Neurocomputing 179 (2016) 264–282

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A tutorial on signal energy and its applications

Rodrigo Capobianco Guido

Instituto de Biociências, Letras e Ciências Exatas, Unesp - Univ Estadual Paulista (São Paulo State University) Rua Cristóvão Colombo 2265, Jd Nazareth, 15054-000, São José do Rio Preto - SP, Brazil

ARTICLE INFO

Article history:
Received 19 October 2015
Received in revised form
2 December 2015
Accepted 7 December 2015
Communicated by Hu Jun
Available online 17 December 2015

Keywords:
Signal energy
Pattern recognition
Feature extraction
Neurophysiological signal processing
Speech processing
Image processing

ABSTRACT

This tutorial, dedicated both to young professionals and students working with digital signal processing and pattern recognition, introduces three feature extraction approaches based on signal energy, characterising alternative and innovative ways for its use. The proposed theory, smoothly presented, is complemented with numerical examples, source-codes in C/C++ programming language, and applications in a diversity of computational problems, namely, neurophysiological signal processing, speech processing, and image processing. The lack of novelty in current energy-based approaches and the feasibility of a balance among *creativity*, *simplicity*, and *accuracy* constitutes the motivation for this text, which reveals how relevant the concept of signal energy may be, if properly employed.

1. Introduction

1.1. Motivation and tutorial structure

I have taught digital signal processing (DSP) and pattern recognition (PR) for almost twenty years, the period for which I have collected impressions from my undergraduate and graduate students of Computer Science, Electrical Engineering, and Applied Physics and Mathematics. This scenario I have been absorbed into, and mused on, inside which I have seen traditional concepts permanently being used without novelty, has motivated me to write this tutorial. Particularly, my intention is to show alternative and innovative ways to work with the concept of signal energy [1], improving its current usage as a feature extraction tool in benefit of DSP systems designed for PR [2,3]. I am absolutely sure that the aforementioned audience, for whom I dedicate this material, will take considerable advantage, not only of the concepts I present, but also of the tips and tricks shown, which provide a balance among *creativity*, *simplicity* and *accuracy*.

I have spent a considerable time to polish this text in order to offer the readers a smoothly written tutorial. Neither obscure equations nor algorithms with high orders of time and space complexity [4] can be found in the material I present. Instead, the mission of my essay, replete of explanations, figures, and source-codes in C/C++ programming language [5], is clarification. It innovates feature extraction based on signal energy, suggesting possible future trends. For that, it is organised as follows. The subject in study is presented in the next subsection of these introductory notes. Then, Section 2 describes the proposed approaches, Section 3 shows numerical examples, and Section 4 reports the tests and results obtained during the analyses of different types of signals. Lastly, Section 5 presents the conclusions.

Throughout the text, the reader will be able to understand that feature extraction, specially based on signal energy, plays an important role as being a methodology which aims to address real world complexities and improve real-life classification schemes, i.e., the ones which lead to uncertainty and partial truth, achieving tractability, robustness and low solution costs [6–22].

1.2. Problem statement and review

DSP systems designed for PR are characterised as being pattern-matching [23–26] or knowledge-based [27–32] approaches. In the former case, one or more template models derived from sample data are initially established in order to represent each class of interest.

E-mail address: guido@ieee.org
URL: http://www.sjrp.unesp.br/~guido

http://dx.doi.org/10.1016/j.neucom.2015.12.012
0925-2312/

Then, the classification of an unlabelled unidimensional (1D) digital signal, $s[\cdot]$, or a bidimensional (2D) digital signal, $m[\cdot][\cdot]$, is based on comparisons between the models and the input signal. The best match represents the class to which $s[\cdot]$, or $m[\cdot][\cdot]$, is assigned. In the latter case, one or more classifiers are trained in advance to identify $s[\cdot]$, or $m[\cdot][\cdot]$, as being a member of one or more specific classes, by means of statistical, numerical or optimisation-oriented approaches. Most of the times, the training procedures are based on sample data. Additionally, current literature describes some recent PR algorithms that combine pattern-matching and knowledge-based approaches. Classical examples of pattern-matching classifiers are Hard Thresholds (HTs), Distance Measures (DMs) and Correlation Coefficients (Rs). In contrast, Bayesian Modelling (BM), Artificial Neural Networks (ANNs) and Support Vector Machines (SVMs) represent knowledge-based approaches.

Disregarding the above-mentioned characterisation, and independent of the task carried out herein, i.e., template modelling, system training, signal assignment or identification, one restriction exists: neither $s[\cdot]$ nor $m[\cdot][\cdot]$ should be *directly* compared to the models or evaluated by the classifier. The reasons for that are their *relevant* and, most of the times, *variable* lengths, increasing computational costs and preventing the use of DMs, ANNs, or SVMs, that are only capable to lead with signals of predetermined dimensions. Furthermore, important information for classification may be hidden in the raw, or noisy, data. Therefore, an intermediary stage that converts digital signals of relevant and variable lengths to important information of reduced and predefined lengths, which are called feature vectors [33], is always required. This idea is shown in Fig. 1, being obvious that the feature-extraction step consists of a lossy procedure because, as mentioned above, it considerably reduces the dimension of the input signal. The direct consequence is that, the more the features preserve the information of interest from the signal under analysis, the more accurate the results of classification are.

Current literature teaches us that there are, basically, two different approaches for choosing features: *feature-learning* and *handcraft selection*. The former, outside the scope of this text, gathers a set of sophisticated techniques that learn the features automatically, forgoing human intervention [34]. Deep learning [35] is one classical example. On the other hand, in the latter, the system engineer is required to select the ideal features among many possibilities [2,3,36–46], being the signal energy, defined as

$$E(s[\cdot]) = \sum_{i=0}^{M-1} (s_i)^2, \tag{1}$$

with M representing the width, i.e., the length, of $s[\cdot]$, the simplest and most common one. Eq. (1) is adopted for 1D signals, however, in case of 2D signals, the definition becomes

$$E(m[\cdot][\cdot]) = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} (m_{i,j})^2, \tag{2}$$

being N and M, respectively, the height and the width of $m[\cdot][\cdot]$.

Physically, the energy of a signal represents its capacity to perform work [47]. Particularly, if a speech signal is considered, its energy describes the work the speaker's lungs and vocal tract performed, as a function of time, to produce sound [48]. On the other hand, if an electrocardiogram (ECG) signal is under analysis, its energy consists of the effort the subject's heart required to pump blood, as a function of time [49]. Assuming that an image, for instance, its energy along the space expresses the corresponding penetrating intensity of its colours [50], and so on. Much research, well represented by the ones that are published in [6–8,51–53], describe the use of signal energy in different applications.

Time and frequency decompositions are commonly used to extract energy-based features from different types of signals, such as biological ones. In [54], a novel energy-based diagnostic distortion measure is proposed to assess the reconstructed signal quality of ECG compression algorithms. Similarly, the authors of the paper [55] analyse ECG beats, from an energy point-of-view, by accounting for the features derived from non-linear component in time and frequency domains using an energy operator. An energy-efficient design for ECG R-peak detection is the objective of the article [56]. Energy features of electroencephalogram (EEG) signals are used for classification in the paper [57], while digital identification of normal and alcoholic EEG signals, based on energy, is explored in [58]. In [59], the authors detect artifacts from high energy bursts in neonatal EEG. On the other hand, energy demands of diverse spiking cells from the neocortex, hippocampus, and thalamus are explored in [60].

In the scope of image processing, we also have many applications of signal energy. The paper [61] is dedicated to the extraction of energy-based features in handwritten Chinese documents. In the paper [62], handwritten character recognition is performed by using energy, as the main feature, in association with an extreme learning machine. The authors of the paper [63] perform handwritten documents text-line segmentation based on energy. Active contours driven by local likelihood image fitting energy are used for image segmentation in the paper [64]. Energy-efficient low bit rate image compression is the focus of the paper [65], while the paper [66] describes an algorithm for dense depth image synthesis via energy minimization. A new multi-focus image fusion algorithm based on energy is the objective of the paper [67].

Speech and music processing is another field in which signal energy plays an important role. The paper [68] explores online speech and music segmentation on its basis, while energy-based speech enhancement for hands-free communication is discussed in [69]. In the paper [70], a monaural speech and music source separation scheme, using discrete energy separation algorithm, is proposed. The effects of energy on robust speech recognition is studied in [71]. On the other hand, a study on the consistency analysis of energy parameter for Mandarin speech recognition is described in [72]. The paper [73] discusses voiced/non-voiced detection in noisy speech signals based on energy, while an energy-efficient spike encoding circuit for speech edge detection is the topic of the paper [74]. The enhancement of noisy speech, based on its energy, is studied in the paper [75]. Additionally, the paper [76] introduces energy-based blind separating algorithms

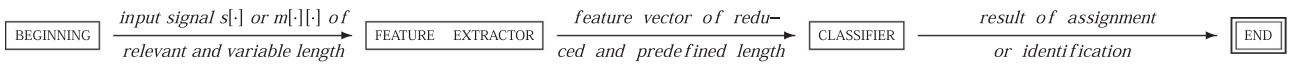


Fig. 1. Usual classification scheme for modern pattern-matching, knowledge-based, and combined approaches.

for speech signals. The papers [77] and [78] discuss, respectively, the influence of alcoholic intoxication on the short-time energy function of speech, and the speech energy redistribution for intelligibility improvement in noise.

Particularly, the problem addressed in this tutorial consists of showing three creative and intelligent ways to efficiently apply signal energy in benefit of neurophysiological, speech, and image processing, being easily extended to other types of signals. Signal energy has shown advantages over other features, such as zero-crossing rates (ZCRs) [79] and entropies [80]. First, it is physically meaningful both in time and frequency domains. Most of the tools used to convert a signal from the time to the frequency domain, or vice-versa, preserves its energy, being the Discrete Fourier Transform [81] and the Discrete Wavelet Transform [82] classical examples. This property allows feature extraction based on energy, and no other concept, be performed in both domains, once again motivating its use. Second, energy measures have advantages over strictly spectral features because they are more robust to different kinds of transmission and recording variations [83]-p.10-6. Third, other features such as ZCR and entropy were not designed to capture the information that signal energy does. ZCRs, for instance, are related to the spectral behaviour of the signal under analysis. On the other hand, entropy is a measure of signal randomness. Lastly, energy is one of the most humble concepts that can be used for feature extraction, simplifying procedures and stimulating the project of optimised DSP algorithms for PR, such as [7–9,11,12].

2. The proposed approaches

Three different methods, i.e., A_1, A_2, and A_3, are proposed in this section. Their corresponding details follow.

2.1. Method A_1

A_1, that is the simplest among the proposed approaches, focuses on grouping segments of the input signal under analysis in order to generate important information for classification. It consists of a sliding rectangular window of length L traversing $s[\cdot]$, or a sliding square of the same size traversing $m[\cdot][\cdot]$, so that, for each placement, the normalised signal energy over the window position, or square position, is computed. Each subsequent positioning overlaps in V% the previous and the excedent samples at the end of $s[\cdot]$, or at the end of $m[\cdot][\cdot]$, which are not long enough to be covered by the window, or by the square, are discarded. The restrictions $(1 \le L \le N)$, $(1 \le L \le M)$, and $(0 \le V < 100)$ apply. Instead of using a rectangular window, there are different possibilities, such as Hamming, Hanning, Kaiser, and others [1], however, they are not considered in this tutorial in order to keep the simplicity. Advanced readers can easily adapt the proposed approach to use them.

Thus, the feature vector, $f[\cdot]$, of length $T = \lfloor \frac{(100 \cdot M) - (L \cdot V)}{(100 - V) \cdot L} \rfloor$, being $\lfloor \cdot \rfloor$ the floor operator, is, for 1D signals,

$$f_k = \frac{\sum_{i=k\cdot\lfloor\frac{(100-V)}{100}L\rfloor}^{k\cdot\lfloor\frac{(100-V)}{100}L\rfloor+L-1} (s_i)^2}{\sum_{y=0}^{T-1}\sum_{r=y\cdot\lfloor\frac{(100-V)}{100}L\rfloor}^{y\cdot\lfloor\frac{(100-V)}{100}L\rfloor+L-1} (s_r)^2}, \quad (0 \le k \le T-1), \tag{3}$$

where the numerator corresponds to the signal energy computed over the kth positioning of the window, i.e., $E(w_k[\cdot])$, and the denominator corresponds to the total signal energy over $k = 0, 1, 2, \ldots, T-1$, i.e., $\sum_{k=0}^{T-1} E(w_k[\cdot])$, which is required for normalisation. Fig. 2 illustrates the sliding window for 1D and Algorithm 1 presents the source code, written in C++ programming language, that implements the corresponding procedure.

By extending Eq. 3 to a 2D signal, the feature vector $f[\cdot]$, now with length $T \cdot P$, being $P = \left\lfloor \frac{(100 \cdot N) - (L \cdot V)}{(100 - V) \cdot L} \right\rfloor$, becomes

$$f_k = \frac{\sum_{j=k\cdot\left\lfloor\frac{(100-V)}{100}L\right\rfloor}^{k\cdot\left\lfloor\frac{(100-V)}{100}L\right\rfloor+L-1} \sum_{i=k\cdot\left\lfloor\frac{(100-V)}{100}L\right\rfloor}^{k\cdot\left\lfloor\frac{(100-V)}{100}L\right\rfloor+L-1} (m_{i,j})^2}{\sum_{x=0}^{P-1}\sum_{y=0}^{T-1}\sum_{s=x\cdot\left\lfloor\frac{(100-V)}{100}L\right\rfloor}^{x\cdot\left\lfloor\frac{(100-V)}{100}L\right\rfloor+L-1} \sum_{r=y\cdot\left\lfloor\frac{(100-V)}{100}L\right\rfloor}^{y\cdot\left\lfloor\frac{(100-V)}{100}L\right\rfloor+L-1} (m_{r,s})^2}, \quad (0 \le k \le (T \cdot P)-1). \tag{4}$$

As in the previous case, the numerator of Eq. (4) corresponds to the energy computed over the kth square position, i.e., $E(w_k[\cdot])$, and the denominator corresponds to the total energy over $k = 0, 1, 2, \ldots, T-1$, i.e., $\sum_{k=0}^{T-1} E(w_k[\cdot])$, which is also required for normalisation. Fig. 3 illustrates the sliding square for 2D and Algorithm 2 shows its corresponding implementation.

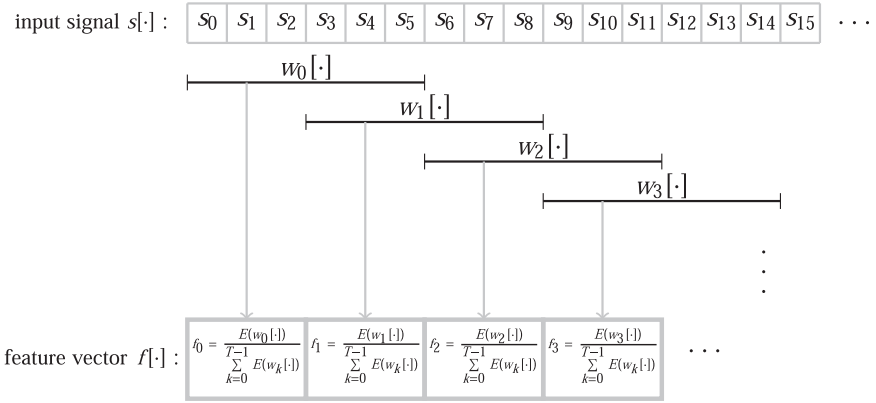


Fig. 2. 1D example for A_1: sliding window with length $L=6$ traversing $s[\cdot]$ with overlap $V=50$%. The symbols w_i represent the kth positioning of the window, for $k = 0, 1, 2, \ldots, T-1$.

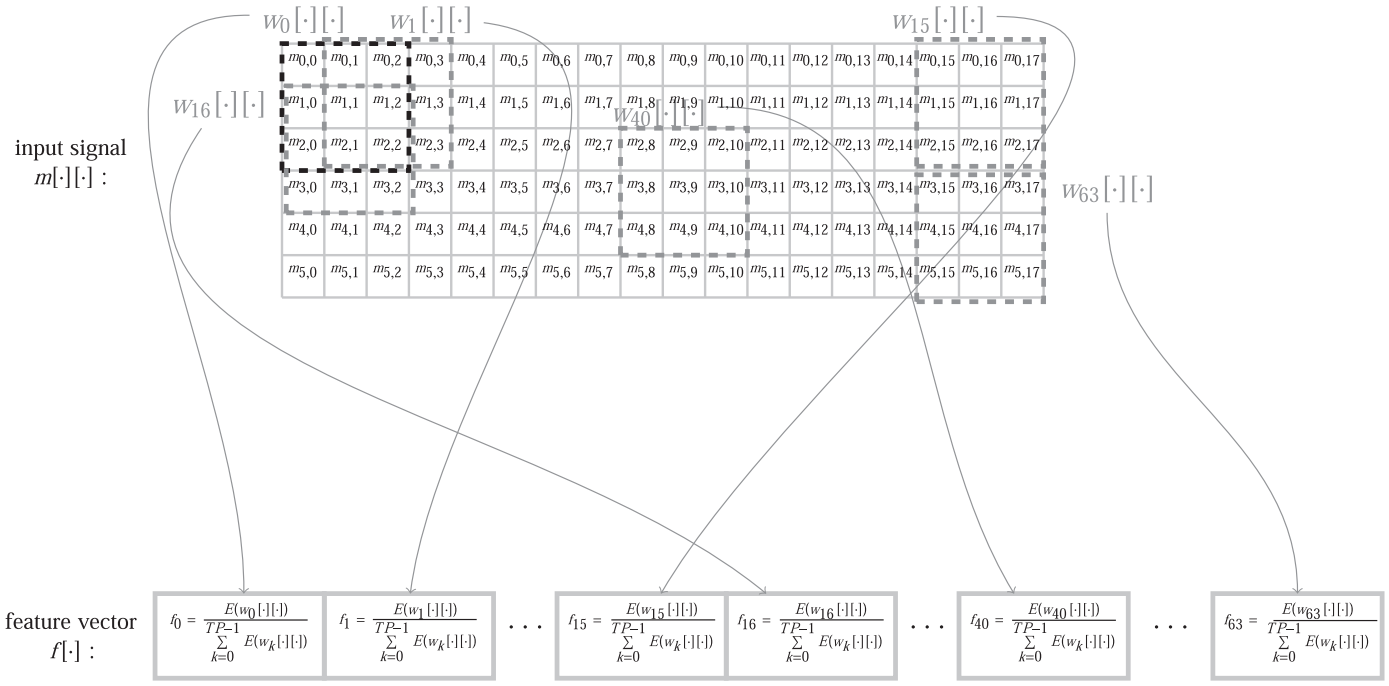


Fig. 3. 2D example for A_1: sliding square with length $L=3$ traversing $m[\cdot][\cdot]$ with overlap $V=66.67\%$. Again, $w_k[\cdot]$ indicate the kth position of the window, for $k=0,1,2,\ldots,(T\cdot P)-1$. Dashed squares in the arbitrary positions 0, 1, 15, 16, 40, and 63, are shown.

Algorithm 1. Fragment of C++ code for method A_1 in 1D.

```
//      ...
//      ensure that s[·], of length M, is available as input
int L= /* the desired value */;
int V= /* the desired value */;
int T= (int)((100*M-L*V)/((100-V) *L));
double E=0;  // E represents the total energy over all the positions of the window, that is required to normalise f[·]
double *f=new double[T];  // dynamic vector declaration
for(int k=0;k<T;k++)
 {
 f[k]=0;
 for(int i=k*((int)(((100-V)/100)*L));i<k*((int)(((100-V)/100)*L))+L;i++)
   f[k]+= pow(s[i],2);
 E+=f[k];
 }
for(int k=0;k<T;k++)
 f[k]/=E; // normalisation
 //      at this point, the feature vector, f[·], is ready
 //      ...
```

One important point is that, for 1D, A_1 is only capable of generating a T sample-long vector $f[\cdot]$ if the value of L is subjected to the value of M. The same holds true for 2D, i.e., a $T\cdot P$ sample-long vector $f[\cdot]$ is only obtained if the value of L is subjected to the values of N and M. Consequently, the value of L absolutely depends on the length of the input signal $s[\cdot]$, or $m[\cdot][\cdot]$, which creates a disadvantage: irregular temporal analysis. On the other hand, a great advantage exists: a few sequential energies over the corresponding window positions, obtained by predefining L and T, allow the detection of some particular event in the signal under analysis, such as an interval between two spoken words in a speech signal or a drastic color transition in an image. In these cases, N or the pair N and M, are disregarded, the input signal is framed, and then each frame, which generates T or $T\cdot P$ features, is analysed separately.

For both 1D and 2D, the main motivation for using A_1 is that, the closer V is to 100%, the more $f[\cdot]$, based on A_1, describes the smooth changes in the signal under analysis. On the other hand, the closer it is to 0%, the more $f[\cdot]$ exhibits the sudden variations existing in $s[\cdot]$, or $m[\cdot][\cdot]$.

Algorithm 2. Fragment of C++ code for method A_1 in 2D.

```
//      ...
//      ensure that m[·][·], with height N and width M, is available as input
int L= /* the desired value */;
int V= /* the desired value */;
int T= (int)((100*M-L*V)/((100-V) *L));
int P= (int)((100*N-L*V)/((100-V) *L));
double E=0 ;  //  E represents the total energy over all the positions of the window, required to normalise f[·]
double *f=new double[T*P];  //  dynamic vector declaration
for(int k=0;k<T*P;k++)
 {
 f[k]=0;
 for(int i=k*((int)(((100-V)/100)*L));i<k*((int)(((100-V)/100)*L))+L;i++)
    for(int j=k*(( int )(((100-V)/100))*L);j<k*(( int )(((100-V)/100))*L)+L;j++)
       f[k]+= pow(m[i][j],2);
 E+=f[k];
 }
for(int k=0;k<T*P;k++)
 f[k]/=E;
 //      at this point, the feature vector, f[·], is ready
 //      ...
```

2.2. Method A_2

Similarly to A_1, A_2 is also based on windowing $s[\cdot]$, or $m[\cdot][\cdot]$. However, there are two differences: no overlaps exist and the window length for 1D, or the rectangle sizes for 2D, vary. Their implications are directly related to the inspection of $s[\cdot]$, or $m[\cdot][\cdot]$, in different levels of resolution, being this the fundamental motivation to use this method.

For A_2, the feature vector, $f[\cdot]$, is defined as being the concatenation of Q sub-vectors of different dimensions, i.e., $f[\cdot]=\{\xi_1[\cdot]\}\cup\{\xi_2[\cdot]\}\cup\{\xi_3[\cdot]\}\cup\ldots\cup\{\xi_Q[\cdot]\}$.

Particularly, for 1D, each sub-vector is created by framing $s[\cdot]$ in T non-overlapping sequential windows, being T a prime number, and then calculating the corresponding normalised energies, i.e.,

- subvector $\xi_1[\cdot]$ is obtained by letting $L=\lfloor\frac{M}{2}\rfloor$, which that $T=2$, and by windowing $s[\cdot]$ based on Eq. 3 with $V=0$;
- idem to subvector $\xi_2[\cdot]$, obtained by letting $L=\lfloor\frac{M}{3}\rfloor$, which that $T=3$;
- idem to subvector $\xi_3[\cdot]$, obtained by letting $L=\lfloor\frac{M}{5}\rfloor$, which that $T=5$;
- ...
- idem to subvector $\xi_Q[\cdot]$, obtained by letting $L=\lfloor\frac{M}{X}\rfloor$, which that $T=X$.

Q may vary according to the objective of the analysis. The values $2,3,5,7,9,11,13,17,\ldots,X$ are possibilities for T, which is restricted to prime numbers in order to avoid one sub-vector to be a linear combination of another, bringing no advantage for classification.

For the 2D case, each sub-vector is created by framing $m[\cdot][\cdot]$ with $T\cdot P$ non-overlapping rectangles, being $T=P$ prime numbers, and then calculating the corresponding normalised energies, i.e.,

- subvector $\xi_1[\cdot]$ is determined by letting $L=\lfloor\frac{M}{2}\rfloor$ to obtain $T=2$, and then by letting $L=\lfloor\frac{N}{2}\rfloor$ to obtain $P=2$. Based on Eq. 4 with $V=0$, $m[\cdot][\cdot]$ is divided in $T\cdot P=2\cdot 2=4$ non-overlapping rectangles;
- idem to subvector $\xi_2[\cdot]$, obtained by letting $L=\lfloor\frac{M}{3}\rfloor$ and then $L=\lfloor\frac{N}{3}\rfloor$, which that $T=3$ and $P=3$, respectively, originating $T\cdot P=3\cdot 3=9$ non-overlapping rectangles;
- idem to subvector $\xi_3[\cdot]$, obtained by letting $L=\lfloor\frac{M}{5}\rfloor$ and then $L=\lfloor\frac{N}{5}\rfloor$, which that $T=5$ and $P=5$, respectively, originating $T\cdot P=5\cdot 5=25$ non-overlapping rectangles;
- ...
- idem to subvector $\xi_Q[\cdot]$, obtained by letting $L=\lfloor\frac{M}{X}\rfloor$ and then $L=\lfloor\frac{N}{X}\rfloor$, which that $T=X$ and $P=X$, respectively, originating $T\cdot P=X\cdot X=X^2$ non-overlapping rectangles.

Figs. 4 and 5 illustrate the sliding window for 1D and the sliding rectangle for 2D, respectively, at work. Additionally, Algorithms 3 and 4 contain the respective implementations.

2.3. Method A_3

As described above, A_1 and A_2 focus on measuring the levels of normalised energy over windows or rectangles of certain sizes. A_3, on the other hand, consists of determining the proportional lengths, or areas, of the signal under analysis that are required to reach pre-defined levels of energy. The direct consequence of this approach is the characterisation of A_3 as being ideal to inspect the constancy in action of the physical entity responsible for generating $s[\cdot]$, or $m[\cdot][\cdot]$.

Specifically, C is defined as being the critical base-level of energy, $(0<C<100)$, and, then, for 1D, the feature vector $f[\cdot]$, of size T, is determined as follows:

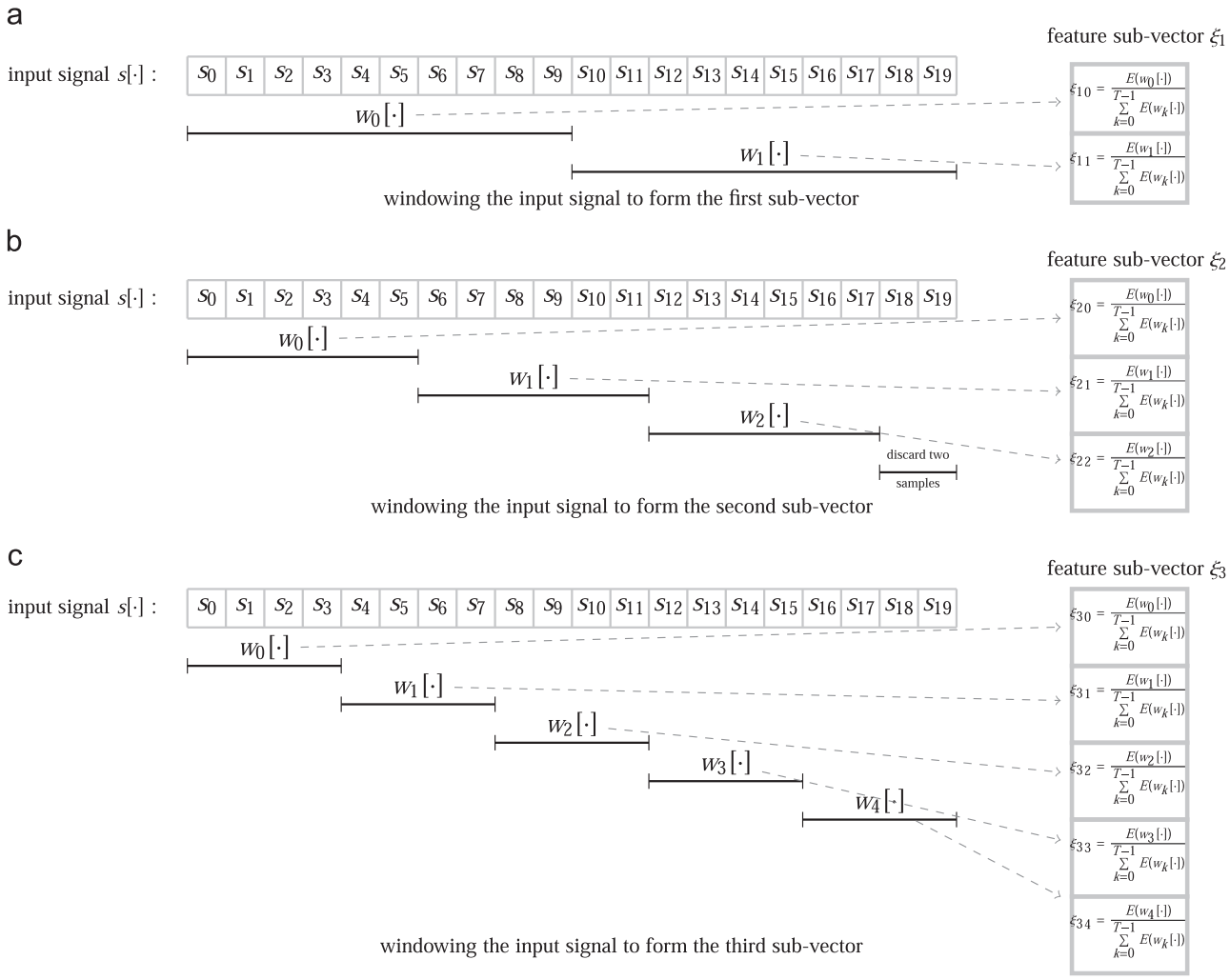


Fig. 4. 1D example for A_2 assuming Q=3: (a) sliding window, with length $L = \lfloor \frac{M}{2} \rfloor = \lfloor \frac{20}{2} \rfloor = 10$ traversing $s[\cdot]$ in order to compose $\xi_1[\cdot]$; (b) sliding window with length $L = \lfloor \frac{M}{3} \rfloor = \lfloor \frac{20}{3} \rfloor = 6$ traversing $s[\cdot]$ in order to compose $\xi_2[\cdot]$; (c) sliding window with length $L = \lfloor \frac{M}{5} \rfloor = \lfloor \frac{20}{5} \rfloor = 4$ traversing $s[\cdot]$ in order to compose $\xi_3[\cdot]$. The window positions do not overlap and the symbols w_i indicate the ith window position, for $i = 0, 1, 2, \ldots, T-1$. (a) Windowing the input signal to form the first sub-vector. (b) Windowing the input signal to form the second sub-vector. (c) Windowing the input signal to form the third sub-vector.

- f_0 is the proportion of $s[\cdot]$ length starting from its beginning, which is covered by the window $w_0[\cdot]$, required to reach C% of the total signal energy;
- f_1 is the proportion of $s[\cdot]$ length starting from its beginning, which is covered by the window $w_1[\cdot]$, required to reach $2 \cdot C$% of the total signal energy;
- f_2 is the proportion of $s[\cdot]$ length starting from its beginning, which is covered by the window $w_2[\cdot]$, required to reach $3 \cdot C$% of the total signal energy;
- …
- f_{T-1} is the proportion of $s[\cdot]$ length starting from its beginning, which is covered by the window $w_{T-1}[\cdot]$, required to reach $(T \cdot C)$% of the total signal energy, so that $(T \cdot C) < 100\%$;

For A_3, the value of T is defined as being:

$$T = \begin{cases} \frac{100}{C} - 1 & \text{if } C \text{ is multiple of 100;} \\ \lfloor \frac{100}{C} \rfloor & \text{otherwise.} \end{cases}$$

The 2D version of A_3 imply that $f[\cdot]$, with the same size T, is determined as follows:

- f_0 is the proportion of $m[\cdot][\cdot]$ area, starting from $m_{0,0}$ and covered by the $\lfloor \alpha_0 \rfloor$ x $\lfloor \beta_0 \rfloor$ rectangle $w_0[\cdot][\cdot]$, required to reach C% of the total signal energy;
- f_1 is the proportion of $m[\cdot][\cdot]$ area, starting from $m_{0,0}$ and covered by the $\lfloor \alpha_1 \rfloor$ x $\lfloor \beta_1 \rfloor$ rectangle $w_1[\cdot][\cdot]$, required to reach $2 \cdot C$% of the total signal energy;
- f_2 is the proportion of $m[\cdot][\cdot]$ area, starting from $m_{0,0}$ and covered by the $\lfloor \alpha_2 \rfloor$ x $\lfloor \beta_2 \rfloor$ rectangle $w_2[\cdot][\cdot]$, required to reach $3 \cdot C$% of the total signal energy;

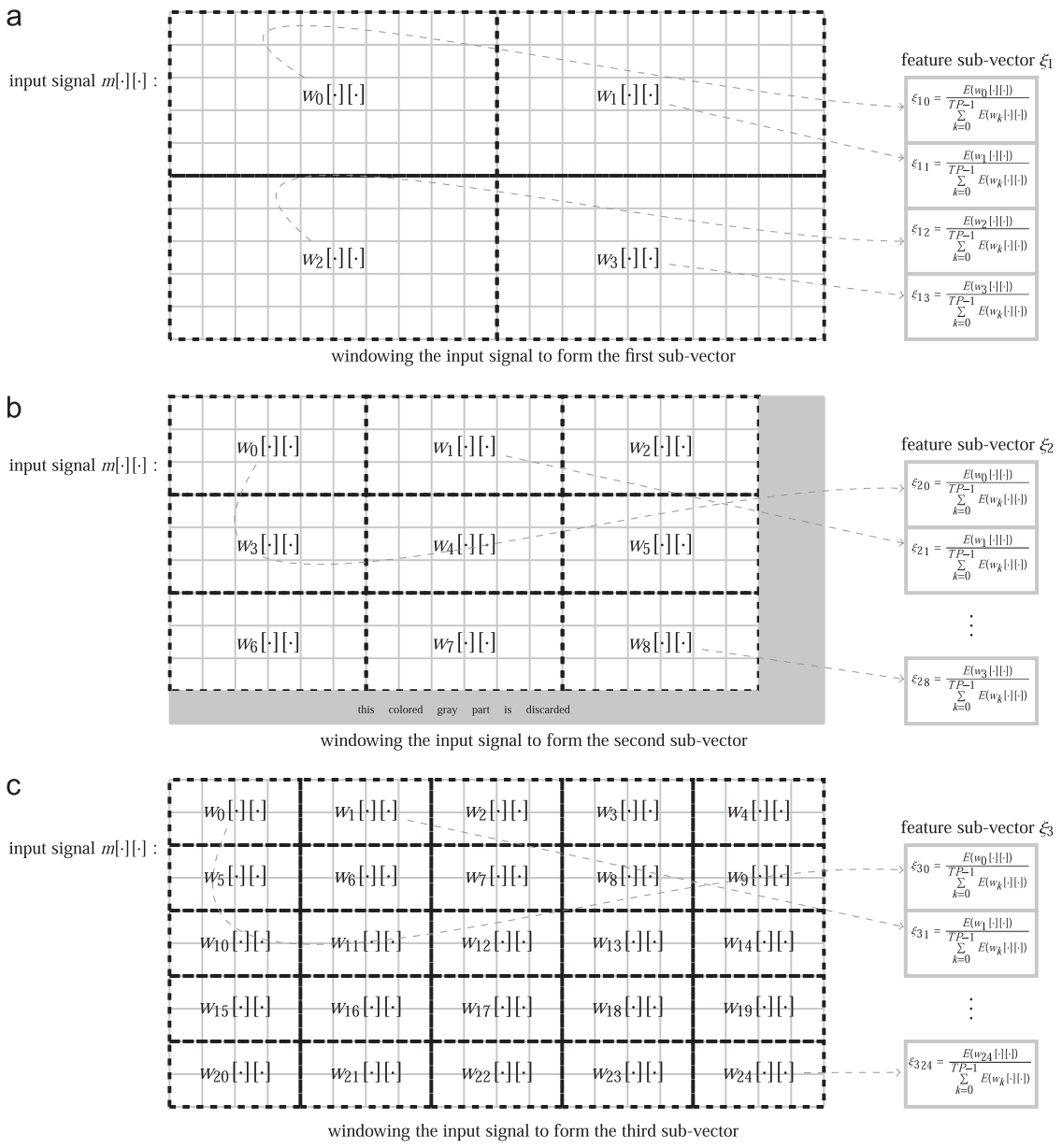


Fig. 5. 2D example for A_2 assuming that $Q=3$ subvectors: [above] sliding square with length $\{\lfloor\frac{M}{2}\rfloor \times \lfloor\frac{N}{2}\rfloor\} = \{\lfloor\frac{10}{2}\rfloor \times \lfloor\frac{20}{2}\rfloor\} = 5 \times 10$ traversing $m[\cdot][\cdot]$ in order to compose $\xi_1[\cdot]$; [middle] sliding square with length $\{\lfloor\frac{M}{3}\rfloor \times \lfloor\frac{N}{3}\rfloor\} = \{\lfloor\frac{10}{3}\rfloor \times \lfloor\frac{20}{3}\rfloor\} = 3 \times 6$ traversing $m[\cdot][\cdot]$ in order to compose $\xi_2[\cdot]$; [below] sliding square with length $\{\lfloor\frac{M}{5}\rfloor \times \lfloor\frac{N}{5}\rfloor\} = \{\lfloor\frac{10}{5}\rfloor \times \lfloor\frac{20}{5}\rfloor\} = 2 \times 4$ traversing $m[\cdot][\cdot]$ in order to compose $\xi_3[\cdot]$. Again, w_i indicates the ith window position, for $i = 0, 1, 2, \ldots, (T \cdot P) - 1$, with no overlap. Dashed squares represent the sliding window in all possible positions. (a) Windowing the input signal to form the first sub-vector. (b) Windowing the input signal to form the second sub-vector. (c) Windowing the input signal to form the third sub-vector.

- …
- f_{TP-1} is the proportion of $m[\cdot][\cdot]$ area, starting from $m_{0,0}$ and covered by the $\lfloor\alpha_{T-1}\rfloor$ x $\lfloor\beta_{T-1}\rfloor$ rectangle $w_{T-1}[\cdot][\cdot]$, required to reach $T \cdot C\%$ of the total signal energy, so that $(T \cdot C) < 100\%$;

The values of α_i and β_i, $(0 \le i \le T-1)$, are determined according to the following rule:

beginning : $(\alpha_i \leftarrow 0)$ and $(\beta_i \leftarrow 0)$, unconditionally.

$$\text{repeat} \begin{cases} (\alpha_i \leftarrow \alpha_i + 1) & \text{and} \quad (\beta_i \leftarrow \beta_i + 1) & \text{if} \quad (N = M) \\ (\alpha_i \leftarrow \alpha_i + 1) & \text{and} \quad (\beta_i \leftarrow \beta_i + \frac{\beta_i}{\alpha_i}) & \text{if} \quad (N > M) \\ (\alpha_i \leftarrow \alpha_i + \frac{\alpha_i}{\beta_i}) & \text{and} \quad (\beta_i \leftarrow \beta_i + 1) & \text{otherwise.} \end{cases}$$

until the desired level of energy, i.e., $C, 2 \cdot C, 3 \cdot C, \ldots, T \cdot C$ is reached.

Figs. 6 and 7 show A_3 at work for 1D and 2D, respectively, and Algorithms 5 and 6 present the respective implementations. In the next section, numerical examples are shown for each one of the proposed methods.

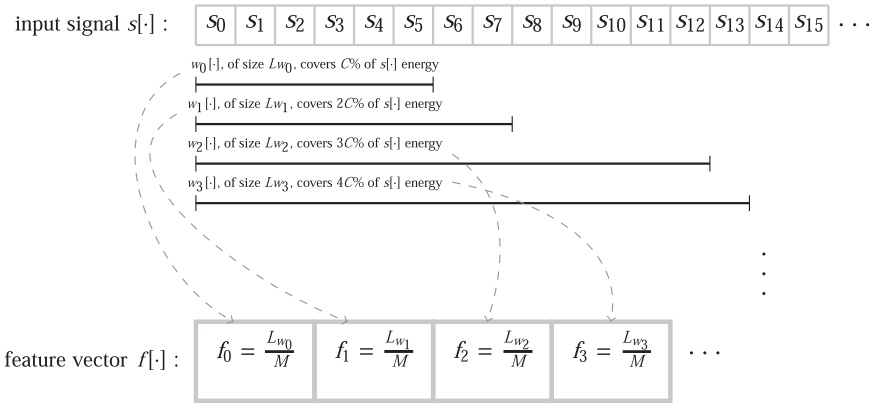


Fig. 6. 1D example for A_3, where L_{w_i} represents the length of the window $w_i[\cdot]$, for $i = 0, 1, 2, 3, \ldots, T-1$.

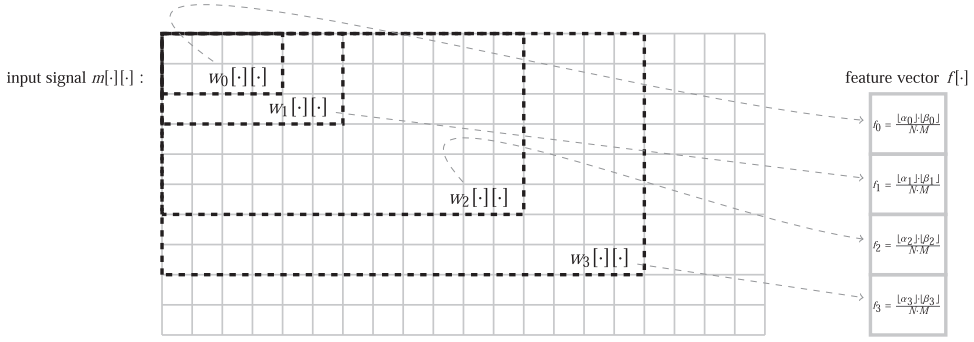


Fig. 7. 2D example for A_3, where $w_i[\cdot]$ corresponds to the ith window, and α_i and β_i represent, respectively, its height and width, for $i = 0, 1, 2, 3, \ldots, TP-1$.

Algorithm 3. Fragment of C++ code for method A_2 in 1D.

```
//      ...
//      ensure that s[·], with length M, is available as input
int L; // window length
double E;  //  E represents the total energy over all the window positions, that is required to normalise f[·]
int X[] = {2, 3, 5, 7, 9, 11, 13, 17}; /* vector containing the prime numbers of interest. It can be changed according to the experiment */
int total_size_of_f =0;
for(int i = 0; i < (int)(sizeof(X)/sizeof(int)); i++) //  number of elements in X[·]
  total_size_of_f+ =X[i];
double *f = new double[total_size_of_f]; /* The total size of f[·] is the sum of the elements in X[·], i.e., the size of the subvector ξ1[·] plus the
   size of the subvector ξ2[·], plus the size of the subvector ξ3[·], ..., and so on */
int jump=0; // helps to control the correct positions to write in f[·]
for(int j = 0; j < (int)(sizeof(X)/sizeof(int)); j++)
  {
  E=0;
  for(int k = 0; k < X[j]; k++)
     {
     L = (int)(M/X[j]);
     f[jump+k] = 0;
     for(int i = (k*L); i < (k*L)+L; i++)
        f[jump+k]+ =  pow(s[i], 2);
     E+ =f[jump+k];
     }
  for(int k = 0; k < X[j]; k++)
     f[jump+k]/ = E;
  jump+ =X[j];
  }
//      at this point, the feature vector, f[·], is ready.
//      ...
```

Algorithm 4. Fragment of C++ code for method A_2 in 2D.

```
//      …
//      ensure that m[·][·], with height N and width M, is available as input
int L1, L2;
double E;  // E represents the total energy over all the window positions, that is required to normalise f[·]
int X[] = {2, 3, 5, 7, 9, 11, 13, 17}; /* vector containing the prime numbers of interest. It can be changed according to the experiment */
int total_size_of_f =0;
for(int i = 0; i < (int)(sizeof(X)/sizeof(int));i++)  // number of elements in X[·]
  total_size_of_f+ =pow(X[i], 2);
double *f =new double[total_size_of_f]; /* The total size of f[·] is the sum of the squares of the elements in X[·], i.e., the size of the subvector
   ξ1[·] plus the size of the subvector ξ2[·], plus the size of the subvector ξ3[·], …, and so on */
int jump=0; // helps to control the correct positions to write in f[·]
for(int i=0;i<  total_size_of_f;i++)
  f[i] = 0;
for(int k=0;k<(int)(sizeof(X)/sizeof(int));k++)
  {
  E=0;
  L1 = (int)(M/X[k]);
  L2 = (int)(N/X[k]);
  for(int i=0;i<((int)(N/X[k]))*X[k];i++)
     for(int j=0;j<((int)(M/X[k]))*X[k]; j++)
     {
     f[jump+(((int)(i/L2))*(X[k]))+((int)(j/L1))]+ =  pow(m[i][j], 2);
     E+ =  pow(m[i][j], 2);
     }
  for(int i =  jump;i<  jump+pow(X[k], 2); i++)
     f[i]/ = E;
  jump+ =pow(X[k], 2);
  }
 //      at this point, the feature vector, f[·], is ready.
 //      …
```

Algorithm 5. Fragment of C++ code for method A_3 in 1D.

```
//      …
//      … ensure that s[·], of length M, is available as input
int L = 0;  // partial lengths
double C = ;  // the desired value, being 0 < C < 100
int T = ((100/C)−((int)(100/C)) = = 0)?(100/C)−1 : (int)(100/C) ;  // the number of elements in T
double *f = new double[T];  // dynamic vector declaration
double z=energy(&s[0], M)*((double)(C)/100);
for(int k = 0; k < T; k++)
  {
  while(energy(&s[0], L)  < ((k+1)*z))
     L++;
  f[k] = (double)(L)/(double)(M);
  }
//      at this point, the feature vector, f[·], is ready
//      …
//      function energy
double energy(double* input_vector, int length)
{
double e=0;
for(int i=0;i  <  length;i++)
  e  +=pow(input_vector[i],2);
return(e);
}
//      …
```

3. Numerical examples

In order to illustrate the proposed approaches, one numerical example follows for each case: methods A_1, A_2, and A_3, both in 1D and 2D, based on hypothetical data.

3.1. Numerical example for A_1 in 1D

Problem Statement: Let $s[\cdot]=\{2,4,6,8,10,9,7,5,3,1\}$, implying in $M=10$. Assume that the associated window length is $L=4$ with overlaps of $V=50\%$. Obtain the feature vector, $f[\cdot]$, following method A_1.

Solution: According to the previous explanations, the corresponding feature vector with length $T=\left\lfloor\frac{(100\cdot M)-(L\cdot V)}{(100-V)\cdot L}\right\rfloor=\left\lfloor\frac{(100\cdot 10)-(4\cdot 50)}{(100-50)\cdot 4}\right\rfloor=4$ is obtained as follows:

- The first window position is $w_0[\cdot]=\{2,4,6,8\}$. Its energy is $E(w_0[\cdot])=(2^2+4^2+6^2+8^2)=120$;
- The second window position is $w_1[\cdot]=\{6,8,10,9\}$. Its energy is $E(w_1[\cdot])=(6^2+8^2+10^2+9^2)=281$;
- The third window position is $w_2[\cdot]=\{10,9,7,5\}$. Its energy is $E(w_2[\cdot])=(10^2+9^2+7^2+5^2)=255$;
- The fourth window position is $w_3[\cdot]=\{7,5,3,1\}$. Its energy is $E(w_3[\cdot])=(7^2+5^2+3^2+1^2)=84$;
- $f_0=\frac{E(w_0[\cdot])}{E(w_0[\cdot])+E(w_1[\cdot])+E(w_2[\cdot])+E(w_3[\cdot])}=\frac{120}{120+281+255+84}=0.1622$;
- $f_1=\frac{E(w_1[\cdot])}{E(w_0[\cdot])+E(w_1[\cdot])+E(w_2[\cdot])+E(w_3[\cdot])}=\frac{281}{120+281+255+84}=0.3797$;
- $f_2=\frac{E(w_2[\cdot])}{E(w_0[\cdot])+E(w_1[\cdot])+E(w_2[\cdot])+E(w_3[\cdot])}=\frac{255}{120+281+255+84}=0.3446$;
- $f_3=\frac{E(w_3[\cdot])}{E(w_0[\cdot])+E(w_1[\cdot])+E(w_2[\cdot])+E(w_3[\cdot])}=\frac{84}{120+281+255+84}=0.1135$.

Thus, $f[\cdot]=\{0.1622,0.3797,0.3446,0.1135\}$.

Algorithm 6. Fragment of C++ code for method A_3 in 2D.

```
//      ...
//      ensure that m[·][·], with height N and width M, is available as input
double alpha=0; //  partial height
double beta=0; //  partial width
double C=; //  the desired value,being 0<C<100
int T=((100/C)-((int)(100/C))==0)?(100/C)-1 : (int)(100/C) ; //  the number of elements in T
double *f=new double[T];  //  dynamic vector declaration
double z=energy(m,N,M)*((double)(C)/100.0);
for(int k=0;k<T;k++)
 {
 while(energy(&m[0][0],(int)(alpha), (int)(beta))<((k+1)*z))
   if (alpha==beta)
     {
     alpha ++;
     beta ++;
     }
   else if (alpha > beta)
     {
     alpha ++;
     beta+=beta/alpha;
     }
   else
     {
     alpha+=alpha/beta;
     beta ++;
     }
 f[k]=((alpha*beta)<=(N*M))?((alpha*beta)/(N*M)):(1);  //  exceptionally, as alpha or beta increases, f[k]>1 for k close to T, so this is
  a correction.
 }
//      at this point, the feature vector, f[·], is ready
//      ...
//      ... function energy
double energy(double** input_matrix, int height, int width)
{
double e=0;
for(int i=0;i < height;i ++)
 for(int j=0;j < width;j ++)
    e +=pow(input_matrix[i][j],2);
return(e);
}
//      ...
```

3.2. Numerical Example for A_1 in 2D

Problem Statement: Let $m[\cdot][\cdot]=\begin{pmatrix}1 & 2 & 3 & 4\\ 5 & 6 & 7 & 8\\ 9 & 10 & 11 & 12\end{pmatrix}$, implying in $N=3$ and $M=4$. Assume that the square window has size $L=2$ with overlaps of $V=50\%$. Obtain the feature vector, $f[\cdot]$, following method A_1.

Solution: The feature vector with length $T\cdot P=\left\lfloor\frac{(100\cdot M)-(L\cdot V)}{(100-V)\cdot L}\right\rfloor\cdot\left\lfloor\frac{(100\cdot N)-(L\cdot V)}{(100-V)\cdot L}\right\rfloor\cdot=\left\lfloor\frac{(100\cdot 4)-(2\cdot 50)}{(100-50)\cdot 2}\right\rfloor\cdot\left\lfloor\frac{(100\cdot 3)-(2\cdot 50)}{(100-50)\cdot 2}\right\rfloor=3\cdot 2=6$ is obtained as follows:

- The first window position is $w_0[\cdot][\cdot]=\left(\begin{smallmatrix}1 & 2\\ 5 & 6\end{smallmatrix}\right)$. Its energy is $E(w_0[\cdot][\cdot])=(1^2+2^2+5^2+6^2)=66$;
- The second window position is $w_1[\cdot][\cdot]=\left(\begin{smallmatrix}2 & 3\\ 6 & 7\end{smallmatrix}\right)$. Its energy is $E(w_1[\cdot][\cdot])=(2^2+3^2+6^2+7^2)=98$;
- The third window position is $w_2[\cdot][\cdot]=\left(\begin{smallmatrix}3 & 7\\ 7 & 8\end{smallmatrix}\right)$. Its energy is $E(w_2[\cdot][\cdot])=(3^2+4^2+7^2+8^2)=138$;
- The fourth window position is $w_3[\cdot][\cdot]=\left(\begin{smallmatrix}5 & 6\\ 9 & 10\end{smallmatrix}\right)$. Its energy is $E(w_3[\cdot][\cdot])=(5^2+6^2+9^2+10^2)=242$;
- The fifth window position is $w_4[\cdot][\cdot]=\left(\begin{smallmatrix}6 & 7\\ 10 & 11\end{smallmatrix}\right)$. Its energy is $E(w_4[\cdot][\cdot])=(6^2+7^2+10^2+11^2)=306$;
- The sixth window position is $w_5[\cdot][\cdot]=\left(\begin{smallmatrix}7 & 8\\ 11 & 12\end{smallmatrix}\right)$. Its energy is $E(w_5[\cdot][\cdot])=(7^2+8^2+11^2+12^2)=378$;
- $f_0=\frac{E(w_0[\cdot][\cdot])}{\sum_{i=0}^{5}E(w_i[\cdot][\cdot])}=\frac{66}{66+98+138+242+306+378}=0.0537$;
- $f_1=\frac{E(w_1[\cdot][\cdot])}{\sum_{i=0}^{5}E(w_i[\cdot][\cdot])}=\frac{98}{66+98+138+242+306+378}=0.0798$;
- $f_2=\frac{E(w_2[\cdot][\cdot])}{\sum_{i=0}^{5}E(w_i[\cdot][\cdot])}=\frac{138}{66+98+138+242+306+378}=0.1124$;
- $f_3=\frac{E(w_3[\cdot][\cdot])}{\sum_{i=0}^{5}E(w_i[\cdot][\cdot])}=\frac{242}{66+98+138+242+306+378}=0.1971$;
- $f_4=\frac{E(w_4[\cdot][\cdot])}{\sum_{i=0}^{5}E(w_i[\cdot][\cdot])}=\frac{306}{66+98+138+242+306+378}=0.2492$;
- $f_5=\frac{E(w_5[\cdot][\cdot])}{\sum_{i=0}^{5}E(w_i[\cdot][\cdot])}=\frac{378}{66+98+138+242+306+378}=0.3078$.

Thus, $f[\cdot]=\{0.0537, 0.0798, 0.1124, 0.1971, 0.2492, 0.3078\}$.

3.3. Numerical example for A_2 in 1D

Problem Statement: Let $s[\cdot]=\{0,1,2,3,4,5,6,7,8,9\}$, implying in $M=10$. Assume that $Q=3$ with no overlaps between window positions. Obtain the feature vector, $f[\cdot]$, following method A_2.

Solution: The feature vector is composed by the concatenation of $Q=3$ sub-vectors, i.e., $f[\cdot]=\{\xi_1[\cdot]\}\cup\{\xi_2[\cdot]\}\cup\{\xi_3[\cdot]\}$. These are obtained as follows:

- To obtain the first subvector, $\xi_1[\cdot]$, two non-overlapping windows, $w_0[\cdot]=\{0,1,2,3,4\}$ and $w_1[\cdot]=\{5,6,7,8,9\}$, are positioned over $s[\cdot]$. Their normalised energies, i.e.,

$$\frac{E(w_0)}{\sum_{i=0}^{1}E(w_i)}=\frac{0^2+1^2+2^2+3^2+4^2}{(0^2+1^2+2^2+3^2+4^2)+(5^2+6^2+7^2+8^2+9^2)}=\frac{30}{285}=0.1053$$

and

$$\frac{E(w_1)}{\sum_{i=0}^{1}E(w_i)}=\frac{5^2+6^2+7^2+8^2+9^2}{(0^2+1^2+2^2+3^2+4^2)+(5^2+6^2+7^2+8^2+9^2)}=\frac{255}{285}=0.8947,$$

originate $\xi_1[\cdot]=\{0.1053, 0.8947\}$;

- To obtain the second subvector, $\xi_2[\cdot]$, three non-overlapping windows, $w_0[\cdot]=\{0,1,2\}$, $w_1[\cdot]=\{3,4,5\}$, and $w_2[\cdot]=\{6,7,8\}$, are positioned over $s[\cdot]$, discarding its last element, i.e., the amplitude 9. Their normalised energies, i.e.,

$$\frac{E(w_0)}{\sum_{i=0}^{2}E(w_i)}=\frac{0^2+1^2+2^2}{(0^2+1^2+2^2)+(3^2+4^2+5^2)+(6^2+7^2+8^2)}=\frac{5}{204}=0.0245,$$

$$\frac{E(w_1)}{\sum_{i=0}^{2}E(w_i)}=\frac{3^2+4^2+5^2}{(0^2+1^2+2^2)+(3^2+4^2+5^2)+(6^2+7^2+8^2)}=\frac{50}{204}=0.2451,$$

and

$$\frac{E(w_2)}{\sum_{i=0}^{2}E(w_i)}=\frac{6^2+7^2+8^2}{(0^2+1^2+2^2)+(3^2+4^2+5^2)+(6^2+7^2+8^2)}=\frac{149}{204}=0.7304,$$

originate $\xi_2[\cdot]=\{0.0245, 0.2451, 0.7304\}$;

- To get the third subvector, $\xi_3[\cdot]$, five non-overlapping windows, $w_0[\cdot]=\{0,1\}$, $w_1[\cdot]=\{2,3\}$, $w_2[\cdot]=\{4,5\}$, $w_3[\cdot]=\{6,7\}$, and $w_4[\cdot]=\{8,9\}$, are positioned over $s[\cdot]$. Their normalised energies, i.e.,

$$\frac{E(w_0)}{\sum_{i=0}^{4}E(w_i)}=\frac{0^2+1^2}{(0^2+1^2)+(2^2+3^2)+(4^2+5^2)+(6^2+7^2)+(8^2+9^2)}=\frac{1}{285}=0.0035,$$

$$\frac{E(w_1)}{\sum_{i=0}^{4}E(w_i)}=\frac{2^2+3^2}{(0^2+1^2)+(2^2+3^2)+(4^2+5^2)+(6^2+7^2)+(8^2+9^2)}=\frac{13}{285}=0.0456,$$

$$\frac{E(w_2)}{\sum_{i=0}^{4}E(w_i)}=\frac{4^2+5^2}{(0^2+1^2)+(2^2+3^2)+(4^2+5^2)+(6^2+7^2)+(8^2+9^2)}=\frac{41}{285}=0.1439,$$

$$\frac{E(w_3)}{\sum_{i=0}^{4}E(w_i)}=\frac{6^2+7^2}{(0^2+1^2)+(2^2+3^2)+(4^2+5^2)+(6^2+7^2)+(8^2+9^2)}=\frac{85}{285}=0.2982,$$

and

$$\frac{E(w_4)}{\sum_{i=0}^{4}E(w_i)}=\frac{8^2+9^2}{(0^2+1^2)+(2^2+3^2)+(4^2+5^2)+(6^2+7^2)+(8^2+9^2)}=\frac{145}{285}=0.5088,$$

originate $\xi_3[\cdot]=\{0.0035,0.0456,0.1439,0.2982,0.5088\}$.

The concatenation of the sub-vectors produce $f[\cdot]=\{0.1053,0.8947,0.0245,0.2451,0.7304,0.0035,0.0456,0.1439,0.2982,0.5088\}$.

3.4. Numerical example for A_2 in 2D

Problem Statement: Let $m[\cdot][\cdot]=\begin{pmatrix}0&1&2&3\\4&5&6&7\\8&9&10&11\\12&13&14&15\end{pmatrix}$, implying in $N=4$ and $M=4$. Assume that $Q=2$ with no overlaps between windows. Obtain the feature vector, $f[\cdot]$, following method A_2.

Solution: The feature vector is composed by the concatenation of $Q=2$ sub-vectors, i.e., $f[\cdot]=\{\xi_1[\cdot]\}\cup\{\xi_2[\cdot]\}$. They are obtained as follows:

- To obtain the first subvector, $\xi_1[\cdot]$, a total of $2\cdot 2=4$ non-overlapping windows, $w_0[\cdot][\cdot]=\left(\begin{smallmatrix}0&1\\4&5\end{smallmatrix}\right)$, $w_1[\cdot][\cdot]=\left(\begin{smallmatrix}2&3\\6&7\end{smallmatrix}\right)$, $w_2[\cdot][\cdot]=\left(\begin{smallmatrix}8&9\\12&13\end{smallmatrix}\right)$, and $w_3[\cdot][\cdot]=\left(\begin{smallmatrix}10&11\\14&15\end{smallmatrix}\right)$, are positioned over $m[\cdot][\cdot]$. Their normalised energies, i.e.,

$$\frac{E(w_0)}{\sum_{i=0}^{3}E(w_i)}=\frac{0^2+1^2+4^2+5^2}{(0^2+1^2+4^2+5^2)+(2^2+3^2+6^2+7^2)+(8^2+9^2+12^2+13^2)+(10^2+11^2+14^2+15^2)}=\frac{42}{42+98+458+642}=0.0339,$$

$$\frac{E(w_1)}{\sum_{i=0}^{3}E(w_i)}=\frac{2^2+3^2+6^2+7^2}{(0^2+1^2+4^2+5^2)+(2^2+3^2+6^2+7^2)+(8^2+9^2+12^2+13^2)+(10^2+11^2+14^2+15^2)}=\frac{98}{42+98+458+642}=0.0790,$$

$$\frac{E(w_2)}{\sum_{i=0}^{3}E(w_i)}=\frac{8^2+9^2+12^2+13^2}{(0^2+1^2+4^2+5^2)+(2^2+3^2+6^2+7^2)+(8^2+9^2+12^2+13^2)+(10^2+11^2+14^2+15^2)}=\frac{458}{42+98+458+642}=0.3694,$$

and

$$\frac{E(w_3)}{\sum_{i=0}^{3}E(w_i)}=\frac{10^2+11^2+14^2+15^2}{(0^2+1^2+4^2+5^2)+(2^2+3^2+6^2+7^2)+(8^2+9^2+12^2+13^2)+(10^2+11^2+14^2+15^2)}=\frac{642}{42+98+458+642}=0.5177,$$

originate $\xi_1[\cdot]=\{0.0339,0.0790,0.3694,0.5177\}$;

- To obtain the second subvector, $\xi_2[\cdot]$, a total of $3\cdot 3=9$ non-overlapping windows, $w_0[\cdot][\cdot]=(0)$, $w_1[\cdot][\cdot]=(1)$, $w_2[\cdot][\cdot]=(2)$, $w_3[\cdot][\cdot]=(4)$, $w_4[\cdot][\cdot]=(5)$, $w_5[\cdot][\cdot]=(6)$, $w_6[\cdot][\cdot]=(8)$, $w_7[\cdot][\cdot]=(9)$, and $w_8[\cdot][\cdot]=(10)$, are positioned over $m[\cdot][\cdot]$, being its fourth row and fourth column discarded. Their normalised energies, i.e.,

$$\frac{E(w_0)}{\sum_{i=0}^{8}E(w_i)}=\frac{0^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2}=\frac{0}{327}=0,$$

$$\frac{E(w_1)}{\sum_{i=0}^{8}E(w_i)}=\frac{1^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2}=\frac{1}{327}=0.0031,$$

$$\frac{E(w_2)}{\sum_{i=0}^{8}E(w_i)}=\frac{2^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2}=\frac{4}{327}=0.0122,$$

$$\frac{E(w_3)}{\sum_{i=0}^{8} E(w_i)} = \frac{4^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2} = \frac{16}{327} = 0.0489,$$

$$\frac{E(w_4)}{\sum_{i=0}^{8} E(w_i)} = \frac{5^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2} = \frac{25}{327} = 0.0765,$$

$$\frac{E(w_5)}{\sum_{i=0}^{8} E(w_i)} = \frac{6^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2} = \frac{36}{327} = 0.1101,$$

$$\frac{E(w_6)}{\sum_{i=0}^{8} E(w_i)} = \frac{8^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2} = \frac{64}{327} = 0.1957,$$

$$\frac{E(w_7)}{\sum_{i=0}^{8} E(w_i)} = \frac{9^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2} = \frac{81}{327} = 0.2477,$$

and

$$\frac{E(w_8)}{\sum_{i=0}^{8} E(w_i)} = \frac{10^2}{(0)^2+(1)^2+(2)^2+(4)^2+(5)^2+(6)^2+(8)^2+(9)^2+(10)^2} = \frac{100}{327} = 0.3058,$$

originate $\xi_2[\cdot] = \{0, 0.0031, 0.0122, 0.0489, 0.0765, 0.1101, 0.1957, 0.2477, 0.3058\}$.

The concatenation of both sub-vectors produce $f[\cdot] = \{0.0339, 0.0790, 0.3694, 0.5177, 0, 0.0031, 0.0122, 0.0489, 0.0765, 0.1101, 0.1957, 0.2477, 0.3058\}$.

3.5. Numerical example for A_3 in 1D

Problem Statement: Let $s[\cdot] = \{1, 1, 2, 7, 10, 9, 8, 20\}$, implying in $M=8$. Assume that $C=20\%$ is the critical level. Obtain the feature vector, $f[\cdot]$, following method A_3.

Solution: The feature vector, of length $T = \frac{100}{20} - 1 = 4$, is composed by the proportional lengths of $s[\cdot]$ required to reach $1 \cdot 20\% = 20\%$, $2 \cdot 20\% = 40\%$, $3 \cdot 20\% = 60\%$, and $4 \cdot 20\% = 80\%$ of its total energy. They are obtained as follows:

- The energy of $s[\cdot]$ is $E(s[\cdot]) = 1^2+1^2+2^2+7^2+10^2+9^2+8^2+20^2 = 700$;
- 20% of $E(s[\cdot]) = 0.2 \cdot 700 = 140$. The energies of the first five elements are required in order to reach at least 140. Since $M=8$, the proportion of $s[\cdot]$ length covered is $\frac{5}{8} = 0.625$;
- 40% of $E(s[\cdot]) = 0.4 \cdot 700 = 280$. The energies of the first seven elements are required in order to reach at least 280. Since $M=8$, the proportion of $s[\cdot]$ length covered is $\frac{7}{8} = 0.875$;
- 60% of $E(s[\cdot]) = 0.6 \cdot 700 = 420$. The energies of all elements are required in order to reach at least 420. Since $M=8$, the proportion of $s[\cdot]$ length covered is $\frac{8}{8} = 1$;
- 80% of $E(s[\cdot]) = 0.8 \cdot 700 = 560$. The energies of all elements are required in order to reach at least 560. Since $M=8$, the proportion of $s[\cdot]$ length covered is $\frac{8}{8} = 1$;

Thus, $f[\cdot] = \{0.625, 0.875, 1, 1\}$.

3.6. Numerical example for A_3 in 2D

Problem Statement: Let $m[\cdot][\cdot] = \begin{pmatrix} 15 & 2 & 4 & 6 \\ 9 & 3 & 5 & 7 \\ 8 & 10 & 12 & 14 \\ 1 & 11 & 13 & 0 \end{pmatrix}$, implying in $N=4$ and $M=4$. Assume that $C=25\%$ is the critical level. Obtain the feature vector, $f[\cdot]$, following method A_3.

Solution: The feature vector, of length $T = \frac{100}{25} - 1 = 3$, is composed of the proportional areas of $m[\cdot][\cdot]$ required to reach $1 \cdot 25\% = 25\%$, $2 \cdot 25\% = 50\%$, and $3 \cdot 25\% = 75\%$ of its total energy. They are obtained as follows:

- The energy of $m[\cdot][\cdot]$ is $E(m[\cdot][\cdot]) = 15^2+2^2+4^2+6^2+9^2+3^2+5^2+7^2+8^2+10^2+12^2+14^2+1^2+11^2+13^2+0^2 = 1238$. Its area is $N \cdot M = 4 \cdot 4 = 16$;
- 25% of $E(m[\cdot][\cdot]) = 0.25 \cdot 1238 = 309.5$. The energies of the $(N=2)$ x $(M=2)$ rectangle, from $m_{0,0}$, are enough to reach at least 309.5. Since $N \cdot M = 2 \cdot 2 = 4$, the proportion of $m[\cdot][\cdot]$ area covered is $\frac{4}{16} = 0.2500$;
- 50% of $E(m[\cdot][\cdot]) = 0.50 \cdot 1238 = 619$. The energies of the $(N=3)$ x $(M=3)$ rectangle, from $m_{0,0}$, are enough to reach at least 619. Since $N \cdot M = 3 \cdot 3 = 9$, the proportion of $m[\cdot][\cdot]$ area covered is $\frac{9}{16} = 0.5625$;
- 75% of $E(m[\cdot][\cdot]) = 0.75 \cdot 1238 = 928.5$. The energies of the $(N=4)$ x $(M=4)$ rectangle, from $m_{0,0}$, are enough to reach at least 928.5. Since $N \cdot M = 4 \cdot 4 = 16$, the proportion of $m[\cdot][\cdot]$ area covered is $\frac{16}{16} = 1$;

Thus, $f[\cdot] = \{0.2500, 0.5625, 1\}$.

4. Example applications

In this section, 1D and 2D real data are used to evaluate the proposed approaches. The corresponding results and additional discussions are also presented.

4.1. Neurophysiological spike and overlap sorting

In prior work from 2003, I, in joint work with collaborators, applied the Discrete Spikelet Transform to distinguish seven electric patterns collected from H1, the motion-sensitive neuron of the fly's visual system, as documented in [84]. That experiment was carried out while the fly views a set of vertical random bars moving horizontally across its visual field and a platinum-iridium electrode captures the corresponding electric data that were digitalised at 44100 Hz, 16-bit, as exemplified in Figs. 8 and 9. Two implementations were possible: off-line, in a personal computer with C/C++ programming language, and real-time, with a dedicated Digital Signal Processor.

In the proposed experiment, the same neurophysiological data were used, but Spikelet was not applied. Instead, a feature vector solely based on signal energy was associated with HT, which consists of a modest and much simpler classifier, allowing for the identification of two patterns: spikes and overlaps. The former refers to signals that exhibit a shape similar to the one shown in Fig. 10(a) and the latter consists of signals composed by the sum of simultaneous spikes, possibly originated not only by H1 but also from its neighbouring neurons, as shown in Fig. 10(b,c). Identifying spikes and overlaps is important for neuroscientists during their study on how neurons react to evoked potentials [85–87].

As mentioned in [84], spikes and overlaps do not last more than $7.25 \cdot 10^{-4}$ s, implying that they fit a 32 samples-long window in the corresponding digitalised signal. Since this length is pre-defined, A_1, particularly with $L = 2 \cdot 32 = 64$ and $V = 50\%$, is the ideal method to be used. The combined choice for L, as being double the length of the searched patterns, and V, allowing each next window position to half-overlap the previous one, assures that such patterns are entirely located within some of the positions. The classifier, which evaluates the feature vector $f[\cdot]$ in groups of three sequential samples, adopts, for $(1 \leq i \leq T-2)$, the rule:

If $((f_i > f_{i-1})$ *and* $(f_i > f_{i+1})$ *and* $(f_i > t))$
a spike or an overlap exists in $s[\cdot]$ *between* $s_{32 \cdot i}$ *and* $s_{(32 \cdot i)+64}$.

By controlling t, that corresponds to a pre-defined HT adopted to avoid a mismatch of noise and patterns, the position of the spikes and overlaps is easily determined in the input signal $s[\cdot]$.

In order to distinguish between spikes and overlaps, a finer procedure is required. According to the definitions of the signals shapes, the 32 sample-long patterns for which the lowest amplitude is at the hth sample of the input signal $s[\cdot]$, start at their sample $h-13$ and end at their sample $h+18$. For both the ranges $[h-13; h]$ and $[h; h+18]$, spikes and overlaps cross the amplitude zero once. The first (FZC) and the

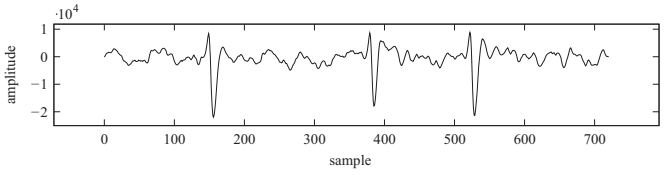


Fig. 8. Small part of the digitalised data from H1 in which three spikes can be noted around background noise.

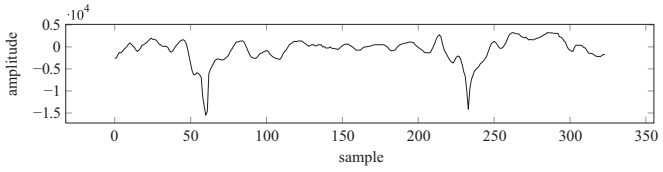


Fig. 9. Small part of the digitalised data from H1 in which two overlaps can be noted around background noise.

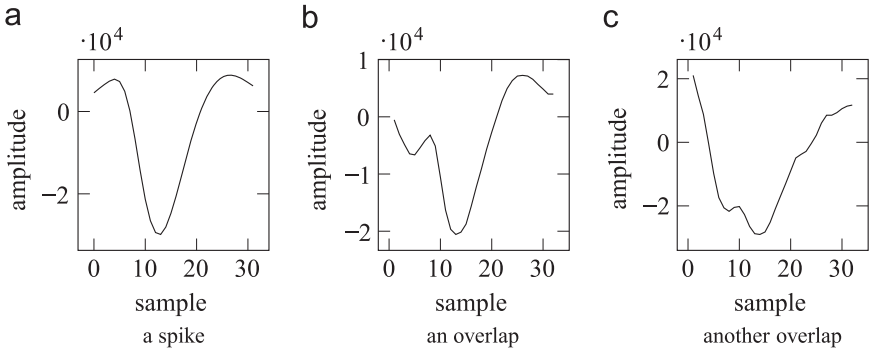


Fig. 10. (a) A typical spike captured from H1; (b,c) Common overlaps from H1 and its neighbouring neurons.

second zero-crossings (SZC) occur, respectively, within the former and the latter ranges. Based on this statement, the 64 sample-long windows positions for which a spike or overlap exist in $s[\cdot]$ are reinvestigated with $L=1$ and $V=0\%$, generating a new $T=64$ sample-long feature vector $\overline{f}[\cdot]$. Then, the following rule applies:

Integer $i \leftarrow$ *arg max* $\{\overline{f_k}\}$, for $(0 \le k \le 63)$.
If ($\overline{f}[\cdot]$ *continuously decreases within the range [FZC to i] and continuously increases within the range [i to SZC])*
  a spike was found in $s[\cdot]$ *within the range [FZC to SZC]*
else
  an overlap was found in $s[\cdot]$ *within the range [FZC to SZC].*

A total of 6325 spike and overlap candidates were found and their corresponding analysis produced $\frac{5180+1145}{6325}=100\%$ of accuracy, according to Table 1, which represents a confusion matrix and shows the potential of signal energy to characterise patterns even in association with a modest classification scheme. Comparing the proposed approach with the previous one [84], taking into account the current limitation, i.e., the capacity to distinguish between spikes and overlaps only, this modest energy-based algorithm is so accurate as the previous one. Clearly, the potential of A_1 to search for smooth or sudden signal variations has been exemplified.

4.2. Image texture classification

Textures [88–91], that are usually inspected based on a feature known as fractal dimension [92], may be interpreted as being a periodic set of energy concentrations, present in different resolutions. This suggests A_2 as being the most appropriate method to explore and characterise them.

In this experiment, Outex Texture Database [93] was partially used to distinguish among 8 fairly similar textures from *barleyrice* pattern. For each texture, 8 images, rotated 5, 10, 15, 30, 45, 60, 75, and 90 degrees, were used, totalling $8 \cdot 8 = 64$ images with 100-dpi resolution, stored in bmp format [50]. The particular textures used, downloaded from Outex website, are those included in the classes *barleyrice001*, *barleyrice002*, *barleyrice003*, *barleyrice004*, *barleyrice006*, *barleyrice008*, *barleyrice009*, and *barleyrice010*. These classes were chosen because they are particularly similar, providing effective tests to the proposed approach.

Randomly, raw data extracted from 4 images of each class were collected to create the template models, being the remaining 4 used to test the proposed approach. A total of $\binom{8}{4}^8 = \left(\frac{8!}{4!(8-4)!}\right)^8 = (70)^8$ cross-validation procedures, that represent all the possible ways to select 4 out of 8 images in 8 classes, were carried out for $Q=1, 2$, and 3, producing, respectively, feature vectors with $T=2^2=4$, $T=2^2+3^2=13$, and $T=2^2+3^2+5^2=38$ coefficients. To emphasise the role of signal energy, a fairly simple DM classifier was adopted: ED with T inputs, being one for each sample of the feature vectors, and one output. The distances from each testing signal to all the template models are registered, then, the class for which the lowest one belongs to, indicated by means of a numeric label that corresponds to the classifier output, characterises the assignment.

For the above-mentioned values of Q, the highest accuracies among the $(70)^8$ possible choices are listed in Table 2, suggesting that $Q=2$ is the best option for the proposed problem. For $Q=1$, the accuracy worsens, although not significantly, due to the incapacity to detect minor details. On the other hand, for $Q=3$, the accuracies persist at 100%, implying that finer analyses are not required in this specific case. The ith row and column of the confusion matrices presented in the table, for $(0 \le i \le 7)$, correspond, respectively, to the actual and predicted classifications, for the *barleyrice* classes listed above, following the order they were mentioned in the text.

Table 1
Results for the distinction between spikes and overlaps.

–	Predicted spikes	Predicted overlaps
Actual spikes	5180	0
Actual overlaps	0	1145

Table 2
Global accuracies for the texture recognition experiment.

Value of Q	1	2	3
confusion matrix	$\begin{pmatrix} 4&0&0&0&0&0&0&0\\ 0&4&0&0&0&0&0&0\\ 0&0&4&0&0&0&0&0\\ 0&0&1&3&0&0&0&0\\ 0&0&0&0&4&0&0&0\\ 0&0&0&0&0&4&0&0\\ 0&0&0&0&0&0&4&0\\ 0&0&0&0&0&0&0&4 \end{pmatrix}$	$\begin{pmatrix} 4&0&0&0&0&0&0&0\\ 0&4&0&0&0&0&0&0\\ 0&0&4&0&0&0&0&0\\ 0&0&0&4&0&0&0&0\\ 0&0&0&0&4&0&0&0\\ 0&0&0&0&0&4&0&0\\ 0&0&0&0&0&0&4&0\\ 0&0&0&0&0&0&0&4 \end{pmatrix}$	$\begin{pmatrix} 4&0&0&0&0&0&0&0\\ 0&4&0&0&0&0&0&0\\ 0&0&4&0&0&0&0&0\\ 0&0&0&4&0&0&0&0\\ 0&0&0&0&4&0&0&0\\ 0&0&0&0&0&4&0&0\\ 0&0&0&0&0&0&4&0\\ 0&0&0&0&0&0&0&4 \end{pmatrix}$
% of accuracy	$\frac{4+4+4+3+4+4+4+4}{32}=\frac{31}{32}=96.875\%$	$\frac{4+4+4+4+4+4+4+4}{32}=\frac{32}{32}=100\%$	$\frac{4+4+4+4+4+4+4+4}{32}=\frac{32}{32}=100\%$

Despite the modest size of the dataset, i.e., 64 images, the results are highly relevant because the proposed feature vector based on A_2 was able to capture the minor differences among the textures. Once again, signal energy was successfully employed in this pattern recognition task, being comparable with the current state-of-the-art approaches, such as the one documented in [94].

4.3. Text-dependent speaker verification

The objective of a speaker verification (SV) algorithm is to confirm, based on a non-speech pre-identification followed by a voice-based biometric analysis, whether or not a claimed speaker is authentic [95–99]. Text-dependent, in which there is a pre-defined utterance, and text-independent, in which the speakers are free to choose their spoken words, are the existing approaches for SV. Intra and inter-speaker variations, respectively characterising the differences among the voices of a same speaker uttering the same sentence and the differences among the voices of different speakers uttering the same sentence, are the relevant aspects for an SV algorithm. The more it is insensitive to intra-speaker variations, the lower the risk of rejecting a genuine speaker. On the other hand, the more it is sensitive to inter-speaker variations, the lower the risk of validating an impostor [99]. Consequently, feature vectors for SV should present, at most, minor changes among utterances extracted from a same speaker, and, necessarily, major differences when they come from different ones.

In this experiment, I present, graphically, the results of a text-dependent SV algorithm in which raw data from 40 speakers, $(S_0, \ldots, S_{39})$, enrolled in the TIMIT database [100], uttering the phonetically rich sentence "*She had your dark suit in greasy wash water all year*", were adopted. $S_0, \ldots, S_{39}$ are the speakers for whom the voices are, respectively, the 11 ones, the 26 ones, and the first 3 ones in the folders */timit/test/dr1*, */timit/test/dr2*, and */timit/test/dr3*. Forty is the limit I imposed to the number of speakers just to allow a confortable visualisation of the results, as presented ahead.

A_1 is not the ideal method for this application due to the fact that the speakers' utterances have variable lengths. On the other hand, both A_2 and A_3 generate feature vectors with pre-defined lengths, however, the latter is preferable for SV because it expresses the effort the speakers' lungs performed to utter, as a function of time, characterising their personalities. This contrasts with current state-of-the-art techniques that retrieve the speakers' vocal tract resonance frequencies to authenticate them [99].

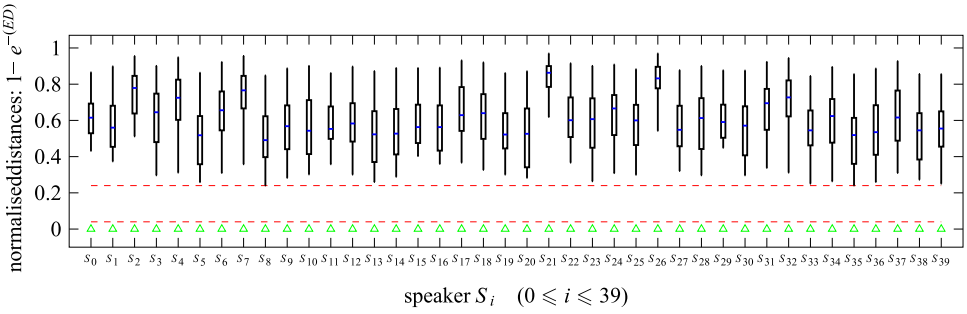


Fig. 11. Results of the proposed all-against-all speaker verification algorithm for $C=1\%$, which produces a 99 sample-long feature vector.

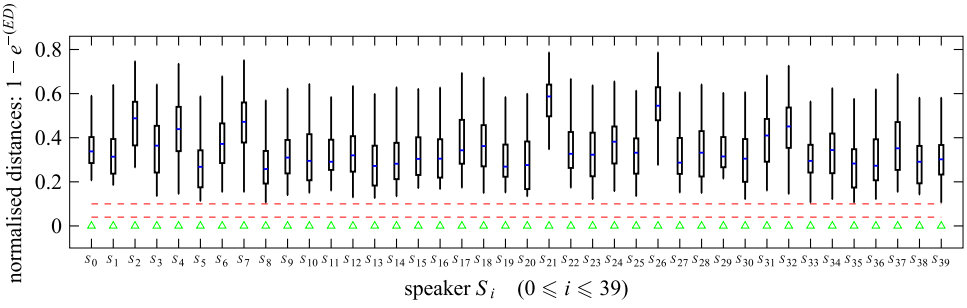


Fig. 12. Results of the proposed all-against-all speaker verification algorithm for $C=5\%$, which produces a 19 sample-long feature vector.

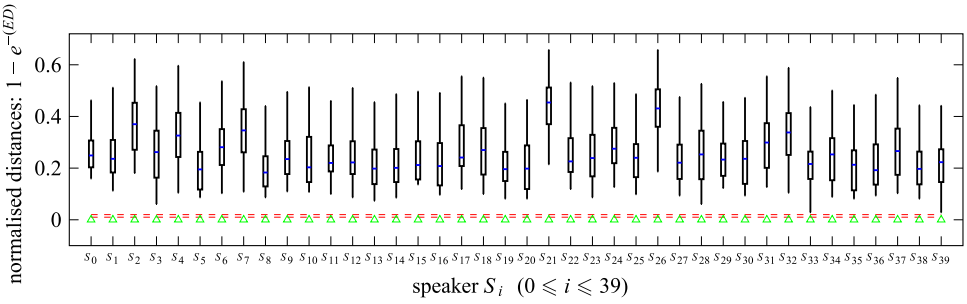


Fig. 13. Results of the proposed all-against-all speaker verification algorithm for $C=10\%$, which produces a 9 sample-long feature vector.

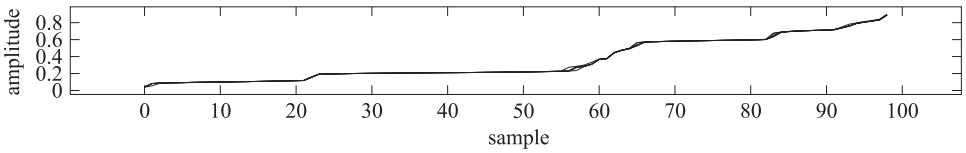


Fig. 14. Six feature vectors, extracted with A_3 using $C=1\%$, from the original and modified speech signals related to the speaker */timit/test/dr1/faks0sa*.

An ordinary ED classifier was associated with A_3 to compare each speaker's feature vector against all the other speakers' ones, being the results registered in Figs. 11–13, respectively for $C=1\%$, $C=5\%$, and $C=10\%$. The correct assignments, represented with green triangles, and the impostors, represented with box plots, confirm that, as C decreases, the separation margins, represented with red dashed lines, become larger, consequently reducing the risk of authenticating an impostor. For better visualisation, the EDs are normalised in the figures before being plotted, so that ($0 \leq EDs \leq 1$), as indicated in the axes. Particularly, in Fig. 13, the linear separation margin between impostors and genuine speakers is minor, as imposed by the speakers S_{33} and S_{39}. This is due to the fact that, for $C=10\%$, $f[\cdot]$ is only a $T=9$ sample-long feature vector, hindering a precise analysis. On the other hand, for $C=1\%$, $f[\cdot]$ becomes a $T=99$ sample-long vector, offering a much finer analysis that increases the barrier width, as can be seen in Fig. 11, i.e., as C decreases, inter and intra-speaker variations are, respectively, major and minor among the feature vectors. Thus, a low C associated with a simple HT authenticate the speakers accurately, implying that the proposed approach is, preserving due proportions, comparable to the existing ones, such as [101–103].

TIMIT contains only one sample of each speaker's utterance, for a given sentence, such as the one chosen for this experiment. Thus, in order to simulate intra-speaker changes, *Audacity*[1] was used to synthesise five similar speech signals from each original one. They correspond to positive and negative amplitude gains of 12.6 dB, to the increase and the decrease of 5% in fluency speed, and to a bass boost of 12 dB around 200 Hz. To exemplify the outcomes, Fig. 14 shows the superimposed plots of the six corresponding feature vectors, i.e., the one extracted from the original speech and the five ones extracted from the synthesised signals, all using $C=1\%$, for the sentence uttered by the speaker */timit/test/dr1/faks0sa*. Clearly, the plots are almost indistinguishable, i.e., A_3 is practically insensitive to the above-mentioned variabilities. All the other remaining speakers' plots, that were not shown due to the lack of space, exhibit the same pattern, revealing, at most, minor differences.

Undoubtedly, signal energy has shown its applicability, based on A_3, to carry out this task.

5. Conclusions

In this tutorial, three methods for feature extraction based on signal energy were presented. A_1, the simplest one, is ideal to search for a specific event or characteristic in a digital signal, such as a biological pattern of pre-defined length. A_2, for which an application on image texture recognition was shown, is employed to analyse how energy is distributed along the time, or space, in different levels of resolution. Lastly, A_3, which was exemplified for speaker verification, is the best method to quantify the partial workloads performed by the agent responsible for its production as time or space advances. Numerical hypothetical examples, both in 1D and 2D, were included in the text to complement the theoretical explanations.

DMs and HTs, which characterise the simplest existing classifiers, were purposely associated with the feature vectors only to demonstrate the potential of signal energy, being that the reason why this study refrains from comparing the association of different classifiers with the proposed features. Furthermore, in this text, designed to serve as a tutorial, I draw the DSP-PR scientific communities's attention to consider the use of signal energy, intending a balance among *creativity*, *simplicity*, and *accuracy*, prior to adopt more complicated options. Obviously, signal energy is not the best option to solve all the existing classification issues, however, a careful analysis on related literature reveals that features extracted on its basis could successfully simplify or improve a considerable number of current DSP algorithms for PR. The example applications described in the previous section clearly reflect this scenario.

Although each one of the proposed approaches was properly designed for a particular type of task, with advantages and disadvantages, I ask your attention: any classification system requires a joint planning and evaluation, i.e., a specific method for feature extraction and a set of features are never definitely the best or the worst for a given problem if the classifier, used in conjunction, is not taken into account. Nevertheless, as demonstrated by the tests and results, signal energy can certainly be considered a useful feature for pattern recognition in neurophysiological, speech, and image processing, among others, even associated with modest classifiers. All the data used during the experiments are available to the scientific community upon prior request[2] so that anyone can repeat the procedures.

Further research involving signal energy should focus on variations of the proposed approaches so that more particular problems could be conveniently addressed. Additional studies can also include the association of features based on signal energy with knowledge-based algorithms dedicated to solve non-linear classification problems. An interesting open discussion: are there simpler features than signal energy that can achieve similar or better results when associated with modest classifiers? For what kind of signals?

Acknowledgements

I am very grateful to CNPQ - "Conselho Nacional de Pesquisa e Desenvolvimento", in Brazil, for the grants provided, through the process 306811/2014-6, to support this research.

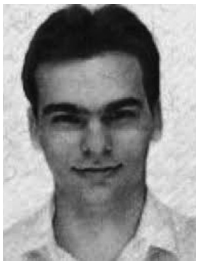
[1] *Audacity application software (http://www.audacityteam.org/)*

[2] Send requests to *guido@ieee.org*

References

[1] S. Haykin, B.V. Veen, Signals and Systems, 2nd ed., Wiley, New Jersey, USA, 2002.
[2] S. Theodoridis, K. Koutroumbas, Pattern Recognition, 4th ed., 2008.
[3] A.R. Webb, K.D. Copsey, Statistical Pattern Recognition, 3rd ed., Wiley, West Sussex, UK, 2011.
[4] S. Arora, Computational Complexity: a modern approach, Cambridge University Press, Cambridge, UK, 2009.
[5] B. Stroustrup, The C++ Programming Language, 4th ed., Addison-Wesley Professional, 2013.
[6] S. He, Energy-to-peak filtering for T–S fuzzy systems with Markovian jumping: the finite-time case, Neurocomputing 168 (2015) 348–355.
[7] X. Zhang, Y. Li, Adaptive energy detection for bird sound detection in complex environments, Neurocomputing 155 (2015) 108–116.
[8] Y.-L. Hsu, Y.-T. Yang, J.S. Wang, C.-Y. Hsu, Automatic sleep stage recurrent neural classifier using energy features of EEG signals, Neurocomputing 104 (2013) 105–114.
[9] J.F. Ramirez-Villegas, D.F. Ramirez-Moreno, Wavelet packet energy, Tsallis entropy and statistical parameterization for support vector-based and neural-based classification of mammographic regions, Neurocomputing 77 (2012) 82–100.
[10] C.-D. Wang, J.-H. Lai, Energy based competitive learning, Neurocomputing 74 (12–13) (2011) 2265–2275.
[11] M. Kotti, K.I. Diamantaras, Efficient binary classification through energy minimisation of slack variables, Neurocomputing 148 (2015) 498–511.
[12] H.H. Bafroui, A. Ohadi, Application of wavelet energy and Shannon entropy for feature extraction in gearbox fault detection under varying speed conditions, Neurocomputing 133 (2014) 437–445.
[13] S. Squartini, E. Principi, R. Rotili, F. Piazza, Environmental robust speech and speaker recognition through multi-channel histogram equalization, Neurocomputing 78 (1) (2012) 111–120.
[14] J. Liu, E. Lughofer, X. Zeng, Could linear model bridge the gap between low-level statistical features and aesthetic emotions of visual textures? Neurocomputing 168 (2015) 947–960.
[15] Z. Tang, Y. Su, M.J. Er, F. Qi, L. Zhang, J. Zhou, A local binary pattern based texture descriptors for classification of tea leaves, Neurocomputing 168 (2015) 1011–1023.
[16] Y. Han, C. Xu, G. Baciu, M. Li, Lightness biased cartoon-and-texture decomposition for textile image segmentation, Neurocomputing 168 (2015) 575–587.
[17] T.P. Nguyen, N-S. Vu, A. Manzanera, Statistical binary patterns for rotational invariant texture classification, Neurocomputing 173 (2016) 1565–1577.
[18] L. Nanni, M. Melucci, Combination of projectors, standard texture descriptors and bag of features for classifying images, Neurocomputing 173 (2016) 1602–1614.
[19] F. Yang, G.*S. Xia, G. Liu, L. Zhang, X. Huang, Dynamic texture recognition by aggregating spatial and temporal features via ensemble SVMs, Neurocomputing 173 (2016) 1310–1321.
[20] L. Wang, H. Liu, F. Sun, Dynamic texture video classification using extreme learning machine, Neurocomputing 174 (2016) 278–285.
[21] L. Goela, D. Gupta, V.K. Panchal, Hybrid bio-inspired techniques for land cover feature extraction: a remote sensing perspective, Appl. Soft Comput. 12 (2) (2012) 832–849.
[22] N.L.A. Krisshna, V.K. Deepak, K. Manikantan, S. Ramachandran, Face recognition using transform domain feature extraction and PSO-based feature selection, Appl. Soft Comput. 22 (2014) 141–161.
[23] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, 2nd ed., Wiley, New Jersey, USA, 2014.
[24] C.C. Aggarwal, Data Classification: Algorithms and Applications, Chapman and Hall, Boca Raton, USA, 2014.
[25] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, 2nd ed., Wiley-Interscience, New York, USA, 2000.
[26] G. Dougherty, Pattern Recognition and Classification: An Introduction, Springer, New York, USA, 2012.
[27] M.-C. Yanga, D.-G. Leeb, S.-Y. Parkc, H.C. Rim, Knowledge-based question answering using the semantic embedding space, Expert Syst. Appl. 42 (23) (2015) 9086–9104.
[28] S. Haykin, Neural Networks and Learning Machines, 3rd ed., Prentice Hall, New Jersey, USA, 2008.
[29] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, Singapore, 2007.
[30] S. Theodoridis, A. Pikrakis, K. Koutroumbas, D. Cavouras, Introduction to Pattern Recognition: A Matlab Approach, Academic Press, Oxford, UK, 2010.
[31] P. Flach, Machine Learning: The Art and Science of Algorithms That Make Sense of Data, Cambridge University Press, New York, USA, 2012.
[32] C.-S. Cheng, K.-K. Huang, P.-W. Chen, Recognition of control chart patterns using a neural network-based pattern recognizer with features extracted from correlation analysis, Pattern Anal. Appl. 18 (1) (2015) 75–86.
[33] B. Bonev, F. Escolano, M. Cazorla, Feature selection, mutual information, and the classification of high-dimensional patterns, Pattern Anal. Appl. 11 (3) (2008) 309–319.
[34] F. Zou, Y. Wang, Y. Yang, K. Zhou, Y. Chen, J. Song, Supervised feature learning via l_2-norm regularized logistic regression for 3D object recognition, Neurocomputing, 151 (part 2) (2015) 603–611.
[35] L. Deng, Dong. Yu, Deep Learning: Methods and Applications (Foundations and Trends in Signal Processing - Book 20), Now Publishers Inc., The Netherlands, 2014.
[36] C.-D. Caleanua, X. Maob, G. Pradelc, S. Mogad, Y. Xueb, Combined pattern search optimization of feature extraction and classification parameters in facial recognition, Pattern Recognit. 32 (9) (2011) 1250–1255.
[37] D. Zhanga, J. Heb, Y. Zhaoa, Z. Luoc, M. Dud, Global plus local: a complete framework for feature extraction and recognition, Pattern Recognit. 47 (3) (2014) 1433–1442.
[38] Y.-H. Yuana, Q.-S. Sunb, Graph regularized multiset canonical correlations with applications to joint feature extraction, Pattern Recognit. 47 (12) (2014) 3907–3919.
[39] J. Luo, W. Wang, H. Qi, Spatio-temporal feature extraction and representation for RGB-D human action recognition, Pattern Recognit. 50 (2014) 139–148.
[40] W. Yanga, Z. Wanga, C. Suna, A collaborative representation based projections method for feature extraction, Pattern Recognit. 48 (1) (2015) 20–27.
[41] Y.A. Ghassabeha, F. Rudzicza, H.A. Moghaddamc, Fast incremental LDA feature extraction, Pattern Recognit. 48 (6) (2015) 1999–2012.
[42] S. Maldonadoa, J. Péreza, R. Weberb, M. Labbéc, Feature selection for support vector machines via mixed integer linear programming, Inf. Sci. 279 (2014) 163–175.
[43] J.M. Chenlo, D.E. Losada, An empirical study of sentence features for subjectivity and polarity classification, Inf. Sci. 280 (2014) 275–288.
[44] X. Han, X. Chang, L. Quan, X. Xiong, J. Li, Z. Zhang, Y. Liu, Feature subset selection by gravitational search algorithm optimization, Inf. Sci. 281 (2014) 128–146.
[45] V. Bolón-Canedoa, N. Sánchez-Maroñoa, A. Alonso-Betanzosa, J.M. Benítezb, F. Herrerab, A review of microarray datasets and applied feature selection methods, Inf. Sci. 282 (2014) 111–135.
[46] A.S. Britto Jr, R. Sabourinc, L.E.S. Oliveira, Dynamic selection of classifiers—a comprehensive review, Pattern Recognit. 47 (11) (2014) 3665–3680.
[47] E. Sejdic, I. Djurovic, J. Jiang, Time-frequency feature representation using energy concentration: an overview of recent advances, Digit. Signal Process. (19) (2009) 153–183.
[48] L. Deng, D. O'Shaughnessy, Speech Processing: A Dynamic and Optimization-Oriented Approach, CRC Press, New York, USA, 2003.
[49] P. Addison, J. Walker, R.C. Guido, Time-frequency Analysis of Biosignals, IEEE Eng. Biol. Med. Mag. 28 (5) (2009) 14–29.
[50] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis, and Machine Vision, 4th ed., Cengage Learning, Stanford, USA, 2014.
[51] M. Liang, H. Faghidi, Intelligent bearing fault detection by enhanced energy operator, Expert Syst. Appl. 41 (16) (2014) 7223–7234.
[52] Y. Zheng, X. Guo, X. Ding, A novel hybrid energy fraction and entropy-based approach for systolic heart murmurs identification, Expert Syst. Appl. 42 (5) (2015) 2710–2721.
[53] S.K. Wajid, A. Hussain, Local energy-based shape histogram feature extraction technique for breast cancer diagnosis, Expert Syst. Appl. 42 (20) (2015) 6990–6999.
[54] M.M. Sabarimalai, S. Dandapat, Wavelet energy based diagnostic distortion measure for ECG, Biomed. Signal Process. Control (2) (2007) 80–96.
[55] C. Kamath, ECG beat classification using features extracted from Teager energy functions in time and frequency domains, IET Signal Process. 5 (6) (2011) 575–581.
[56] Y. Zou, J. Han, S. Xuan, S. Huang, S. Weng, D. Fang, X. Zeng, An energy-efficient design for ECG recording and R-Peak detection based on wavelet transform, IEEE Trans. Circuits Syst.-II: Express Briefs 62 (2) (2015) 119–123.
[57] Y. Hsu, Y. Yang, J. Wang, C. Hsu, Automatic sleep stage recurrent neural classifier using energy features of EEG signals, Neurocomputing 104 (2013) 105–114.
[58] O. Faust, W. Yu, N.A. Kadri, Computer-based identification of normal and alcoholic EEG signals using wavelet-packets and energy measures, J. Mech. Med. Biol. 13 (3) (2013) 1–17.
[59] S. Bhattacharyya, A. Biswas, J. Mukherjee, A.K. Majumdar, B. Majumdar, S. Mukherjee, A.K. Singh, Detection of artifacts from high energy bursts in neonatal EEG, Comput. Biol. Med. 43 (2013) 1804–1814.
[60] A. Moujahid, S. DAnjou, M. Grana, Energy demands of diverse spiking cells from the neocortex, hippocampus, and thalamus, Front. Comput. Neurosci. 8 (2014) 1–12.
[61] H.I. Koo, N.I. Cho, Extraction in handwritten chinese documents based on an energy minimization framework, IEEE Trans. Image Process. 21 (3) (2012) 1169–1175.
[62] B.P. Chacko, V.R.V. Krishnan, G. Raju, P.B. Anto, Handwritten character recognition using wavelet energy and extreme learning machine, Int. J. Mach. Learn. Cybern. (3) (2012) 149–161.
[63] C.A. Boiangiu, M.C. Tanase, R. Ioanitescu, Handwritten documents text line segmentation based on information energy, Int. J. Comput. Commun. 9 (1) (2014) 8–15.
[64] Z. Ji, Y. Xia, Q. Sun, G. Cao, Q. Chen, Active contours driven by local likelihood image fitting energy for image segmentation, Inf. Sci. (301) (2015) 285–304.
[65] A.V.Y. Phamila, R. Amutha, Energy-efficient low bit rate image compression in wavelet domain for wireless image sensor networks, Electron. Lett. 51 (11) (2015) 824–826.

[66] Y. Yang, Q. Liu, H. Liu, L. Yu, F. Wang, Dense depth image synthesis via energy minimization for three-dimensional video, Signal Process. (112) (2015) 199–208.
[67] J. Yang, L. Guo, H. Yang, A new multi-focus image fusion algorithm based on BEMD and improved local energy, IEEJ Trans. Electr. Electron. Eng. 10 (2015) 447–452.
[68] M. Kos, M. Grasic, Z. Kacic, Online speech/music segmentation based on the variance mean of filter bank energy, EURASIP J. Adv. Signal Process. 2009 (2009) 1–13.
[69] M. Rahmani, N. Yousefian, A. Akbari, Energy-based speech enhancement technique for hands-free communication, Electron. Lett. 45 (1) (2009) 1–2.
[70] Y. Litvin, I. Cohen, D. Chazan, Monaural speech/music source separation using discrete energy separation algorithm, Signal Process. (90) (2010) 3147–3163.
[71] D. Dimitriadis, P. Maragos, A. Potamianos, On the effects of filterbank design and energy computation on robust speech recognition, IEEE Trans. Audio Speech Lang. Process. 19 (6) (2015) 1504–1516.
[72] L. Shen, C. Yeh, S. Hwang, A study on the consistency analysis of energy parameter for Mandarin speech, EURASIP J. Audio Speech Music Process. 2012 (2012) 1–28.
[73] P. Jain, R.B. Pachori, Marginal energy density over the low frequency range as a feature for voiced/non-voiced detection in noisy speech signals, J. Frankl. Inst. (350) (2013) 698–716.
[74] D. Du, K. Odame, An energy-efficient spike encoding circuit for speech edge detection, Analog Integr. Circuits Signal Process. (75) (2013) 447–458.
[75] T.F. Sanam, C. Shahnaz, A semisoft thresholding method based on Teager energy operation on wavelet packet coefficients for enhancing noisy speech, EURASIP J. Audio Speech Music Process. (25) (2013) 1–15.
[76] M. Mahdikhani, M.H. Kahaei, An introduction to energy-based blind separating algorithm for speech signals, ETRI J. 36 (1) (2014) 175–178.
[77] C. Heinricha, F. Schiel, The influence of alcoholic intoxication on the short-time energy function of speech, J. Acoust. Soc. Am. 5 (135) (2014) 2942–2951.
[78] C.H. Taal, R.C. Hendriks, R. Heusdens, Speech energy redistribution for intelligibility improvement in noise based on a perceptual distortion measure, Comput. Speech Lang. (28) (2014) 858–872.
[79] M. Al-Akaidi, Fractal Speech Processing, Cambridge University Press, Cambridge, UK, 2004.
[80] J.V. Stone, Information Theory: A Tutorial Introduction, Sebtel Press, Sheffield, UK, 2015.
[81] A.V. Oppenheim, R.W. Schafer, Discrete-time Signal Processing, 3rd ed., Prentice-Hall, New Jersey, USA, 2009.
[82] R.C. Guido, P. Addison, J. Walker, Introducing wavelets and time-frequency analysis, IEEE Eng. Biol. Med. Mag. 28 (5) (2009) 13.
[83] V.K. Madisetti, The Digital Signal Processing Handbook, 2nd ed., CRC Press, Boca Raton, USA, 2009.
[84] R.C. Guido, J.F.W. Slaets, R. Koberle, L.O.B. Almeida, J.C. Pereira, A new technique to construct a wavelet transform matching a specified signal with applications to digital, real time, spike and overlap pattern recognition, Digit. Signal Process. 16 (1) (2006) 24–44.
[85] L. Vezarda, P. Legranda, M. Chaventa, F. Faita-Ainsebac, L. Trujillod, EEG classification for the detection of mental states, Appl. Soft Comput. 32 (2015) 113–131.
[86] H.S. Lopes, Genetic programming for epileptic pattern recognition in electroencephalographic signals, Appl. Soft Comput. 7 (1) (2007) 343–352.
[87] S.R. Devasahayam, Signals and Systems in Biomedical Engineering: Signal Processing and Physiological Systems Modelling, Springer, New York, USA, 2000.
[88] A.R. Yadava, R.S. Ananda, M.L. Dewala, S. Guptab, Multiresolution local binary pattern variants based texture feature extraction techniques for efficient classification of microscopic images of hardwood species, Appl. Soft Comput. 32 (2015) 101–112.
[89] Y. Kayaa, O.F. Ertugrulb, R. Tekinc, Two novel local binary pattern descriptors for texture analysis, Appl. Soft Comput. 34 (2015) 728–735.
[90] Y. He, N. Sang, C. Gao, Multi-structure local binary patterns for texture classification, Pattern Anal. Appl. 16 (4) (2013) 595–607.
[91] R.F.C. Guerreiro, P.M.Q. Aguiar, Optimized filters for efficient multi-texture discrimination, Pattern Anal. Appl. 18 (1) (2015) 61–73.
[92] A.-M. Saad, G.E. Loay, N.K. Raid, Texture Analysis Using Fractal, Wavelet & Cubic Spline Representations, LAP Lambert Academic Publishing, Saarbrücken, Germany, 2015.
[93] Outex Image texture dataset. Available at: ⟨http://www.outex.oulu.fi/⟩.
[94] J.Q. Shen, G.G. Li, X. Zou, Y. Li, Fabric weave pattern's recognition based on texture orientation features, in: Applied Materials and Technologies for Modern Manufacturing, Book Series: Applied Mechanics and Materials, vol. 423–426, 2013, pp. 2404–2408.
[95] K. Daqrouq, T.A. Tutunji, Speaker identification using vowels features through a combined method of formants, wavelets, and neural network classifiers, Appl. Soft Comput. 27 (2015) 231–239.
[96] S. Sarkar, K.S. Rao, Stochastic feature compensation methods for speaker verification in noisy environments, Appl. Soft Comput. 19 (2014) 198–214.
[97] M. Pal, G. Saha, On robustness of speech based biometric systems against voice conversion attack, Appl. Soft Comput. 30 (2015) 214–228.
[98] R. Justo, M.I. Torres, Integration of complex language models in ASR and LU systems, Pattern Anal. Appl. 18 (3) (2015) 493–505.
[99] H. Beiji, Fundamentals of Speaker Recognition, Springer, New York, 2011.
[100] TIMIT speech corpus. Linguistic Data Consortium (LDC) ⟨https://catalog.ldc.upenn.edu/LDC93S1⟩.
[101] A. Larcher, K.A. Lee, B. Ma, H.Z. Li, Text-dependent speaker verification: classifiers, databases and RSR2015, Speech Commun. 60 (2014) 56–77.
[102] K.K. George, C.S. Kumar, K.I. Ramachandran, A. Panda, Cosine distance features for improved speaker verification, IEEE Electron. Lett. 51 (12) (2015) 939–940.
[103] N. Li, M.W. Mak, SNR-Invariant PLDA modeling in nonparametric subspace for robust speaker verification, IEEE-ACM Trans. Audio Speech Lang. Process. 23 (10) (2015) 1648–1659.



Rodrigo Capobianco Guido received his B.Sc. degrees in Computer Science and in Computer Engineering, his M.Sc. degree in Electrical Engineering, and his Ph.D. degree in Computational Applied Physics, respectively from São Paulo State University (UNESP) at São José do Rio Preto, Brazil, from Educational Foundation at Votuporanga (FEV), Brazil, in 1998, from Campinas State University (UNICAMP), Brazil, in 2000, and from University of São Paulo at São Carlos (USP), Brazil, in 2003, all titles focusing on signal processing. He has already participated in two postdoctoral programs in signal processing at USP, from 2003 to 2007 and obtained the title of Associate Professor (Livre-Docência) in Digital Signal Processing from the School of Engineering at São Carlos – USP, in 2008. He has taught signal processing and electronics since 1999 and has published scientific articles in IEEE and Elsevier journals and magazines plus papers in international and national conferences. He has served or is serving as guest-editor for many special issues of IEEE and Elsevier journals, and as organizer and chairman for IEEE conferences, including IEEE ISM, IEEE ICSC, plus others. He is a member of the editorial board of respected scientific journals, including IEEE Signal Processing Magazine, IEEE Transactions on Audio, Speech, and Language Processing, Elsevier Neurocomputing, and the International Journal of Semantic Computing, and has also served as a reviewer for many IEEE and Elsevier journals and conferences. He received several grants and awards from the State of São Paulo Research Foundation (FAPESP), from National Council of Research and Development (CNpQ), and Ministry of Education and Culture (MEC) of Brazil. He has supervised many theses in his field and is a senior member of the IEEE. The main focus of Guido's team, established at The São Paulo State University (UNESP), is concentrated in digital speech processing, i.e., speech recognition, speech analysis for biomedical purposes, speaker identification and verification, voice morphing, and speech compression, specially based on wavelets associated with machine learning techniques.