; the number of erroneous points corresponds to
 613 the filter size. Depending on the sample length, substantial
 614 loss of information may occur here.

615 In the frequency domain, the spectrum (Fig. 12) is changed
 616 similarly as if the FIR filter were applied to the original signal:
 617 frequencies above the cutoff point are suppressed (see Fig. 9).
 618 In effect, the encoding performs filtering at the same time.
 619 However, some artifact components with low frequencies can
 620 appear, especially if the filter size is large.

621 The first step of parameter selection and optimization for
 622 BSA is the filter design which is aimed at retaining the
 623 task-relevant frequency spectrum. It is suggested that cutoff
 624 frequency and filter size are selected jointly since these
 625 determine the frequency response together. A grid search
 626 or other optimization can be performed to determine the
 627 highest encoding fit for possible cutoff size (Fig. 12). SNR
 628 is recommended as the error metric of the search since
 629 negative values indicate forbidden areas. Other optimization,
 630 e.g., genetic or differential evolving algorithms, could also be
 631 applied. As an initial guess, cutoff could be selected at twice
 632 the highest important frequency with a filter size of 20–24.
 633 It can be observed that the solution across the search space
 634 is not smooth (Fig. 12) as there are multiple local peaks. The
 635 resulting filter coefficients and especially the cutoff frequency

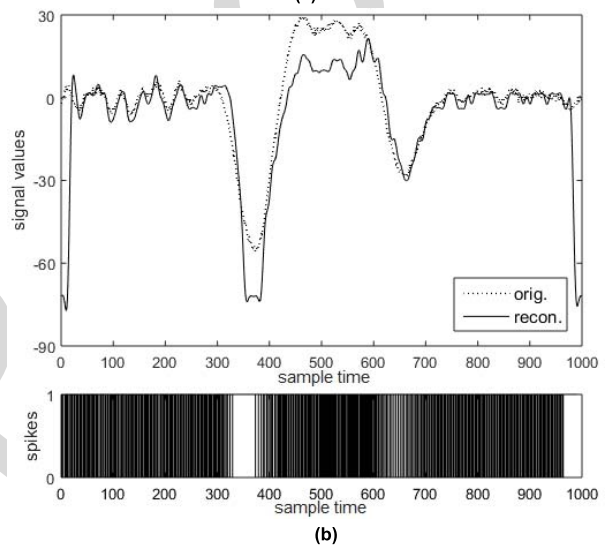
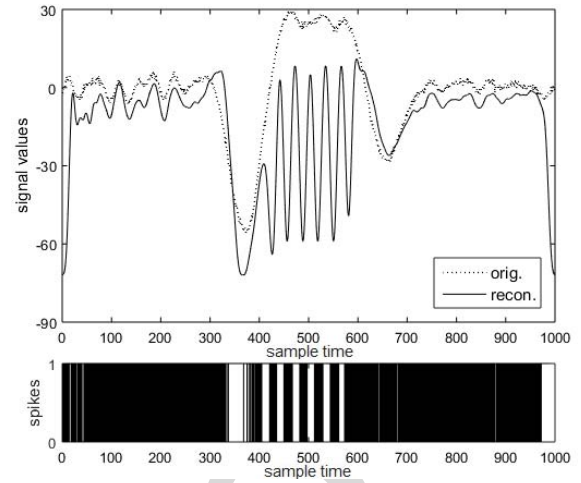


Fig. 11. (a) Error at high plateaus with poorly scaled coefficients of BSA encoding. (b) High plateau error disappears with well-scaled coefficients.

636 must be cross-checked against the signal’s relevant frequency
 637 spectrum. With increased size, the filtering is sharper; a
 638 limitation on the filter size is the false encoding and decoding
 639 at the start and end of the signal due to the convolution. For the
 640 optimal error threshold parameter corresponding to a set filter,
 641 all error metrics give the same results (Fig. 13). For many
 642 signals, a (multiplicative) threshold in the range of 0.94–0.98
 643 provides a good solution.

644 In sum, BSA encoding is the only one of the four encod-
 645 ings analyzed here that produces a unipolar (only positive)
 646 spike train and thus may be the only option for some SNN
 647 architectures. BSA produces major errors for suddenly chang-
 648 ing, step-like signals and also has problems with plateaus,
 649 especially in the higher value ranges. The filter design and
 650 optimization are also nontrivial. Using a multiplicative error
 651 threshold provides a robust solution, but the effects of this
 652 algorithmic modification need further analysis. The false
 653 encoding start and end are also of concern; padding the signal
 654 with constant values might address this issue to a certain
 655 extent.

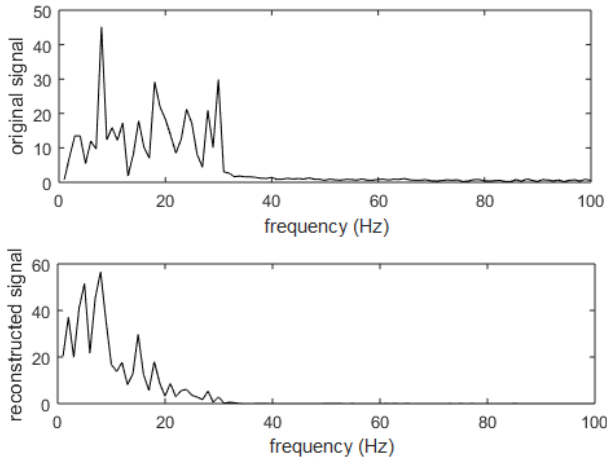


Fig. 12. Frequency spectrum effects of BSA encoding: single-sided amplitude spectrum is filtered in effect and pink noise appears.

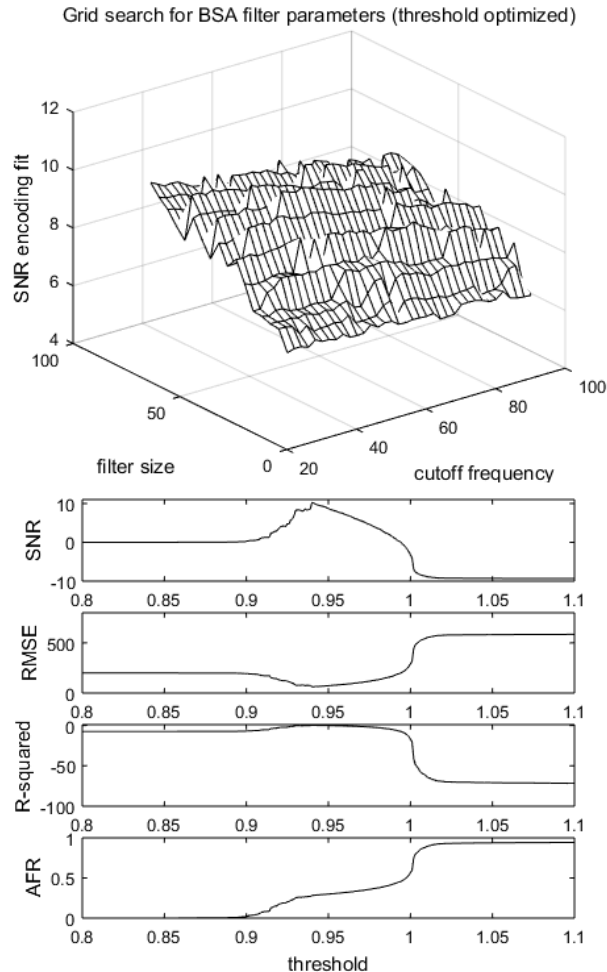


Fig. 13. Grid search for BSA filter parameters, threshold optimization curves for a selected filter (smooth test signal).

F. Comparison of Encoding Methods

A comparison of the key characteristics of analyzed encoding methods is given in Table II to aid with the first step of the encoding process, i.e., selecting the method. The first and foremost selection criterion is whether the SNN architecture

TABLE II

SUMMARY AND COMPARISON OF ENCODING METHOD CHARACTERISTICS

Methods	BSA	TBR	SF	MW
spike train polarity	+	+/-	+/-	+/-
false encoding at start/end	yes, both	no	no	yes, at start
cuts into frequency band	yes	no	no	yes
suppresses (white) noise	yes, greatly	yes, little	yes, little	yes, greatly
optimization	non-trivial	non-trivial	easy	non-trivial

accepts only positive or bipolar spike trains as well. Of the analyzed methods, BSA was the only one producing unipolar spikes (Table II). BSA is not suitable to encode step-like signals; continuously changing, constant mean signals are represented most efficiently while a trend in the signal causes some scaling error. Signals with high plateaus are most difficult to encode with BSA. Noise suppression above the cutoff frequency is excellent. However, it must be stressed that the filter design has to be congruent with the task-relevant frequency spectrum and there is a single peak optimum threshold value for a set filter.

Comparing the three temporal contrast methods, SF proved to be very effective for all test signals (Table I) with the further advantage of having a single parameter with a wide, high fit optimization plateau (Table II). In addition, SF does not suffer from systematic errors other than offset in the reconstruction and even suppresses white noise to a small degree. Thus, SF is recommended as a universal encoding method. In the case of a very noisy raw signal, choosing MW encoding could be justified. However, this moving average filtering cuts into the frequency spectrum and the reconstructed signal suffers from pink noise, artifact components, and often scaling error as well (Table II). Thus, a preprocessing step with a digital filter and SF encoding is recommended instead. The numerous disadvantages (Table II) of TBR mean that it should be used only when necessary, e.g., for the simulation, development or deployment of hardware implementations.

G. Observations on the Error Metrics

The optimization curves for the test signals showed that all three metrics give the same optimization curve (Figs. 4, 6, 8, and 13). To account for this, consider that R -squared and RMSE are both calculated based on the fitting error variance while SNR calculates the power ratio of original signal and fitting error. Furthermore, it was assumed that the initial signal value is available for the decoding step; thus, it was possible to match the initial original and reconstructed signal values. If this was not the case, that would mean strong implications for the optimization step. The error metrics would

699 yield different curves and thus different optimal parameter
 700 values. The R -squared optimization works to represent the
 701 signal dynamics as well as possible (even with a constant offset
 702 error), while SNR aims to diminish the difference altogether
 703 such that the signal values will match. This often leads to false
 704 encoding–decoding at the start of the signal. RMSE would still
 705 yield the same results as SNR.

706 The question of which error metric to choose for optimiza-
 707 tion arises. SNR can be interpreted based on its sign; it has to
 708 be positive for meaningful signal reconstruction. Therefore,
 709 SNR is recommended because of favorable interpretability.
 710 For computational load, RMSE is favorable, but note that
 711 the absolute value of RMSE obtained will differ for different
 712 inputs as there are no broadly accepted ways to normalize
 713 RMSE, making comparisons across signals difficult.

714 The low-frequency erroneous components in the recon-
 715 structed signal may cause a significant drift away from the
 716 original signal. Although the final output of the encoding
 717 optimization is the resulting spike train, this drift is of interest
 718 because such a great difference between the two signals
 719 heavily influences the error metrics. It is possible that applying
 720 boundaries to the reconstruction could address this issue to
 721 some extent.

722 *H. Observations on the Validation Step*

723 As outlined in the three-step encoding workflow, a final
 724 validation step is required to check that task-relevant, mean-
 725 ingful information is retained in the spike train. Currently, a
 726 visual check is the only available method for this step. The
 727 original signal can be visually compared with the reconstructed
 728 signal or the spike train itself. Frequency spectra can also
 729 be compared, especially for a sinusoid signal in which the
 730 frequency components are of particular interest (Fig. 12).
 731 Another option for validation would be testing the accuracy of
 732 the whole SNN application. However, this would be influenced
 733 by the many parameters of the SNN as well.

734 *I. Observations on Biological Plausibility*

735 It could be of interest to consider the biological rationale of
 736 different encoding methods. As mentioned previously, BSA as
 737 a neuron stimulus estimation method has a strong biological
 738 plausibility considering the functioning of individual neurons.
 739 As for the temporal contrast encodings, these mostly rely on
 740 the biological strategy that only (large enough) changes of the
 741 important property are registered to improve robustness and
 742 energy efficiency. TBR registers only large changes between
 743 consecutive values, much like retinal cells [3]. SF does the
 744 same, but the basis of comparison is the previously registered
 745 large change that caused a spike and not necessarily the
 746 previous signal value; in this way functioning as a short-term
 747 memory or adaptation. MW takes a moving average of pre-
 748 vious signal values as a baseline to improve robustness, for
 749 which the biological rationale is hard to determine.

750 *J. Limitations of This Investigation*

751 To limit the scope of investigation, the analyzed signals
 752 were all 1000 samples long. A future study may address

Algorithm 1 TBR Encoding

```

1: input:  $s$  signal,  $f$  factor
2: startpoint =  $s(1)$ 
3:  $diff = \text{zeros}(\text{length}(s))$ 
4: for  $t = 1:(\text{length}(s)-1)$ 
5:    $diff(t) = s(t+1) - s(t)$ 
6: end for
7:  $diff(\text{end}) = diff(\text{end}-1)$ 
8: threshold =  $\text{mean}(diff) + f \cdot \text{std}(diff)$ 
9:  $out = \text{zeros}(\text{length}(s))$ 
10: for  $t = 1:\text{length}(s)$ 
11:   if  $diff(t) > \text{threshold}$ 
12:      $out(t) = 1$ 
13:   elseif  $diff(t) < -\text{threshold}$ 
14:      $out(t) = -1$ 
15:   end if
16: end for
17: output:  $out$ 

```

Algorithm 2 SF Encoding

```

1: input:  $s$  signal,  $threshold$ 
2: startpoint =  $s(1)$ 
3:  $out = \text{zeros}(\text{length}(s))$ 
4:  $base = s(1)$ 
5: for  $t = 2:\text{length}(s)$ 
6:   if  $s(t) > base + threshold$ 
7:      $out(t) = 1$ 
8:      $base = base + threshold$ 
9:   elseif  $s(t) < base - threshold$ 
10:     $out(t) = -1$ 
11:     $base = base - threshold$ 
12:   end if
13: end for
14: output:  $out$ ,  $startpoint$ 

```

753 effects of the sample size, e.g., short (50–100) or long
 754 (5000–10000) samples. The momentum-like TBR algorithm
 755 introduced in [14] was not considered because the corre-
 756 sponding decoding method has not been demonstrated. The
 757 GAGamma methodology [21] is based on the knowledge about
 758 the signal generation, and this study is aimed at evaluation
 759 different encoding methods for a wide variety of signals;
 760 GAGamma was not included in our analysis. Another lim-
 761 itation was that only individual samples were included in
 762 the encoding parameter optimization. Future work is to be
 763 carried out with regards to optimizing multiple sample data
 764 and furthermore, data with multiple features.

765 VI. CONCLUSION

766 Any machine learning process can be effective and valid
 767 only if the input data contain the relevant information in a
 768 meaningful representation. For SNNs, this input format is
 769 a unipolar or bipolar spike event sequence (spike trains).
 770 Encoding real-valued data (signals) to spike trains, therefore,
 771 has to retain the task-relevant information with as little artifacts

Algorithm 3 MW Encoding

```

1: input:  $s$  signal,  $threshold$ ,  $window$ ,  $startpoint$ 
2:  $startpoint = s(1)$ 
3:  $out = \text{zeros}(\text{length}(s))$ 
4:  $base = \text{mean}(s(1:window + 1))$ 
5: for  $t = 1:(window + 1)$ 
6:   if  $s(t) > base + threshold$ 
7:      $out(t) = 1$ 
8:   elseif  $s(t) < base - threshold$ 
9:      $out(t) = -1$ 
10:  end if
11: end for
12: for  $t = (window+2):\text{length}(s)$ 
13:    $base = \text{mean}(s(t-window-1:t-1))$ 
14:   if  $s(t) > base + threshold$ 
15:      $out(t) = 1$ 
16:   elseif  $s(t) < base - threshold$ 
17:      $out(t) = -1$ 
18:   end if
19: end for
20: output:  $out$ ,  $startpoint$ 

```

Algorithm 4 BSA Encoding

```

1: input:  $s$  signal,  $fir$ ,  $threshold$ 
2:  $L = \text{length}(s)$ 
3:  $F = \text{length}(fir)$ 
4:  $out = \text{zeros}(L)$ 
5:  $shift = \min(s)$ 
6:  $s = s - shift$ 
7: for  $t = 1:(L-F)$ 
8:    $err1 = 0$ 
9:    $err2 = 0$ 
10:  for  $k = 1:F$ 
11:     $err1 = err1 + \text{abs}(s(t+k)-fir(k))$ 
12:     $err2 = err2 + \text{abs}(s(t+k-1))$ 
13:  end for
14:  if  $err1 \leq (err2 * threshold)$ 
15:     $out(t) = 1$ 
16:    for  $k = 1:F$ 
17:       $s(t+k+1) = s(t+k+1) - fir(k)$ 
18:    end for
19:  end if
20: end for
21: output:  $out$ ,  $shift$ 

```

772 as possible. A three-step workflow methodology is proposed
773 here: selecting the encoding method appropriate to the original
774 signal, optimizing its parameters, and validating the encoded
775 signal. This paper aimed at providing analysis and guidelines
776 for these steps. The investigation was limited to temporal
777 signals and algorithms: temporal contrast (TBR, SF, and MW)
778 and stimulus encoding (BSA) method. If the SNN architecture
779 allows only unipolar spikes, BSA is the only appropriate
780 method of these. If bipolar spike trains are allowed, temporal

Algorithm 5 TBR, SF, MW Decoding

```

1: input:  $spikes$ ,  $threshold$ ,  $startpoint$ 
2:  $recon = \text{zeros}(\text{length}(spikes))$ 
3:  $recon(1) = startpoint$ 
4: for  $t = 2:\text{length}(spikes)$ 
5:   if  $spikes(t) == 1$ 
6:      $recon(t) = recon(t-1) + threshold$ 
7:   elseif  $spikes(t) == -1$ 
8:      $recon(t) = recon(t-1) - threshold$ 
9:   else
10:     $recon(t) = recon(t-1)$ 
11:  end if
12: end for
13: output:  $recon$ 

```

Algorithm 6 BSA Decoding

```

1: input:  $spikes$ ,  $fir$ ,  $shift$ 
2:  $out = \text{conv}(spikes, fir) + shift$ 
3: output:  $out$ 

```

contrast methods can also be used. In most cases for bipolar systems, SF should be the encoding method of choice due to its versatility and robustness. TBR has numerous disadvantages and, therefore, is only recommended for hardware simulation, development, or implementation since it suits online, fast applications. MW encoding suits very noisy signals but significantly cuts into the frequency domain; if this is not permissible, a digital filtering preprocessing step and SF encoding are recommended. Parameter optimization is based on calculated error metrics between the original and reconstructed real-value signals. Here, SNR is the recommended metric since negative values indicate prohibited areas and SNR values are invariant of signal amplitude which allows for comparisons. An additional validation step is done through the visual exploration of the original signal, encoded spike train, and reconstructed (decoded) signal in the time and/or frequency domain.

Future work is planned in the following directions:

- 1) development of error metrics in the frequency domain;
- 2) multiple variable encoding optimization in parallel, e.g., optimizing the encoding for each of the EEG channels;
- 3) automated validation of the encoding process;
- 4) adaptive encoding method selection and parameter optimization for streaming data with concept drifts.

Custom MATLAB software with graphical user interface for the selection and optimization of encoding methods is available on www.kedri.aut.ac.nz/neucube (named “spike encoding tools”).

APPENDIX

Encoding algorithms 1–4 and corresponding decoding algorithms 5 and 6 for all methods are as follows.

ACKNOWLEDGMENT

This work was carried out while B. Petro was on an extended academic visit at the Knowledge Engineering and Discovery Research Institute (KEDRI), Auckland University of Technology (AUT), Auckland, New Zealand. B. Petro would like to thank Dr. J. I. Espinosa-Ramos for helpful discussion and J. D’Mello for technical support.

REFERENCES

[1] N. Sengupta and N. Kasabov, “Spike-time encoding as a data compression technique for pattern recognition of temporal data,” *Inf. Sci.*, vols. 406–407, pp. 133–145, Sep. 2017.

[2] S. Furber, “To build a brain,” *IEEE Spectr.*, vol. 49, no. 8, pp. 44–49, Aug. 2012.

[3] P. Lichtsteiner and T. Delbruck, “A 64×64 aer logarithmic temporal derivative silicon retina,” in *Proc. Res. Microelectron. Electron.*, vol. 2, 2005, pp. 202–205.

[4] T. Delbruck. (2007). *JAER (Java Tools for Address-Event Representation Neuromorphic Processing) Open Source Project*. [Online]. Available: <http://jaerproject.net> and <https://github.com/SensorsINI/>

[5] S. Thorpe, A. Delorme, and R. Van Rullen, “Spike-based strategies for rapid processing,” *Neural Netw.*, vol. 14, nos. 6–7, pp. 715–725, 2001.

[6] A. Delorme and S. J. Thorpe, “SpikeNET: An event-driven simulation package for modelling large networks of spiking neurons,” *Netw., Comput. Neural Syst.*, vol. 14, no. 4, pp. 613–627, 2003.

[7] R. C. de Charms and M. M. Merzenich, “Primary cortical representation of sounds by the coordination of action-potential timing,” *Nature*, vol. 381, no. 6583, pp. 610–613, Jun. 1996.

[8] H. A. Mallot, *Coding and Representation*. Heidelberg, Germany: Springer, 2013, pp. 113–129.

[9] S. M. Bohte, “The evidence for neural information processing with precise spike-times: A survey,” *Natural Comput.*, vol. 3, no. 2, pp. 195–206, 2004.

[10] W. Maass, “Networks of spiking neurons: The third generation of neural network models,” *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, Dec. 1997.

[11] E. Chicca *et al.*, “A multichip pulse-based neuromorphic infrastructure and its application to a model of orientation selectivity,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 5, pp. 981–993, May 2007.

[12] B. Schrauwen and I. Van Campenhout, “BSA, a fast and accurate spike train encoding scheme,” in *Proc. Int. Joint Conf. Neural Netw.*, vol. 4, 2003, pp. 2825–2830.

[13] N. Kasabov *et al.*, “Evolving spatio-temporal data machines based on the NeuCube neuromorphic framework: Design methodology and selected applications,” *Neural Netw.*, vol. 78, pp. 1–14, Jun. 2016.

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

[15] M. Hough, H. De Garis, M. Korkin, F. Gers, and N. E. Nawa, “SPIKER: Analog waveform to digital spiketrain conversion in ATR’s artificial brain (cam-brain) project,” in *Proc. Int. Conf. Robot. Artif. Life*, 1999, pp. 1–4.

[16] N. Nuntalid, K. Dhoble, and N. Kasabov, “EEG classification with BSA spike encoding algorithm and evolving probabilistic spiking neural network,” in *Neural Information Processing (Lecture Notes in Computer Science)*, vol. 7062, 2011, pp. 451–460.

[17] N. K. Kasabov, “NeuCube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data,” *Neural Netw.*, vol. 52, pp. 62–76, Apr. 2014.

[18] N. Kasabov and E. Capecci, “Spiking neural network methodology for modelling, classification and understanding of EEG spatio-temporal data measuring cognitive processes,” *Inf. Sci.*, vol. 294, pp. 565–575, Feb. 2015.

[19] H. de Garis, N. E. Nawa, M. Hough, and M. Korkin, “Evolving an optimal de/convolution function for the neural net modules of ATR’s artificial brain project,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 1, 1999, pp. 438–443.

[20] J. Hu, Z.-G. Hou, Y.-X. Chen, N. Kasabov, and N. Scott, “EEG-based classification of upper-limb ADL using SNN for active robotic rehabilitation,” in *Proc. 5th IEEE RAS/EMBS Int. Conf. Biomed. Robot. Biomechatronics*, Aug. 2014, pp. 409–414.

[21] N. Sengupta, N. Scott, and N. Kasabov, “Framework for knowledge driven optimisation based data encoding for brain data modelling using spiking neural network architecture,” in *Proc. 5th Int. Conf. Fuzzy Neuro Comput.* in *Advances in Intelligent Systems and Computing*, vol. 415, 2015, pp. 109–118.

[22] N. Kasabov, K. Dhoble, N. Nuntalid, and G. Indiveri, “Dynamic evolving spiking neural networks for on-line spatio- and spectro-temporal pattern recognition,” *Neural Netw.*, vol. 41, pp. 188–201, May 2013.

[23] P. L. Nunez and R. Srinivasan, *Electric Fields of the Brain: The Neurophysics of EEG*. London, U.K.: Oxford Univ. Press, 2006.

[24] J. P. Varghese, R. E. McIlroy, and M. Barnett-Cowan, “Perturbation-evoked potentials: Significance and application in balance control research,” *Neurosci. Biobehavioral Rev.*, vol. 83, pp. 267–280, Dec. 2017.

[25] S. W. Smith, *The Scientist and Engineer’s Guide to Digital Signal Processing*. Bengaluru, Karnataka: California Technical, 1997.



Balint Petro was born in Hungary, in 1991. He received the B.Sc. degree in energy engineering and the M.Sc. degree in mechatronics engineering from the Budapest University of Technology and Economics (BUTE), Budapest, Hungary, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree with the Department of Mechatronics, Optics and Engineering Informatics with a focus on human upright balancing and developing methods to assess balancing abilities.

His current research interests include biomechanics and analysis of human motion and associated biomedical signals.

Mr. Petro received the Erasmus Mundus Action 2—PANTHER Project Grant to conduct an 8-month-long academic visit to the Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, Auckland, New Zealand, where he studied the application of spiking neural networks (SNNs) in biomedical signal processing.



Nikola Kasabov (F’10) received the M.Sc. degree in electrical engineering, with a specialization in computer science, and the Ph.D. degree in mathematical sciences from Technical University, Sofia, Bulgaria, in 1971 and 1975, respectively.

He is currently the Founder and the Director with the Knowledge Engineering and Discovery Research Institute (KEDRI), Auckland University of Technology, Auckland, New Zealand, where he is also the Personal Chair of knowledge engineering. He has authored over 620 works in the areas of intelligent systems, neural networks, connectionist and hybrid connectionist systems, fuzzy systems, expert systems, bioinformatics, and neuroinformatics.

Dr. Kasabov is a fellow of the Royal Society of New Zealand and the Distinguished Visiting Fellow of the Royal Academy of Engineering, U.K. He was the President of the International Neural Network Society (INNS). He is the President of the Asia-Pacific Neural Network Society (APNNS).



Rita M. Kiss received the M.Sc. and Ph.D. degrees in civil engineering and the Habilitated (Dr.habil.) degree in biomechanics from the Budapest University of Technology and Economics (BUTE), Budapest, Hungary, in 1991, 1997, and 2008, respectively, and the D.Sc. degree from Hungarian Academy of Sciences (HAS), in 2013.

She has been the Principal Investigator of multiple National Scientific Fund research groups in the area of biomedical engineering. She is currently a Full Professor with the Department of Mechatronics, Informatics, BUTE, where he is also the Director with the Biomechanical Cooperation Research Centre. She has authored or co-authored more than 300 publications in the areas of biomedical engineering, biomechanics, human motion analysis, biomaterial properties, and composite materials.

Dr. Kiss was a member of the International Society of Electromyography and Kinesiology in 2000 and the International Society of Biomechanics in Sport in 1999. He is a member of the International Association of Science and Technology for Development (IASTED) in 2004. From 2010 to 2017, she was on the Technical Committee on Biomedical Engineering of IASTED.

885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900

901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917

918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934

935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955

813
AQ:5 814
815
816
817
818
819

820

821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
AQ:6 840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
AQ:7 870
871
872
873
874
875
876
877
878
879
880
881
882
883
884