MethodsX 8 (2021) 101314

Contents lists available at ScienceDirect

MethodsX

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

Method Article

Construction of wavelet dictionaries for ECG modeling

Dana Černá[a,*], Laura Rebollo-Neira[b]

[a] Department of Mathematics and Didactics of Mathematics, Technical University of Liberec, Studentská 2, Liberec, Czech Republic

[b] Mathematics Department, Aston University, Birmingham B3 7ET, UK

A B S T R A C T

Technical details, algorithms, and MATLAB implementation for a method advanced in the paper "Wavelet Based Dictionaries for Dimensionality Reduction of ECG Signals", are presented. This work aims to be the companion of that publication, in which an adaptive mathematical model for a given ECG record is proposed. The method comprises the following building blocks.

(i) Construction of a suitable redundant set, called 'dictionary', for decomposing an ECG signal as a superposition of elementary components, called 'atoms', selected from that dictionary.
(ii) Implementation of the greedy strategy Optimized Orthogonal Matching Pursuit (OOMP) for selecting the atoms intervening in the signal decomposition.

This paper gives the details of the algorithms for implementing stage (i), which is not fully elaborated in the previous publication. The proposed dictionaries are constructed from known wavelet families, but translating the prototypes with a shorter step than that corresponding to a wavelet basis. Stage (ii) is readily implementable by the available function OOMP.

- The use of the software and the power of the technique is illustrated by reducing the dimensionality of ECG records taken from the MIT-BIH Arrhythmia Database.
- The MATLAB software has been made publicly available on a dedicated website.
- We provide the explanations, algorithms and software for the construction of scaling functions and wavelet prototypes for 17 different wavelet families. The procedure is designed to allow for straightforward extension of the software by the inclusion of additional options for the wavelet families.

* Corresponding author.
E-mail address: dana.cerna@tul.cz (D. Černá).

https://doi.org/10.1016/j.mex.2021.101314

ARTICLE INFO
Method name: Software for Constructing Wavelet Dictionaries with Application to ECG Signal, Modeling
Keywords: ECG signal, Sarse representation, Dimensionality reduction, Wavelet, Dictionary, Optimized orthogonal matching pursuit
Article history: Received 23 November 2020; Accepted 15 March 2021; Available online 23 March 2021

Specifications table

Subject Area:	Medicine and Dentistry
More specific subject area:	*Biomedical Engineering*
Method name:	*Software for Constructing Wavelet Dictionaries with Application to ECG Signal Modeling*
Name and reference of original method:	*L. Rebollo-Neira, D. Černá, Wavelet Based Dictionaries for Dimensionality Reduction of ECG Signals Biomedical Signal Processing and Control 54 (2019), article No. 101593. doi:*https://doi.org/10.1016/j.bspc.2019.101593.
Resource availability:	*MATLAB software for construction of dictionaries and ECG signal modeling:* www.nonlinear-approx.info/examples/node013.html

Method details

The electrocardiogram (ECG) represents electrical activity of human heart and is widely applied to the diagnosis of heart diseases. Sparse representation of ECG signals is a subject of interest in different applications such as analysis, compression, and classification of ECG signals. In [17] we proposed a method for ECG modeling which proceeds as follows.

Assuming that the signal is given as an N-dimensional array, this array is partitioned into Q cells $\boldsymbol{f}^{\mathrm{c}}\{q\}, q = 1, \ldots, Q$. Thus, each cell $\boldsymbol{f}^{\mathrm{c}}\{q\}$ is an N_b-dimensional vector, which is modeled by an 'atomic decomposition' of the form

$$\boldsymbol{f}^{\mathrm{a}}\{q\} = \sum_{n=1}^{k(q)} \boldsymbol{c}\{q\}(n)\boldsymbol{d}_{\ell\{q\}(n)}. \tag{1}$$

For each cell q, the atoms $\boldsymbol{d}_{\ell_{\{q\}(n)}}$, $n = 1, \ldots, k(q)$ are selected from a dictionary through the greedy OOMP algorithm [18],[19]. The array $\ell\{q\}$ is a vector whose components $\ell\{q\}(n)$, $n = 1, \ldots, k(q)$ contain the indices of the selected atoms for decomposing the q-th cell in the signal partition. The OOMP method, for selecting these indices and computing the corresponding coefficients $\boldsymbol{c}\{q\}(n)$, $n = 1, \ldots, k(q)$ in (1), is fully implemented by the OOMP function included as a tool of the software.

In this Communication we focus on the details for the construction of the dictionaries giving rise to the piecewise model (1) for ECG signals. The aim is to support the task of dimensionality reduction. In this respect, each dictionary improves the results achieved with the wavelet basis of the same family [17],[20]. Consequently, using a dictionary the ECG record is represented by significantly fewer elementary components than those required by the corresponding wavelet basis for reproducing the signal at the same quality. The gain in dimensionality reduction enhances compression performance. As demonstrated in [17] the results arising from the proposed wavelet dictionaries attain compression ratios distinctly improving upon previously reported benchmarks [[13–15],[24] for compressing the whole MIT-BIH Arrhythmia data set [8]. The suitability of using dictionaries for ECG signal modeling goes beyond compression applications. Indeed, in [16] Gabor dictionaries are used with success for automated recognition of cardiac arrhythmias. In this regard, we expect that our software contribute to the development of advanced tools for the automation of ECG interpretation.

Each of the proposed dictionaries consists of two components. One of the components contains a few elements, say M_c, from a discrete cosine basis. This component allows for the fact that ECG signals are normally superimposed to a smooth background. It is given as a $N_b \times M_c$ matrix $\boldsymbol{D}^C$. The other component is the wavelet dictionary, which is given as a $N_b \times M_w$ matrix $\boldsymbol{D}^W$. Thus, the whole dictionary $\boldsymbol{D}$ is an $N_b \times (M_c + M_w)$ matrix obtained by the horizontal concatenation of $\boldsymbol{D}^C$ and $\boldsymbol{D}^W$.

The paper is organized as follows. First, we give the details for the construction of different wavelet prototypes and the concomitant wavelet dictionaries generated by those prototypes. Then, we provide

details and examples of the use of the MATLAB software for modeling of ECG signals within the proposed framework. The software has been made available on a dedicated website [9].

Wavelet dictionaries

In this section we produce all the pseudo-codes for the construction of wavelets dictionaries, which can be used to achieve the model of every segment in a signal partition. As already mentioned, each dictionary is obtained by taking the prototypes from a wavelet basis and translating them within a shorted step than that corresponding to the wavelet basis.

Throughout the paper we adopt the following notation. Boldface fonts are used to indicate Euclidean vectors and matrices. Standard mathematical fonts are used to indicate components, e.g., $\boldsymbol{d} \in \mathbb{R}^N$ is a vector of N-components $d(i) \in \mathbb{R}$, $i = 1, \ldots, N$ and $\boldsymbol{D} \in \mathbb{R}^{N\times M}$ is a matrix of elements $D(i, j)$, $i = 1 \ldots, N$, $j = 1, \ldots, M$. The symbol $L^2(\mathbb{R})$ denotes the space of square integrable functions.

Wavelets are usually constructed starting from a *multiresolution analysis*, which is a sequence $\{V_j\}_{j=j_0}^{\infty}$ of closed subspaces of the space $L^2(\mathbb{R})$ which are nested and their union is dense in $L^2(\mathbb{R})$, i.e.

$$V_j \subset V_{j+1} \quad \forall\, j \geq j_0, \quad \overline{\mathrm{U}_{j=j_0}^{\infty} V_j} = L^2(\mathbb{R}). \tag{2}$$

We assume that there exists a function $\phi \in L^2(\mathbb{R})$ such that for $j \geq j_0$ the functions

$$\phi_{j,k}(x) = 2^{j/2}\phi\left(2^j x - k\right), \quad k \in \mathbb{Z}, \tag{3}$$

form uniformly stable bases of the spaces V_j, i.e. the bases are Riesz bases with bounds independent of the level j, see e.g. [4]. The functions $\phi_{j,k}$ are called *scaling functions* and the function ϕ is called a *generator* of scaling functions. Next we present a method for the actual construction of the scaling functions.

Generation of scaling functions

We assume that ϕ has a compact support $[0, K]$ for some $K \in \mathbb{N}$. From the nestedness of the multiresolution spaces V_j, it follows that there exists a *scaling filter* $\boldsymbol{h} = (h(1), \ldots, h(K+1))$such that

$$\phi(x) = \sum_{k=1}^{K+1} h(k)\phi(2x + 1 - k) \quad \forall\, x \in \mathbb{R}. \tag{4}$$

Out of the several approaches outlined in [22] for solving the scaling Eq. (4) we adopt the one described below, which is equivalent to that appearing in [23].

If $\int_0^K \phi(x)dx = c \neq 0$ then, integrating (4), we obtain

$$c = \sum_{k=1}^{K+1} h(k)\frac{c}{2} \tag{5}$$

which implies that $\boldsymbol{h}$ has to be normalized such that

$$\sum_{k=1}^{K+1} h(k) = 2. \tag{6}$$

The scaling Eq. (4) enables computing values of the scaling function ϕ at points $k/2^u$ for $k = 0, \ldots, K2^u$, $u \in \mathbb{N}$. First we compute values of ϕ at integer points. Since $\operatorname{supp}\phi = [0, K]$, we have $\phi(k) = 0$ for $k \neq (0, K)$. Let us define a vector

$$\Phi = (\phi(0), \ldots, \phi(K-1))^T, \tag{7}$$

where the $(.)^T$ indicates the transpose operation. Substituting $x = 0, \ldots, K-1$, into (4), we obtain

$$\Phi(i) = \phi(i-1) = \sum_{k=1}^{K+1} h(k)\phi(2i - 1 - k) = \sum_{j=2i-1}^{2i-K-1} h(2i - j)\phi(j-1) = \sum_{j=2i-1}^{2i-K-1} h(2i - j)\Phi(j). \tag{8}$$

We set $h(k) = 0$ for $k < 1$ and $k > K + 1$ and define a matrix $\boldsymbol{A}$ by

$$A(i, j) = h(2i - j), \qquad i, j = 1, \ldots, K. \tag{9}$$

Then, (8) is equivalent to

$$\Phi = \boldsymbol{A}\Phi. \tag{10}$$

This means that Φ is an eigenvector corresponding to the eigenvalue 1 of the matrix $\boldsymbol{A}$. If the multiplicity of this eigenvalue is 1, then Φ is given uniquely up to a multiplication by a constant. Our aim is to compute a vector ***phi*** such that

$$\mathrm{phi}(m) = \phi\Big(\frac{m-1}{2^u}\Big), \quad m = 1, \ldots, K\,2^u + 1, \tag{11}$$

for a chosen level $u \in \mathbb{N}$. From (7) and (11) we have

$$\mathrm{phi}\big(2^u(l-1)+1\big) = \phi(l-1) = \Phi(l), \qquad l = 1, \ldots, K. \tag{12}$$

We compute values of ϕ at points $l/2$. Note that for l even we already know these values. Using (4) and (12) we obtain

$$\mathrm{phi}\big(l2^{u-1}+1\big) = \phi\left(\frac{l}{2}\right) = \sum_{k=1}^{K+1} h(k)\phi(l+1-k) = \sum_{k=1}^{K+1} h(k)\mathrm{phi}\big((l+1-k)2^u+1\big) \tag{13}$$

for $l = 1, 3, \ldots, 2N-1$. Similarly, we compute values of ϕ at points $l/4$, and thus we continue until we determine values at points $l/2^u$. More precisely, for $i = 1, \ldots, u$ we assume that we know values of ϕ at $l/2^{i-1}$, $l = 0, \ldots, K2^{i-1} + 1$, and we compute the values

$$\mathrm{phi}(m) = \phi(x), \quad m = x2^u + 1, \quad x = l/2^{i-1} + 1/2^i. \tag{14}$$

Using (4) we obtain

$$\mathrm{phi}(m) = \phi(x) = \sum_{k=1}^{K+1} h(k)\phi(2x+1-k) = \sum_{k=1}^{K+1} h(k)\mathrm{phi}\big((2x+1-k)2^u+1\big). \tag{15}$$

Remark 1. Some scaling functions such as spline scaling functions are known in an explicit form and their values can be evaluated directly. However, an advantage of our approach is that it is more general and can be used for a large class of wavelet families.

Construction of wavelet generators from scaling functions

Let W_j be complement spaces such that $V_{j+1} = V_j \oplus W_j$, where $\oplus$ denotes a direct sum. Wavelet functions $\psi_{j,k}$ are constructed in the form:

$$\psi_{j,k}(x) = 2^{j/2}\psi\big(2^j x - k\big), \quad k \in \mathbb{Z}, \tag{16}$$

to be a basis for W_j and such that

$$\mathcal{B} = \big\{\phi_{j_0,k}, k \in \mathbb{Z}\big\} \cup \big\{\psi_{j,k}, k \in \mathbb{Z}, j \geq j_0\big\} \tag{17}$$

called a *wavelet basis*, is a Riesz basis of the space $L^2(\mathbb{R})$.

Since $W_j \subset V_{j+1}$ there exists a vector $\boldsymbol{g} = (g(1), \ldots, g(M+1))$ such that

$$\psi(x) = \sum_{k=1}^{M+1} g(k)\phi(2x+1-k). \tag{18}$$

The vector $\boldsymbol{g}$ is called a *wavelet filter*. From (18) we have

$$\mathrm{supp}\,\psi = \left[0, \frac{M+K}{2}\right]. \tag{19}$$

In Algorithm 1 we compute a vector ***psi*** such that

$$\mathrm{psi}(m) = \psi\Big(\frac{m-1}{2^u}\Big), \quad m = 1, \ldots, (M+K)2^{u-1} + 1, \tag{20}$$

Algorithm 1 Procedure [***phi***,***psi***] = WaveletGen(***h***,***g***,u)

Input:
h scaling filter
g wavelet filter
u level (integer) that determines points $l/2^u$
Output:
phi vector of scaling function values (c.f. (11))
psi vector of wavelet values (c.f. (20))

$K = \text{length}(\boldsymbol{h}) - 1$ {support length of ϕ}
$\boldsymbol{h} = 2\,\boldsymbol{h}/\text{sum}(\boldsymbol{h})$ {normalization of $\boldsymbol{h}$ (c.f. (6))}
{Compute a matrix $\boldsymbol{A}$ using (9)}
$\boldsymbol{A} = \text{zeros}(K)$
for $i = 1 : K$ **do**
  for $j = 1 : K$ **do**
    if $1 \le 2i - j \le K + 1$ **then**
      $A(i, j) = h(2i - j)$
    end if
  end for
end for
{Compute eigenvalues and eigenvectors of the matrix $\boldsymbol{A}$}
[V,D]=eig($\boldsymbol{A}$)
{Find an index of a column corresponding to eigenvalue 1}
$k = 0$ {k is the multiplicity of eigenvalue 1}
for $i = 1 : K$ **do**
  if $|(D(i, i) - 1| < 10^{-7}$ **then**
    column=i; $k = k + 1$
  end if
end for
if $k \neq 1$ **then**
  error('Impossible to construct scaling function: eigenvalue 1 must have multiplicity 1')
else
  phi $= \text{zeros}(K2^u + 1, 1)$
  {Eigenvector $V(:, \text{column})$ represents values of ϕ at integer points}
  $\text{phi}(1 : 2^u : (K - 1)2^u + 1) = V(:, \text{column})$ {c.f. (12)}
  {Compute values of ϕ at points $l/2^u$}
  for $i = 1 : u$ **do**
    for $l = 1 : K2^{i-1}$ **do**
      $x = 2^{-i} + (l - 1)\,2^{-i+1}$; $m = x\,2^u + 1$ {c.f. (14)}
      for $k = 1 : K + 1$ **do**
        if $0 \le 2x - k + 1 \le K$ **then**
          $\text{phi}(m) = \text{phi}(m) + h(k)\text{phi}((2x - k + 1)\,2^u + 1)$ {c.f. (15)}
        end if
      end for
    end for
  end for
  $M = \text{length}(\boldsymbol{g}) - 1$
  {Compute ***psi*** containing values of ψ at points $l/2^u$}
  psi $= \text{zeros}((K + M)\,2^{u-1} + 1, 1)$
  for $k = 1 : M + 1$ **then**
    $i_1 = (k - 1)\,2^{u-1} + 1$; $i_2 = (k - 1)\,2^{u-1} + 1 + K\,2^{u-1}$
    $\text{psi}(i_1 : i_2) = \text{psi}(i_1 : i_2) + g(k)\,\text{phi}(1 : 2 : K\,2^u + 1)$ {c.f. (25)}
  end for
end if

in the following way. Due to (18) and (20), we have

$$\text{psi}(m) = \sum_{k=1}^{M+1} g(k)\phi\big((m-1)\,2^{1-u} + 1 - k\big) = \sum_{k=1}^{M+1} g(k)\text{phi}\big(2m - 1 + (1-k)\,2^u\big). \tag{21}$$

The sum in the last equation is computed as a cyclic sum. For

$$m = 1, \ldots, (M + K)2^{u-1} + 1 \tag{22}$$

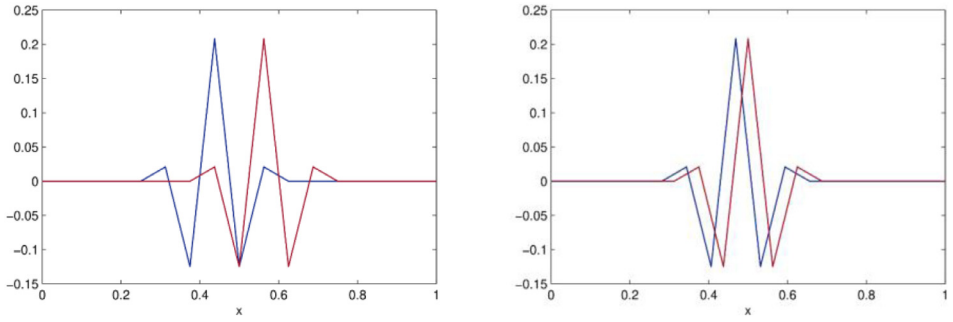

Fig. 1. Wavelet functions taken from a basis (left) and a dictionary (right) corresponding to a linear spline-wavelet prototype from [6].

we set psi$(m) = 0$ and for $k = 1, \ldots, M+1$ we do

$$\text{psi}(m) = \text{psi}(m) + g(k)\text{phi}\left(2m - 1 + (1-k)\,2^u\right), \tag{23}$$

if $1 \le 2m - 1 + (1-k)\,2^u \le K2^u + 1$. Using the substitution

$$2m - 1 + (1-k)\,2^u \;= 2i - 1, \tag{24}$$

for $i = 1, \ldots, K2^{u-1} + 1$, we obtain

$$\text{psi}\left(i + (k-1)2^{u-1}\right) = \text{psi}\left(i + (k-1)2^{u-1}\right) + g(k)\text{phi}(2i-1). \tag{25}$$

Algorithm 1 computes vectors ***phi*** and ***psi*** for given scaling and wavelet filters. The filters corresponding to the wavelet families supported by the software are given in Appendix A (Algorithm 8).

Construction of wavelet bases and dictionaries

Hereafter we drop all normalization factors and normalize all the vectors once they has been constructed. Note that in (17) we used a translation parameter $k \in \mathbb{Z}$ and since $\mathcal{B}$ is a Riesz basis the functions from $\mathcal{B}$ are linearly independent. Now, we choose a parameter b such that $b = 2^{-m}$ for some integer m. We define functions

$$\phi_{j_0,k,b}(x) = \phi\left(2^{j_0}x - bk\right), \quad k \in \mathbb{Z}, \tag{26}$$

and

$$\psi_{j,k,b}(x) = \psi\left(2^{j}x - bk\right), \quad k \in \mathbb{Z}, \quad j \ge j_0, \tag{27}$$

which form a redundant dictionary [1],[2],[21]. Obviously, $b = 1$ corresponds to a basis.

The left graph of Fig. 1 shows two consecutive wavelet functions taken from a linear spline bases [6]. The right graph of Fig. 1 corresponds to two consecutive wavelet functions taken from the dictionary spanning the same space which corresponds to $b = 1/4$.

Algorithm 2 constructs a discrete dictionary, i.e., a matrix $\boldsymbol{D}^W$ which contains values of functions from (26) and (27) at N_b equidistant points for some chosen levels determined by the vector $\boldsymbol{j}$. Since Algorithm 1 enables us to construct values at points of the form $l/2^u$, we evaluate functions (26) and (27) at the points

$$x = \frac{l}{2^r}, \quad l = 0, \ldots, N_b - 1, \quad r = \left\lceil \frac{log\ (N_b - 1)}{log\ (2)} \right\rceil, \tag{28}$$

where $\lceil y \rceil$ denotes the smallest integer number larger than y.

Algorithm 2 Procedure $[\boldsymbol{D}^W, \boldsymbol{ind}, \boldsymbol{col}]$ = WaveletDict(namef, N_b, $\boldsymbol{j}$, b)

Input:

namef	name of a wavelet family, for available choices see Appendix A
N_b	number of points
$\boldsymbol{j}$	vector of levels
b	translation factor $b = 2^{-r_b}$ for some integer r_b

Output:

$\boldsymbol{D}^W$	wavelet dictionary
ind	ind(1) is the number of scaling functions at level $j(1)$, and ind(k) for $k > 1$ is the number of wavelets at level $j(k-1)$ the number of wavelets at level $j(k-1)$
col	cell array such that col{n} = {j, k, type, function} if the n-th column of $\boldsymbol{D}^W$ corresponds to values of a scaling function $\phi(2^j x - bk)$ or a wavelet $\psi(2^j x - bk)$; type='inner'or 'boundary' characterizes type of a function; function='scaling' or 'wavelet' indicates whether the column corresponds to the values of a scaling function or a wavelet

{Compute scaling and wavelet filters using Algorithm 8 from Appendix A}
[***h***,***g***,correct_name] = Filters(namef)
{Test if a wavelet family namef is available}
if correct_name= 0 **then**
  $\boldsymbol{D}^W = [\,]$; ***ind*** $= [\,]$; ***col*** $= [\,]$; return
end if
K = length(***h***) − 1 {support length of ϕ}
$s = (K + \text{length}(\boldsymbol{g}) - 1)/2$ {support length of ψ }
r = $\lceil \log\,(N_b - 1)/\log\,(2) \rceil$ {level characterizing N_b (c.f. (28)}
{Remove levels from ***j*** that contain no inner function}
$j_{min} = \log\,(s\,2^r/(N_b - 1))/\log\,(2))$ {coarsest possible level}
$\boldsymbol{j} = \boldsymbol{j}(\boldsymbol{j} >= j_{min})$ {removing the levels smaller than j_{min}}
{Test of parameters}
d_j = length(***j***); $r_b = \lceil \log(1/b)/\log(2) \rceil$ parameter r_b from $b = 1/2^{r_b}$
if $d_j = 0$ **then**
  fprintf('no inner functions for these values of levels ***j***, increase ***j***')
  $\boldsymbol{D}^W = [\,]$; ***ind*** $= [\,]$; ***col*** $= [\,]$; return
else if $r <$ max $(\boldsymbol{j}) + r_b$ **then**
  fprintf('small number of points N_b for these values of ***j*** and b')
  $\boldsymbol{D}^W = [\,]$; ***ind*** $= [\,]$; ***col*** $= [\,]$; return
end if
{Compute scaling and wavelet generators using Algorithm 1}
[***phi***, ***psi***]=WaveletGen(***h***, ***g***, $r - j(1)$)
{Compute number of scaling functions at level $j(1)$}
ind = zeros($d_j + 1, 1$), $a = 1/b$;
ind(1) = $Ka - 1 + \lceil (N_b - 1)\,2^{j(1)-r}/b \rceil$ {c.f. (34)}
{Compute number of wavelets for level l}
for $l = 1 : d_j$ **do**
  ind($1 + l$) = $sa - 1 + \lceil (N_b - 1)\,2^{j(l)-r}/b \rceil$ {c.f. (35)}
end for
{Compute columns of $\boldsymbol{D}^W$ corresponding to scaling functions}
n_f = sum(***ind***); $\boldsymbol{D}^W$ = zeros(N_b, n_f); ***col*** = cell($n_f, 1$)
l_s = length(***phi***); $n_1 = Ka - 1$ {c.f. (32)}
$n_2 = \lceil (N_b - 1)\,2^{j(1)-r}/b \rceil - \lfloor ((N_b - 1)\,2^{j(1)-r} - K)/b \rfloor - 1$ {c.f. (33)}
{Compute columns corresponding to inner scaling functions (c.f. (38))}
for $i = n_1 + 1 :$ ind(1) $- n_2$ **do**
  $D^W\,(b(i - Ka)2^{r-j(1)} + 1 : b(i - Ka)\,2^{r-j(1)} + K\,2^{r-j(1)} + 1, i) =$ ***phi***
  col{i} = {$j(1)$, $i - Ka$, 'inner', 'scaling'}
end for
{Compute columns corresponding to boundary scaling functions (c.f. (38))}
for $i = 1 : n_1$ **do**
  $D^W\,(1 : l_s - b\,(n_1 - i + 1)\,2^{r-j(1)}, i)$ = phi($(n_1 - i + 1)\,b2^{r-j(1)} + 1 : l_s$)
  col{i} = {$j(1), -n_1 + i - 1$,' boundary', 'scaling'}
end for

(*continued on next page*)

Algorithm 2 (*continued*)

for $i = 1 : n_2$ **do**
  $p = \text{ind}(1) - n_2 + i$ {index of column}
  $D^W(b\,(p - Ka)2^{r-j(1)} + 1 : N_b, p) = \text{phi}(1 : N_b - b(p - Ka)2^{r-j(1)})$
  $\text{col}\{p\} = \{j(1), -n_1 + p - 1, \text{'boundary'}, \text{'scaling'}\}$
end for
{Compute columns of $\boldsymbol{D}^W$ corresponding to wavelets (c.f. (40))}
$n_p = \text{ind}(1)$ {number of functions on previous levels}
for $l = 1 : d_j$ **do**
  $n_1 = sa - 1$
  $k_1 = \lfloor((N_b - 1)2^{j(l)-r} - s)/b\rfloor$, $k_2 = \lceil((N_b - 1)2^{j(l)-r})/b) - 1\rceil$
  $n_2 = k_2 - k_1$; $n_f = n_1 + n_2 + k_1 + 1$; $l_w = \text{length}(\boldsymbol{psi})$
  for $i = n_1 + 1 : n_f - n_2$ **do**
    $D^W(b(i - sa)2^{r-j(l)} + 1 : b(i - sa)2^{r-j(l)} + s2^{r-j(l)} + 1, i + n_p) = \boldsymbol{psi}$
    $\text{col}\{i + n_p\} = \{j(l), i - sa, \text{'inner'}, \text{'wavelet'}\}$
  end for
  for $i = 1 : n_1$ **do**
    $D^W(1 : l_w - b(n_1 - i + 1)2^{r-j(l)}, i + n_p) = \text{psi}((n_1 - i + 1)b2^{r-j(l)} + 1 : l_w)$
    $\text{col}\{i + n_p\} = \{j(l), -n_1 + i - 1, \text{'boundary'}, \text{'wavelet'}\}$
  end for
  for $i = 1 : n_2$ **do**
    $p = n_f - n_2 + i$
    $D^W(b(p - sa)2^{r-j(l)} + 1 : N_b, p + n_p) = \text{psi}(1 : N_b - b(p - sa)2^{r-j(l)})$
    $\text{col}\{n_p + p\} = \{j(l), -n_1 + p - 1, \text{'boundary'}, \text{'wavelet'}\}$
  end for
  $\boldsymbol{psi} = \text{psi}(1 : 2 : \text{length}(\boldsymbol{psi}))$, $n_p = n_p + \text{ind}(l + 1)$
end for

For a chosen vector of levels $\boldsymbol{j}$, we define a vector of indices $\boldsymbol{ind}$ such that ind(1) is the number of scaling functions at level $j(1)$, and ind(l) is the number of wavelets at level $j(l-1)$ for $l = 1, \ldots, J$, where J is the length of $\boldsymbol{j}$. We have

$$\text{supp}\,\phi_{j(1),k,b} = \left[\frac{bk}{2^{j(1)}}, \frac{bk + K}{2^{j(1)}}\right], \quad \text{supp}\,\psi_{j,k,b} = \left[\frac{bk}{2^{j}}, \frac{bk + \frac{K+M}{2}}{2^{j}}\right]. \tag{29}$$

Comparing the supports of these functions and the interval

$$I = \left[0, \frac{N_b - 1}{2^r}\right] \tag{30}$$

which contains the points from (28), we find that the number of inner scaling functions, i.e., scaling functions with the whole support in I, is

$$n_i = \left\lfloor \frac{(N_b - 1)\,2^{j(1)-r} - K}{b} \right\rfloor + 1, \tag{31}$$

where the symbol $\lfloor y \rfloor$ denotes the largest integer number smaller than y. The number of left boundary scaling functions, i.e., functions that have only a part of the support in the interior of I and their support contains 0, is

$$n_1 = Ka - 1, \quad a = 1/b, \tag{32}$$

and similarly the number of right boundary scaling functions is

$$n_2 = \left\lceil \frac{(N_b - 1)\,2^{j(1)-r}}{b} \right\rceil - \left\lfloor \frac{(N_b - 1)\,2^{j(1)-r} - K}{b} \right\rfloor - 1. \tag{33}$$

Hence, we have

$$\text{ind}(1) = n_1 + n_i + n_2 = Ka - 1 - \left\lceil \frac{(N_b - 1)\,2^{j(1)-r}}{b} \right\rceil. \tag{34}$$

Algorithm 3 Procedure $[\boldsymbol{D}^W, \boldsymbol{ind}, \boldsymbol{col}]$= GenDict(name,pars)

Input:
name — name of a wavelet family, for available choices see Appendix A
pars — parameters in the form pars $= \{N_b, \boldsymbol{j}, b\}$
Description of the parameters:
N_b — number of points
$\boldsymbol{j}$ — vector of levels
b — translation factor $b = 2^{-r_b}$ for some integer r_b
Output:
$\boldsymbol{D}^W$ — wavelet dictionary
ind — ind(1) is the number of scaling functions at level $j(1)$, and ind(k) for $k > 1$ is the number of wavelets at level $j(k-1)$
col — cell array such that col{n} = {j, k, type, function}, if the n-th column of $\boldsymbol{D}^W$ corresponds to values of scaling function $\phi(2^j x - bk)$ or wavelet $\psi(2^j x - bk)$; type='inner' or 'boundary' characterizes type of a function; function='scaling' or 'wavelet' indicates whether the column corresponds to the values of a scaling function or a wavelet

{Define cell array of names of all available families}
families= {'CW2','CW3','CW4','CDF97','CDF97d','CDF53',
'Short4','Short3','Short2','Db3','Db4','Db5', 'Sym3','Sym4','Sym5','Coif26','Coif38'}
{Validate input parameters}
if nargin $\neq 2$ **then**
    error('Need 2 input arguments')
end if
if ~ischar(namef) **then**
    error('Name must be a string')
end if
N_b = pars{1}; $\boldsymbol{j}$ = pars{2}; b = pars{3}; $\boldsymbol{j} = \text{sort}(\boldsymbol{j})$
if $b \leq 0$ **then**
    error('I expect $b > 0$')
end if
$r = \log(1/b)/\log(2)$
if $|r - \text{round}(r)| > 10^{-10}$ **then**
    fprintf('Choose b such that $1/b = 2^r$ for some integer r')
    $\boldsymbol{D}^W = [\,]$; ***ind*** $= [\,]$; ***col*** $= [\,]$; return
else if ismember({namef},families) **then**
    {Generate dictionary using Algorithm 2}
    $[\boldsymbol{D}^W, \boldsymbol{ind}, \boldsymbol{col}]$ = WaveletDict(namef, N_b, $\boldsymbol{j}$, b)
    {Normalize columns of $\boldsymbol{D}^W$ using Algorithm 9 from Appendix A}
    $\boldsymbol{D}^W$ = NormDict($\boldsymbol{D}^W$,1)
else
    error('Unknown name of a wavelet family')
end if

Similarly, the number of wavelet functions on the level $j(l)$ is

$$\text{ind}(1+l) = sa - 1 + \left\lceil \frac{(N_b - 1)\, 2^{j(l)-r}}{b} \right\rceil. \tag{35}$$

The first ind(1) columns of $\boldsymbol{D}^W$ contain values of scaling functions (26), which restricted to I are not identically zero, at points given in (28), i.e.,

$$D^W(k,l) = \phi\left(2^{j(1)}\frac{k-1}{2^r} - b(l - Ka)\right) \tag{36}$$

for $k = 1, \ldots, N_b$, $l = 1, \ldots, \text{ind}(1)$. The above equation can be recast:

$$D^W(k,l) = \phi\left(\frac{(k-1) - b(l-Ka)2^{r-j(1)}}{2^{r-j(1)}}\right) = \text{phi}\big(k - b(l-Ka)2^{r-j(1)}\big), \tag{37}$$

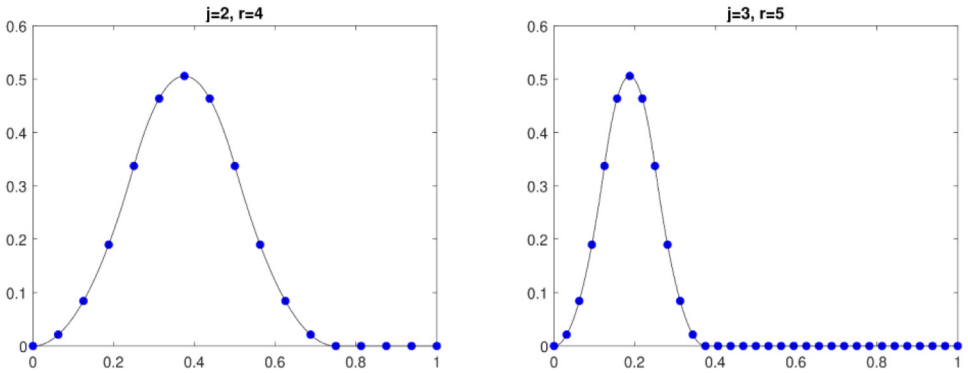

Fig. 2. Vectors of values $\varphi_{j,0,b}\ (l/2^r)$ for $j = 2$ and $r = 4$ (left) and for $j = 3$ and $r = 5$ (right).

where ***phi*** is defined by (11) for the level $u = r - j(1)$. Using the substitution $m = k - b(l - Ka)2^{r-j(1)}$, we obtain

$$D^W\big(m + b(l - Ka)2^{r-j(1)}, l\big) = \mathrm{phi}(m), \quad m = 1, \ldots, K2^{r-j(1)} + 1, \tag{38}$$

under the assumption that $1 \le m + b(l - Ka)2^{r-j(1)} \le N_b$.

The other columns of $\boldsymbol{D}^W$ contain values of wavelet functions (27) for levels $j = j(1), \ldots, j(J)$ at points (28), i.e.,

$$D^W\big(k, n_p + l\big) = \psi\left(2^j \frac{k-1}{2^r} - b(l - sa)\right), \quad s = \frac{M+K}{2}, \tag{39}$$

for $k = 1, \ldots, N_b$, $l = 1, \ldots, \mathrm{ind}(j+1)$, and $n_p = \sum_{p=j(1)}^{j} \mathrm{ind}(p + 1 - j(1))$. Similarly as above we obtain

$$D^W\big(m + b(l - sa)\, 2^{r-j(l)}, l + n_p\big) = \mathrm{psi}(m), \tag{40}$$

where $1 \le m + b(l - sa)2^{r-j(1)} \le N_b$ and ***psi*** is defined by (20) for the level $u = r - j(1)$.

The following procedure WaveletDict computes a wavelet dictionary.

The main procedure GenDict validates input parameters, generates dictionaries $\boldsymbol{D}^W$ and normalizes their columns.

Remark 2. It is worth remarking that the range of scales, say $\boldsymbol{j} = (j_0, \ldots, J)$ depends on length of the signal partition. For a signal segment of length $N_b = 2^r + 1$ a dictionary contains values of scaling functions and wavelets at points $l/2^r$ for some integer r. For a signal segment of length $2N_b - 1 = 2^{r+1} + 1$ a dictionary contains values of functions at points $l/2^{r+1}$. Thus, we have

$$\phi_{j,k,b}\left(\frac{l}{2^r}\right) = \phi\left(2^j \frac{l}{2^r} - kb\right) = \phi\left(2^{j+1} \frac{l}{2^{r+1}} - kb\right) = \phi_{j+1,k,b}\left(\frac{l}{2^{r+1}}\right) \tag{41}$$

and

$$\psi_{j,k,b}\left(\frac{l}{2^r}\right) = \psi\left(2^j \frac{l}{2^r} - kb\right) = \psi\left(2^{j+1} \frac{l}{2^{r+1}} - kb\right) = \psi_{j+1,k,b}\left(\frac{l}{2^{r+1}}\right). \tag{42}$$

Therefore, nonzero elements of vectors on the level j in a dictionary for $\mathbb{R}^{N_b}$ correspond to nonzero elements of vectors on the level $j+1$ in a dictionary for $\mathbb{R}^{2N_b-1}$. This situation is illustrated in Fig. 2, where vectors of values $\phi_{j,0,b}(l/2^r)$ are displayed for $j = 2$ and $r = 4$ and for $j = 3$ and $r = 5$. Note that the nonzero elements in these vectors are the same. Therefore, if for the signal segment of length N_b the vector $\boldsymbol{j} = (j_0, \ldots, J)$ is used, then we recommend to use the vector $\boldsymbol{j} = (j_0, \ldots, J+1)$ for the signal segment of length $2N_b - 1$, and similarly to use levels $\boldsymbol{j} = (j_0, \ldots, J+m)$ for the signal segment of length $2^m N_b - 1$.

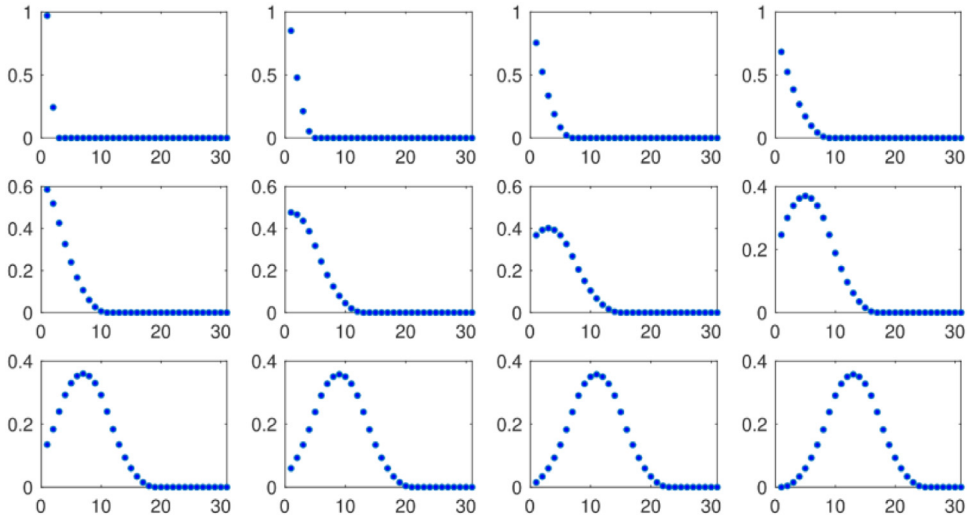

Fig. 3. Plots of 12 vectors from the dictionary $\mathbf{D}^W$ from Example 1 corresponding to scaling functions on the level 2.

Example 1. To build dictionaries for the wavelet family 'Short3' at levels 2 and 3, for translation parameter $b = 1/4$, and the number of points $N_b = 33$, use the procedure TestDict below.

Algorithm 4 Procedure TestDict

namef='Short3'; $N_b = 33$; $\boldsymbol{j} = 2:3$; $b = 1/4$
$[\boldsymbol{D}^W, \boldsymbol{ind}, \boldsymbol{col}]$=GenDict(namef,$\{N_b, \boldsymbol{j}, b\}$)

The output is the matrix $\boldsymbol{D}^W$ of size 33×97 and the vector $\boldsymbol{ind} = [27, 27, 43]$. This means that there are 27 scaling functions at level 2, 27 wavelets at level 2, and 43 wavelets at level 3. The cell array $\boldsymbol{col}$ characterizes functions corresponding to columns of $\boldsymbol{D}^W$. For example

$$\text{col}\{30\} = \left\{2, -9, 'boundary', 'wavelet'\right\} \tag{43}$$

which means that 30th column of the matrix $\boldsymbol{D}^W$ contains values of a wavelet function $\psi(2^2x - b(-9))$. This wavelet is a boundary wavelet, i.e., only a part of its support lies in the interval I defined by (30). Some of the vectors from this dictionary corresponding to values of scaling functions are displayed in Fig. 3 and some ofthe vectors corresponding to values of wavelets are displayed in Fig. 4.

Construction of dictionaries for ECG modeling

As mentioned above, because ECG signals are usually superimposed to a baseline or smooth background, the full dictionary $\boldsymbol{D}$ we use for ECG modeling is built as follows

$$\boldsymbol{D} = \left[\boldsymbol{D}^C \boldsymbol{D}^W\right], \tag{44}$$

where $\boldsymbol{D}^W$ is the output of Algorithm 5 and $\boldsymbol{D}^C$ is a matrix containing a few low frequency components from a discrete cosine basis. Before normalization $\boldsymbol{D}^C$ is given as

$$D^C(k, n) = \cos\left(\pi(2k-1)(n-1)/(2N_b)\right),\ k = 1, \ldots, N_b,\ n = 1, \ldots, M_c, \tag{45}$$

where M_c is a small number in comparison to N_b. For the numerical examples of the next section we consider $M_c = 10$. Algorithm 5 computes $\boldsymbol{D}^C$.

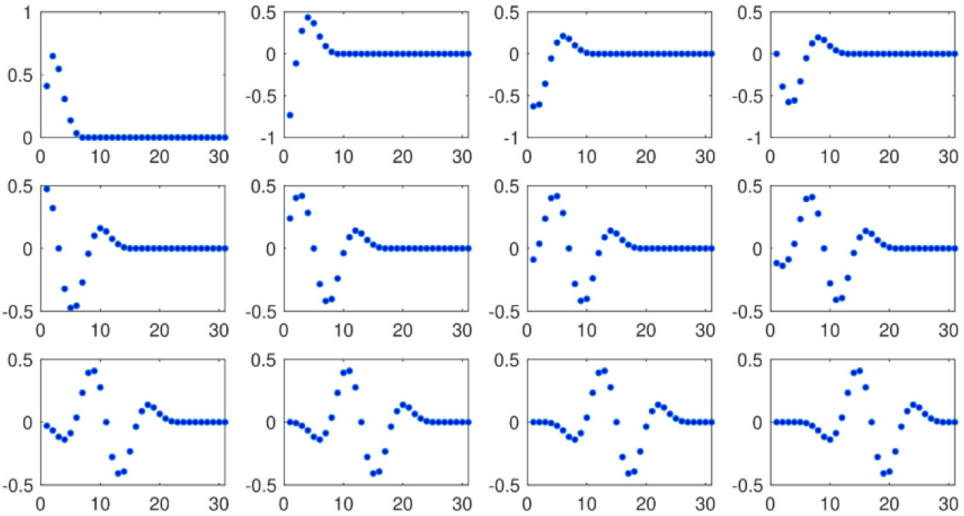

Fig. 4. Plots of 12 vectors from the dictionary $\mathbf{D}^W$ from Example 1 corresponding to wavelets on the level 2.

Algorithm 5 Procedure $\boldsymbol{D}^C = \mathrm{DCos}(N_b, M_c)$

Input:
N_b the size of the Euclidean space the vectors should belong to
M_c number of frequencies to use starting from 0
Output:
$\boldsymbol{D}^C$ matrix whos columns are discrete cosine vectors

$n = 1 : M_c;\ k = 1 : N_b$
$\boldsymbol{D}^C = \cos\ (\pi\,(2k-1)^T\,(n-1)/(2N_b))$
$\boldsymbol{D}^C = \mathrm{NormDict}(\boldsymbol{D}^C,1)$

Construction of the model

In this section we present the procedures for constructing the ECG signal model (c.f. Algorithm 6) and for calculating the assessment metrics. The quality of the signal approximation is assessed with respect to the PRD defined as follows

$$\mathrm{PRD} = \frac{\|\boldsymbol{f} - \boldsymbol{f}^r\|}{\|\boldsymbol{f}\|} \times\ 100\%, \tag{46}$$

where $\boldsymbol{f}$ is the original signal and $\boldsymbol{f}^{\mathrm{r}}$ is the signal reconstructed by concatenation of the approximated segments $\boldsymbol{f}^{\mathrm{a}}\{q\}$, $q = 1, \ldots, Q$.

The local PRD with respect to every segment in the signal partition is indicated as $\mathrm{prd}(q)$, $q = 1, \ldots, Q$ and calculated as

$$\mathrm{prd}(q) = \frac{\|\boldsymbol{f}\{q\} - \boldsymbol{f}^a\{q\}\|}{\|\boldsymbol{f}\{q\}\|} \times\ 100\%, \quad q = 1, \ldots, Q. \tag{47}$$

For the signal approximation the OOMP method is stopped through a fixed value tol so as to achieve the same value of prd for all the segments in the records. Assuming that the target prd before quantization is prd_0 we set $\mathrm{tol} = \mathrm{prd}_0\|\boldsymbol{f}_q\|/100$.

The goal of the signal model is to approximate each segment in the signal partition using as few atoms as possible. Thus, for a fixed value of PRD, the sparsity of the signal representation is assessed by the sparsity ratio (SR)

$$\mathrm{SR} = \frac{N}{K}, \tag{48}$$

Algorithm 6 Procedure $[\boldsymbol{f}^{\mathrm{r}}, \ell, \boldsymbol{c}, \mathrm{prd}, \mathrm{sr}, \mathrm{PRD}, \mathrm{SR}] = \mathrm{SignalModel}(\boldsymbol{f}, N_b, \mathrm{prd}_0, \mathrm{namef}, \mathrm{pars}, M_c)$

Input:

$\boldsymbol{f}$	signal
N_b	number of points in each segment of the partition
prd_0	parameter to control the approximation error
namef	name of a wavelet family
pars	parameters as described in Algorithm 3
M_c	number M_c of components in the cosine subdictionary

Output:

$\boldsymbol{f}^{\mathrm{r}}$	approximated signal
ℓ	cell with the indices of the atoms in the atomic decomposition of each element in the partition
$\boldsymbol{c}$	cell with the coefficients in the atomic decomposition of each element in the partition
prd	vector $\mathrm{prd} \in \mathbb{R}^Q$ (cf. (47))
sr	vector $\mathrm{sr} \in \mathbb{R}^Q$ (cf. (49))
PRD	global PRD
SR	global SR

{Create the signal partition using Algorithm 10}
$[\boldsymbol{f}^{\mathrm{c}}, Q, \boldsymbol{f}] = \mathrm{Partition}(\boldsymbol{f}, N_b)$
{Construct the wavelet dictionary $\boldsymbol{D}^W$ using Algorithm 3 given in Appendix B}
$[\boldsymbol{D}^W, \mathrm{ind}] = \mathrm{GenDict}(\mathrm{namef}, \mathrm{pars})$
{Construct the component $\boldsymbol{D}^C$ using Algorithm 5 given in Appendix B}
$[\boldsymbol{D}^C] = \mathrm{DCos}(N_b, M_c)$
{Merge $\boldsymbol{D}^C$ and $\boldsymbol{D}^W$ to create dictionary $\boldsymbol{D}$}
$\boldsymbol{D} = [\boldsymbol{D}^C\ \boldsymbol{D}^W]$
Set $\boldsymbol{f}^{\mathrm{r}} = [\ ]$, $K = 0$ and $N = \mathrm{length}(\boldsymbol{f})$.
for q=1:Q **do**
  $\mathrm{tol} = \mathrm{prd}_0 \|\boldsymbol{f}^{\mathrm{c}}\{q\}\|/100$
  {Call the OOMP function to construct the model (c.f. (1))}
  $[\boldsymbol{f}^{\mathrm{a}}\{q\}, \ell\{q\}, \boldsymbol{c}\{q\}] = \mathrm{OOMP}(\boldsymbol{f}^{\mathrm{c}}\{q\}, \boldsymbol{D}, \mathrm{tol}, 1)$
  {Calculate local sr and prd (c.f. (49) and (47))}
  $\mathrm{prd}(q) = \frac{\|\boldsymbol{f}^{\mathrm{c}}\{q\} - \boldsymbol{f}^{\mathrm{a}}\{q\}\|}{\|\boldsymbol{f}^{\mathrm{c}}\{q\}\|} \times 100$
  $k(q) = \mathrm{length}(\boldsymbol{c}\{q\}))$
  $\mathrm{sr}(q) = N_b/k(q)$
  $K = K + k(q)$
  $\boldsymbol{f}^{\mathrm{r}} = [\boldsymbol{f}^{\mathrm{r}}\, \boldsymbol{f}^{\mathrm{a}}\{q\}]$
end for
{Calculate global SR and PRD (c.f. (48) and (46))}
$\mathrm{SR} = N/K$
$\mathrm{PRD} = \frac{\|\boldsymbol{f} - \boldsymbol{f}^{\mathrm{r}}\|}{\|\boldsymbol{f}\|} \times 100$

where N is the total length of the signal and $K = \sum_{q=1}^{Q} k(q)$, with $k(q)$ the number of atoms in the atomic decomposition (1) of each segment of length N_b. The corresponding quantity evaluated for every cell in the partition is the local sparsity ratio

$$\mathrm{sr}(q) = \frac{N_b}{k(q)}, \quad q = 1, \ldots, Q. \tag{49}$$

This local quantity is relevant to the detection of non-stationary noise, significant distortion in ECG patterns, or changes of morphology in the heart beats.

Given an ECG signal $\boldsymbol{f}$ the procedure described in Algorithm 6 constructs the signal approximation, $\boldsymbol{f}^{\mathrm{r}}$, using the dictionaries introduced in the previous section.

Numerical examples

We illustrate now the use of the software to approximate records 117, 202, and 231 in the MIT-BIH Arrhythmia database. Each record consists of 650,000 samples and is partitioned for the approximation in segments of $N_b = 500$ points each. Let us remark that, while the size N_b should be adapted to the convenience of the application at hand, for the only purpose of reducing dimensionality, this size is not crucial. Table 1 gives the values of the SR (c.f. (48)) achieved

Table 1
SRs achieved using dictionaries with $\mathbf{D}^W$ component as indicated in the first column of the table. SR_B are values of SR obtained if $\mathbf{D}^W$ is a basis and SR_D if $\mathbf{D}^W$ is a dictionary.

Rec.	117		202		231	
$\mathbf{D}W$	SR_B	SR_D	SR_B	SR_D	SR_B	SR_D
CW2	17.5	26.5	17.3	24.5	15.7	23.0
CW3	17.4	28.1	15.9	24.9	15.6	24.0
CW4	15.7	24.8	14.3	22.5	18.4	21.9
CDF97	21.5	30.3	21.4	28.4	19.5	27.5
CDF97d	17.2	23.5	17.3	22.5	15.8	21.9
CDF53	22.4	29.6	23.6	27.0	20.2	27.0
Db3	18.5	23.7	18.1	22.7	16.6	22.7
Db4	19.0	25.7	19.1	24.7	17.7	24.1
Db5	20.4	26.1	18.7	24.2	17.8	24.1
Sym3	18.4	23.8	18.1	22.7	16.6	22.7
Sym4	19.7	27.5	19.5	25.8	17.7	25.1
Sym5	20.5	28.3	20.6	28.5	18.4	25.4
Short2	8.2	27.9	8.7	26.3	8.1	24.7
Short3	19.6	31.8	18.3	27.6	17.8	27.3
Short4	9.5	29.1	10.1	27.6	9.1	26.6
Coif26	17.7	23.0	17.7	21.8	16.3	24.7
Coif38	19.5	28.5	19.7	26.5	17.8	26.1

Algorithm 7

Procedure Run_ECG_Approx

{Read the signal $\boldsymbol{f}$}
file='Record_231_11bits.dat'
fid=fopen(file,'r')
$\boldsymbol{f}$=fread(fid,'ubit11')
fclose(fid)
{Set the required PRD for the approximation}
$\text{prd}_0 = 0.53$
{Set the length for each segment in the signal partition}
$N_b = 500$
{Set the parameters for the wavelet dictionary}
namef ='CDF97'; $b = 0.25$; $\boldsymbol{j} = 3:7$; pars = $\{N_b, \boldsymbol{j}, b\}$
{Set the number of cosine components}
$M_c = 10$
{Construct the signal module}
$[\boldsymbol{f}^{\text{r}}, \ell, \boldsymbol{c}, \text{prd}, \text{sr}, \text{PRD}, \text{SR}] = \text{SignalModel}(\boldsymbol{f}, N_b, \text{prd}_0, \text{namef}, \text{pars}, M_c)$
{Plote the first 2000 sample points in the signal, the approximation and the error}

using wavelet bases, denoted as SR_B, and wavelet dictionaries denoted as SR_D. The wavelet families are indicated in the first column of Table 1. The wavelet dictionary is constructed with scales $\boldsymbol{j} = (3, \ldots, 7)$ and translation parameter $b = 1/4$, whilst the wavelet basis entails to add one more scale and a translation parameter $b = 1$. In all the cases the approximation is realized to obtain PRD = 0.51%.

Table 1 is produced by running the script 'Run_ECG_Appox' and changing the variable 'namef' to the corresponding family option.

As observed in the Table 1, the gain in dimensionality reduction (larger value of SR) is significant when consider a wavelet dictionary, instead of a wavelet basis, as for constructing the component $\boldsymbol{D}^W$ of the full dictionary. This results were demonstrated in [17] on the whole MIT-BIH Arrhythmia database, which motivated this Communication to provide the details and algorithm for the actual construction of wavelet dictionaries from different options for the wavelet prototypes.

The top left graph in Fig. 5 illustrates the first 2000 points in the record 231 and the approximation for PRD = 0.51%. The top right graph represents the values of local sparsity $1/\text{sr}(q)$, $q = 1, \ldots, 1300$ for the same record. It is noticed that these values can be classified into two well defined bands. The

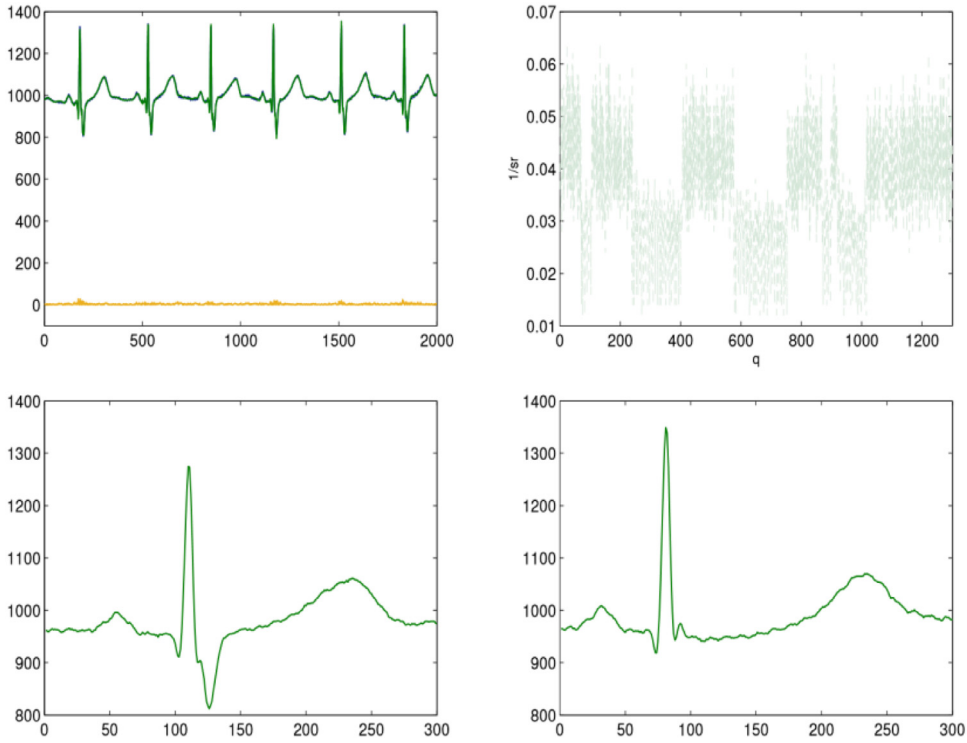

Fig. 5. The waveforms in the top left graph are the raw data and the approximations corresponding to 2000 points in the records 231 (the bottom line in the same graph is the point-wise error). The top right graph plots the values 1/sr for record 231. The bottom left graph is a typical heart beat in a segment for which the values of 1/sr(q) belongs to the upper band. In the bottom right graph the heart beat corresponds to a frame in the lower band.

bottom left graph in Fig. 5 shows a typical heart beat in a frame corresponding to a value 1/sr in the upper band, and the bottom right graph to a value in the lower band. The morphologic difference between the two heart beats is noticeable at a glance. However, the 1/sr values provide only a 'crude' feature about changes in an ECG record. In order to perform further analysis for classification using the features produced by the model machine learning techniques would be required [16].

Conclusions

Details on the construction of wavelets dictionaries for modeling ECG signals have been provided. The use of the software, which has been made publicly available on a dedicated website [9], was illustrated to reduce the dimensionality of three records from the MIT-BIH Arrhythmia database. The conclusions coincide with those that were drawn in the previous publication [17] using the whole database. However, regardless of the particular application, the purpose of this paper was to provide a complete description of the construction of the wavelet dictionaries, which had not been addressed in [17]. We believe the proposed dictionaries should be of assistance to general applications which relay on dimensionality reduction at low level distortion as a first step of further ECG signal processing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by grant No. PURE-2020-4003 funded by the Technical University of Liberec.

Appendix A

In this appendix, we present auxiliary procedures used in algorithms. In the algorithms, 'namef' denotes a name of a wavelet family, available choices are:

namef =	'CW2'	Chui-Wang linear spline wavelets [6]
	'CW3'	Chui-Wang quadratic spline wavelets [6]
	'CW4'	Chui-Wang cubic spline wavelets [6]
	'CDF97'	primal CDF97 wavelets [3]
	'CDF97d'	dual CDF97 wavelets [3]
	'CDF53'	primal CDF53 wavelets [3]
	'Short4'	cubic spline wavelet with short support and 4 vanishing moments [5],[12]
	'Short3'	quadratic spline wavelet with short support and 3 vanishing moments [5],[12]
	'Short2'	linear spline wavelet with short support and 2 vanishing moments [5],[12]
	'Db3'	Daubechies wavelet with 3 vanishing moments [10]
	'Db4'	Daubechies wavelet with 4 vanishing moments [10]
	'Db5'	Daubechies wavelet with 5 vanishing moments [10]
	'Sym3'	symlet with 3 vanishing moments [11]
	'Sym4'	symlet with 4 vanishing moments [11]
	'Sym5'	symlet with 5 vanishing moments [11]
	'Coif26'	coiflet with 2 vanishing moments and the support length 6 that is most regular [11]
	'Coif38'	coiflet with 3 vanishing moments and the support length 8 that is most symmetrical [11]

A wavelet basis is determined by its scaling and wavelet filters. Algorithm 8 assigns these filters for a chosen wavelet family, the values of filters are computed by methods from [3],[5],[6],[7],[10]–[12].

Now, we introduce a simple procedure NormDict for normalization of dictionaries. More precisely, this procedure normalizes the columns of dictionary $\boldsymbol{D}$ to have the Euclidean norm equaled to $1/\sqrt{\delta}$.

Algorithm 8 Procedure [$\boldsymbol{h}$,$\boldsymbol{g}$,correct_name] = Filters(namef)

Input:
namef name of a wavelet family
Output:
$\boldsymbol{h}$ scaling filter for a wavelet family specified by 'namef'
$\boldsymbol{g}$ wavelet filter for a wavelet family specified by 'namef'
correct_name returns 1 if 'namef' is a name of an available wavelet family, otherwise returns 0

correct_name=1
switch (namef)
case 'CW2':
$\boldsymbol{h} = [1/2, 1, 1/2]$; $\boldsymbol{g} = [1, -6, 10, -6, 1]/12$
case 'CW3':
$\boldsymbol{h} = [1/4, 3/4, 3/4, 1/4]$; $\boldsymbol{g} = [1, -29, 147, -303, 303, -147, 29, -1]/480$
case 'CW4':
$\boldsymbol{h} = [1/8, 1/2, 3/4, 1/2, 1/8]$
$\boldsymbol{g} = [1, -124, 1677, -7904, 18482, -24264, 18482, -7904, 1677, -124, 1]/2520$
case 'CDF97':
$\boldsymbol{h} = [-0.045635881557, -0.028771763114, 0.295635881557, 0.557543526229, 0.295635881557, -0.028771763114, -0.045635881557]$
$\boldsymbol{g} = [0.026748757411, 0.016864118443, -0.078223266529, -0.266864118443, 0.602949018236, -0.266864118443, -0.078223266529, 0.016864118443, 0.026748757411]$
case 'CDF97d':
$\boldsymbol{h} = [0.026748757411, -0.016864118443, -0.078223266529, 0.266864118443, 0.602949018236, 0.266864118443, -0.078223266529, -0.016864118443, 0.026748757411000]$
$\boldsymbol{g} = [0.045635881557, -0.028771763114, -0.295635881557, 0.557543526229, -0.295635881557, -0.028771763114, 0.045635881557]$
case 'CDF53':
$\boldsymbol{h} = [1/2, 1, 1/2]$; $\boldsymbol{g} = [-1/8, -1/4, 3/4, -1/4, -1/8]$
case 'Short4':
$\boldsymbol{h} = [1/8, 1/2, 3/4, 1/2, 1/8]$; $\boldsymbol{g} = [1/8, -1/2, 3/4, -1/2, 1/8]$
case 'Short3':
$\boldsymbol{h} = [1/4, 3/4, 3/4, 1/4]$; $\boldsymbol{g} = [-1/4, 3/4, -3/4, 1/4]$
case 'Short2':
$\boldsymbol{h} = [1/2, 1, 1/2]$; $\boldsymbol{g} = [-1/2, 1, -1/2]$
case 'Db3':
$\boldsymbol{h} = [0.035226291882101, -0.085441273882241, -0.135011020010391, 0.459877502119331, 0.806891509313339, 0.332670552950957]$
$\boldsymbol{g} = [-0.332670552950957, 0.806891509313339, -0.459877502119331, -0.135011020010391, 0.085441273882241, 0.035226291882101]$

(*continued on next page*)

Algorithm 8 (*continued*)

case 'Db4':
$\boldsymbol{h} = [0.162901714025620, 0.505472857545650, 0.446100069123190, -0.019787513117910, -0.132253583684370, 0.021808150237390, 0.023251800535560, -0.007493494665130]$
$\boldsymbol{g} = -\text{fliplr}([0.162901714025620, -0.505472857545650, 0.446100069123190, 0.019787513117910, -0.132253583684370, -0.021808150237390, 0.023251800535560, 0.007493494665130])$
case 'Db5':
$\boldsymbol{h} = [0.003335725285002, -0.012580751999016, -0.006241490213012, 0.077571493840065, -0.032244869585030, -0.242294887066190, 0.138428145901103, 0.724308528438574, 0.603829269797473, 0.160102397974125]$
$\boldsymbol{g} = [-0.160102397974125, 0.603829269797473, -0.724308528438574, 0.138428145901103, 0.242294887066190, -0.032244869585030, -0.077571493840065, -0.006241490213012, 0.012580751999016, 0.003335725285002]$
case 'Sym3':
$\boldsymbol{h} = [0.035226291882101, -0.085441273882241, -0.135011020010391, 0.459877502119331, 0.806891509313339, 0.332670552950957]$
$\boldsymbol{g} = [-0.332670552950957, 0.806891509313339, -0.459877502119331, -0.135011020010391, 0.085441273882241, 0.035226291882101]$
case 'Sym4':
$\boldsymbol{h} = [0.022785172948000, -0.008912350720850, -0.070158812089500, 0.210617267102000, 0.568329121705000, 0.351869534328000, -0.020955482562550, -0.053574450709000]$
$\boldsymbol{g} = \text{fliplr}([0.022785172948000, 0.008912350720850, -0.070158812089500, -0.210617267102000, 0.568329121705000, -0.351869534328000, -0.020955482562550, 0.053574450709000])$
case 'Sym5':
$\boldsymbol{h} = [0.027333068345078, 0.029519490925775, -0.039134249302383, 0.199397533977394, 0.723407690402421, 0.633978963458212, 0.016602105764522, -0.175328089908450, -0.021101834024759, 0.019538882735287]$
$\boldsymbol{g} = [-0.019538882735287, -0.021101834024759, 0.175328089908450, 0.016602105764522, -0.633978963458212, 0.723407690402421, -0.199397533977394, -0.039134249302383, -0.029519490925775, 0.027333068345078]$
case 'Coif26':
$\boldsymbol{h} = [9 - \sqrt{(15)}, 13 + \sqrt{(15)}, 6 + 2\sqrt{(15)}, 6 - 2\sqrt{(15)}, 1 - \sqrt{(15)}, -3 + \sqrt{(15)}]/32$
$\boldsymbol{g} = -\text{fliplr}([9 - \sqrt{(15)}, -13 - \sqrt{(15)}, 6 + 2\sqrt{(15)}, -6 + 2\sqrt{(15)}, 1 - \sqrt{(15)}, 3 - \sqrt{(15)}]/32)$
case 'Coif38':
$\boldsymbol{h} = [-1/32 - \sqrt{(7)}/128, -3/128, 9/32 + 3\sqrt{(7)}/128, 73/128, 9/32 - 3\sqrt{(7)}/128, -9/128, -1/32 + \sqrt{(7)}/128, 3/128]$ $\boldsymbol{g} = -\text{fliplr}([-1/32 - \sqrt{(7)}/128, 3/128, 9/32 + 3\sqrt{(7)}/128, -73/128, 9/32 - 3\sqrt{(7)}/128, 9/128, -1/32 + \sqrt{(7)}/128, -3/128])$
otherwise
disp('wrong name of a wavelet family')
correct_name = 0
end switch

Algorithm 9 Procedure $\boldsymbol{D}$ = NormDict($\boldsymbol{D}$, δ)

Input:
$\boldsymbol{D}$ wavelet dictionary
δ parameter such that prescribed norm size is $1/\sqrt{\delta}$
Output:
$\boldsymbol{D}$ normalized wavelet dictionary such that the Euclidean norm of each column is $1/\sqrt{\delta}$

tol=10^{-5}
if nargin=1 **then**
$\delta = 1$
end if
N=size($\boldsymbol{D}$,2); $i = 0$
while $i < N$ **do**
$i = i + 1$; nor $= \sqrt{\delta}\,\|D(:, i)\|$
if nor > tol **then**
$D(:, i) = D(:, i)/\text{nor}$
else
$D(:, i) = [\,]; N = N - 1$
end if
end while

Appendix B

In this appendix, we present auxiliary procedures used in algorithms for the model construction. The next procedure Partition creates a partition of the signal $\boldsymbol{f}$ into Q segments of the prescribed length N_b.

The procedure for signal approximation using OOMP method is presented below.

Algorithm 10 Procedure [$\boldsymbol{f}^{\mathrm{c}}$, Q, $\boldsymbol{f}$]=Partition($\boldsymbol{f}$, N_b)

Input:
$\boldsymbol{f}$ signal
N_b length of each segment in the partition
Output:
$\boldsymbol{f}^{\mathrm{c}}$ cells $\boldsymbol{f}^{\mathrm{c}}\{q\}$, $q = 1, \ldots, Q$ with the signal partition
Q number of cells in the partition
$\boldsymbol{f}$ resized signal to be of length QN_b

$N = \text{length}(\boldsymbol{f})$; $Q = \lfloor \frac{N}{N_b} \rfloor$; $t_o = 1$
$\boldsymbol{f} \leftarrow f(1 : QN_b)$
for $q = 1 : Q$ **do**
$t = t_o : t_o + N_b - 1$; $t_0 = t_0 + N_b$
$\boldsymbol{f}^{\mathrm{c}}\{q\} = f(t)$
end for

Algorithm 11 Procedure [$\boldsymbol{f}^{\mathrm{a}}$, ℓ, $\boldsymbol{c}$]= OOMP($\boldsymbol{f}$, $\boldsymbol{D}$, tol, $\mathfrak{t}_1$)

Input:
$\boldsymbol{f}$ signal to be approximated by an atomic decomposition
$\boldsymbol{D}$ wavelet dictionary
tol parameter to control the approximation error
$\mathfrak{t}_1$ index of the atom for initializing the OOMP algorithm
Output:
$\boldsymbol{f}^{\mathrm{a}}$ approximation of the signal $\boldsymbol{f}$ (c.f. (1))
ℓ vector whose components are the indices of the selected columns from the input dictionary
$\boldsymbol{c}$ coefficients $\boldsymbol{c} \in \mathbb{R}^{N_b}$ of the atomic decomposition (c.f. (1))

{The method implemented in this function is fully described in the main paper [17].}

References

[1] M. Andrle, L. Rebollo-Neira, Cardinal B-spline dictionaries on a compact interval, Appl. Comput. Harmon. Anal. 18 (2005) 336–346, doi:10.1016/j.acha.2005.01.001.
[2] M. Andrle, L. Rebollo-Neira, From cardinal spline wavelet bases to highly coherent dictionaries, J. Phys. A 41 (172001) (2008) 172001 article No., doi:10.1088/1751-8113/41/17/172001.
[3] A. Cohen, I. Daubechies, J.C. Feauveau, Biorthogonal bases of compactly supported wavelets, Commun. Pure Appl. Math. 45 (1992) 485–560, doi:10.1002/cpa.3160450502.
[4] A. Cohen, Numerical Analysis of Wavelet methods, Studies in Mathematics and its Applications 32, Elsevier, Amsterdam, 2003.
[5] D. Chen, Spline wavelets of small support, SIAM J. Math. Anal. 26 (1995) 500–517, doi:10.1137/S0036141093245264.
[6] C. Chui, J. Wang, On compactly supported spline wavelets and a duality principle, Trans. Am. Math. Soc. 330 (1992) 903–915, doi:10.2307/2153941.
[7] D. Černá, V. Finěk, K. Najzar, On the exact values of coefficients of coiflets, Cent. Eur. J. Math. 6 (2008) 159–169, doi:10.2478/s11533-008-0011-2.
[8] https://physionet.org/physiobank/database/mitdb/ (Last access February 2021).
[9] http://www.nonlinear-approx.info/examples/node013.html (Last access February 2021).
[10] I. Daubechies, Orthonormal bases of compactly supported wavelets, Commun. Pure Appl. Math. 41 (1988) 909–996, doi:10.1002/cpa.3160410705.
[11] I. Daubechies, Orthonormal bases of compactly supported wavelets II, variations on a theme, SIAM J. Math. Anal. 24 (1993) 499–519, doi:10.1137/0524031.
[12] B. Han, Z. Shen, Wavelets with short support, SIAM J. Math. Anal. 38 (2006) 530–556, doi:10.1137/S0036141003438374.
[13] S.J. Lee, J. Kim, M. Lee, A real-time ECG data compression and transmission algorithm for an e-health device, IEEE Trans. Biomed. Eng. 58 (2011) 2448–2455, doi:10.1109/TBME.2011.2156794.
[14] J.L. Ma, T.T. Zhang, M.C. Dong, A novel ECG data compression method using adaptive Fourier decomposition with security guarantee in e-health applications, IEEE J. Biomed. Health Inform. 19 (2015) 986–994, doi:10.1109/JBHI.2014.2357841.
[15] A. Pandey, B. Singh, S Sain, N. Sood, A joint application of optimal threshold based discrete cosine transform and ASCII encoding for ECG data compression with its inherent encryption, Australas. Phys. Eng. Sci. Med. 39 (2016) 833–855, doi:10.1007/s13246-016-0476-4.
[16] S. Raj, K.C. Ray, Sparse representation of ECG signals for automated recognition of cardiac arrhythmias, Expert Syst. Appl. 10 (2018) 49–64, doi:10.1016/j.eswa.2018.03.038.
[17] L. Rebollo-Neira, D. Černá, Wavelet based dictionaries for dimensionality reduction of ECG signals, Biomed. Signal Process. Control 54 (101593) (2019) 101593 article No., doi:10.1016/j.bspc.2019.101593.
[18] L. Rebollo-Neira, D. Lowe, Optimized orthogonal matching pursuit approach, IEEE Signal Process. Lett. 9 (2002) 137–140, doi:10.1109/LSP.2002.1001652.
[19] L. Rebollo-Neira, Cooperative greedy pursuit strategies for sparse signal representation by partitioning, Signal Process. 125 (2016) 365–375, doi:10.1016/j.sigpro.2016.02.008.
[20] L. Rebollo-Neira, Effective high compression of ECG signals at low level distortion, Sci. Rep. 9 (4564) (2019) 4564 article No., doi:10.1038/s41598-019-40350-x.
[21] L. Rebollo-Neira, Z. Xu, Sparse signal representation by adaptive non-uniform B-spline dictionaries on a compact interval, Signal Process. 90 (2010) 2308–2313, doi:10.1016/j.sigpro.2010.02.004.
[22] G. Strang, Wavelets and dilation equations: a brief introduction, SIAM Rev. 31 (4) (1989) 614–627 https://www.jstor.org/stable/2031540.
[23] J. Williams, K. Amaratunga, Introduction to wavelets in engineering, Int. J. Numer. Methods Eng. 37 (1994) 2365–2388 Wiley, doi:10.1002/nme.1620371403 . hal-01311772.
[24] C. Tan, L. Zhang, H. Wu, A novel Blaschke unwinding adaptive Fourier decomposition based signal compression algorithm with application on ECG signals, IEEE J. Biomed. Health Inform. 23 (2019) 672–682, doi:10.1109/JBHI.2018.2817192.