PHILOSOPHICAL TRANSACTIONS A

rsta.royalsocietypublishing.org

Research

Cite this article: Hou TY, Shi Z. 2016 Sparse time-frequency decomposition based on dictionary adaptation. *Phil. Trans. R. Soc. A* **374**: 20150192.
http://dx.doi.org/10.1098/rsta.2015.0192

Accepted: 29 September 2015

One contribution of 13 to a theme issue 'Adaptive data analysis: theory and applications'.

Subject Areas:
applied mathematics

Keywords:
Sparse time-frequency decomposition, instantaneous frequency, dictionary adaptation

Author for correspondence:
Zuoqiang Shi
e-mail: zqshi@math.tsinghua.edu.cn

Sparse time-frequency decomposition based on dictionary adaptation

Thomas Y. Hou[1] and Zuoqiang Shi[2]

[1]Applied and Comput. Math, MC 9-94, Caltech, Pasadena, CA 91125, USA
[2]Yau Mathematical Sciences Center, Tsinghua University, Beijing 100084, People's Republic of China

In this paper, we propose a time-frequency analysis method to obtain instantaneous frequencies and the corresponding decomposition by solving an optimization problem. In this optimization problem, the basis that is used to decompose the signal is not known *a priori*. Instead, it is adapted to the signal and is determined as part of the optimization problem. In this sense, this optimization problem can be seen as a dictionary adaptation problem, in which the dictionary is adaptive to one signal rather than a training set in dictionary learning. This dictionary adaptation problem is solved by using the augmented Lagrangian multiplier (ALM) method iteratively. We further accelerate the ALM method in each iteration by using the fast wavelet transform. We apply our method to decompose several signals, including signals with poor scale separation, signals with outliers and polluted by noise and a real signal. The results show that this method can give accurate recovery of both the instantaneous frequencies and the intrinsic mode functions.

1. Introduction

Nowadays we must process a massive amount of data in our daily life and scientific research. Data analysis methods have played an important role in processing and analysing these data. Most data analysis methods use a predetermined basis, including the most commonly used Fourier transform and wavelet transform. While these data analysis methods are very efficient in processing data, each component of the decomposition in general does not reveal the intrinsic physical information of these data due to the presence of harmonics in the decomposition. For example, application of these traditional data analysis methods to a modulated oscillatory chirp signal would produce many components. Thus, it is essential to develop a truly adaptive data analysis method that can extract hidden physical information such as trend and time varying cycles from these data and preserve the integrity of the physically meaningful components. To achieve this, we need to use a data-driven basis that is adapted to the signal instead of being determined *a priori*.

The empirical mode decomposition (EMD) method, which was proposed by Prof. Norden E. Huang in 1998 [1,2], provides an efficient adaptive method to analyse nonlinear and non-stationary data. In the EMD method, the signal is decomposed to several intrinsic mode functions (IMFs) by the so-called sifting process. The EMD method is empirical in nature. Several methods with more clear mathematical structures have been proposed as a variant of the EMD method (for example the synchrosqueezed wavelet transform [3], empirical wavelet transform [4] and variational mode decomposition [5]). Inspired by the EMD method and the recently developed compressed (compressive) sensing theory, we have proposed a data-driven time-frequency analysis method [6–8]. In this method, we formulate the problem as a nonlinear optimization problem and decompose the signal by looking for the sparsest representation of a multiscale signal over a dictionary consisting of all IMFs.

In our data-driven time-frequency analysis, we decompose a given signal $f(t)$ into the following form:

$$f(t)=\sum_{j=1}^{M} a_j(t)\cos\theta_j(t), \quad t\in\mathbb{R}, \tag{1.1}$$

where $a_j(t)$, $\theta_j(t)$ are smooth functions, $\theta_j'(t)>0$, $j=1,\ldots,M$, and M is an integer that is unknown *a priori*. We assume that $a_j(t)$ and $\theta_j'(t)$ are less oscillatory than $\cos\theta_j(t)$. We call $a_j(t)\cos\theta_j(t)$ the intrinsic mode functions (IMFs) and $\theta_j'(t)$, $j=1,\ldots,M$ the instantaneous frequencies [1]. The objective of our data-driven time-frequency analysis is to extract the IMFs and their instantaneous frequencies.

In [7], we proposed to decompose the signal by solving the following optimization problem:

$$\min_{(a_k)_{1\le k\le M},\,(\theta_k)_{1\le k\le M}} M, \quad \text{subject to: } f=\sum_{k=1}^{M} a_k\cos\theta_k, \quad a_k\cos\theta_k\in\mathcal{D}, \tag{1.2}$$

where $\mathcal{D}$ is the dictionary consisting of all IMFs (see [7] for its precise definition). To simplify the notation, when no ambiguity arises, we drop the argument t from the functions $a_k(t)$ and $\theta_k(t)$ and in the rest of this paper, we will use this notation to make the formula more concise. Further, an efficient algorithm based on matching pursuit and fast Fourier transform has been proposed to solve the above nonlinear optimization problem. In a subsequent paper [9], we proved the convergence of the algorithm in [7] for periodic data that satisfy a certain scale separation property.

In this paper, we will introduce another formulation to get the sparsest time-frequency decomposition based on dictionary adaptation. In this formulation, we assume that we have already known the number of IMFs, M. Then, we can obtain the desirable decomposition by solving the following optimization problem:

$$\min_{\mathbf{x},\boldsymbol{\theta}_1\ldots,\boldsymbol{\theta}_M} \|\mathbf{x}\|_1, \quad \text{subject to } \boldsymbol{\Phi}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M}\cdot\mathbf{x}=f, \tag{1.3}$$

where

$$\boldsymbol{\Phi}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M}=[\boldsymbol{\Phi}_{\boldsymbol{\theta}_1},\ldots,\boldsymbol{\Phi}_{\boldsymbol{\theta}_M}], \tag{1.4}$$

and $\boldsymbol{\Phi}_{\boldsymbol{\theta}_j}$, $j=1,\ldots,M$ is the basis to decompose the signal f. The specific form of the basis as a function of $\boldsymbol{\theta}_j$ will be given in (2.3). And we use bold font to denote the vectors (lowercase) or matrices (uppercase).

In (1.3), the basis $\boldsymbol{\Phi}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M}$ is not known *a priori*. It is determined by the phase functions $\boldsymbol{\theta}_j$, $j=1,\ldots,M$ and the phase functions are adaptive to the data. We need to solve not only the optimal coefficients $\mathbf{x}$ but also the optimal basis $\boldsymbol{\Phi}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M}$. In this sense, the problem described above seems to be a dictionary learning problem. Many efficient dictionary learning methods have been proposed [10–14]. But the problem we study here has some important differences from the dictionary learning problem. In a dictionary learning problem, a common set-up starts with a training set, a collection of training vectors. The atoms of the dictionary can be any function. For the problem we consider here, we only have one signal not a training set. Moreover, the atoms of the dictionary in our problem are given in the form of $\cos(\boldsymbol{\theta}_j)$. If we do not put any constraint on the atoms of the dictionary, we may get only trivial decompositions. In order to get a reasonable result, the atoms of the dictionary are restricted to be nonlinear functionals of the phase function $\boldsymbol{\theta}_j$. At the same time, the phase functions are confined to a low dimensional space to make sure that the overall degree of freedom is not too large. As we will demonstrate later, we can still get a reasonable decomposition with only one measurement of the signal. To distinguish our problem from the dictionary learning problem, we call it dictionary adaptation.

We need to develop a new method to solve this non-convex optimization problem. The key part is to find the phase functions. Once the phase functions are known, we need to solve a l_1 optimization problem to get the decomposition. Based on this observation, we develop an iterative algorithm to solve (1.3). This algorithm starts from one initial guess of the phase functions. In each step, the phase functions are fixed and one l_1 optimization problem is solved. The phase functions will be updated in the next step. This iteration is repeated until the changing of the phase function between two consecutive iterations is smaller than a given tolerance. In each step, the augmented Lagrangian multiplier (ALM) method is used to solve the l_1 optimization problem. We further accelerate the ALM method in each iteration by using the fast wavelet transform. This method can be also generalized to decompose signals that contain outliers by enlarging the dictionary to include impulses.

We will demonstrate the effectiveness of our method by applying it to decompose several multiscale data, including synthetic data and real data. For the data that we consider in this paper, we will demonstrate that we can recover both the instantaneous frequencies and the IMFs very accurately. Even for those signals that are polluted by noise, we still can approximate the instantaneous frequencies and the IMFs with reasonable accuracy comparable to the noise level.

The remaining of the paper is organized as follows. In §2, we give the formulation of our problem. In §3, an iterative algorithm is introduced to solve the nonlinear optimization problem (1.3). An accelerating procedure is introduced based on the fast wavelet transform in §4. In §5, we generalize this method to deal with signals that have outliers. In §6, several numerical results are presented to demonstrate the effectiveness of our method. Finally, some concluding remarks are made in §7.

2. Formulation based on dictionary adaptation

In this section, we will set up the framework of the data-driven time-frequency decomposition. First, we construct a dictionary to represent the signal $f(t) \in L^2(\mathbb{R})$ which satisfies the model (1.1).

In this paper, we construct the dictionary following an idea similar to that in [7]. However, we will use the wavelet basis instead of the overcomplete Fourier basis used in [7]. We choose the wavelet basis because there are fast decomposition and reconstruction algorithms. This feature makes our algorithm very efficient. Another advantage is that the wavelet basis can handle non-periodic functions much better.

Let $\{V_l\}_{l\in\mathbb{Z}}$ be a multi-resolution approximation of $L^2(\mathbb{R})$ and φ the associated scaling function, ψ the corresponding wavelet function. Assume that φ is real and $\hat{\varphi}$ has compact support, $\mathrm{supp}(\hat{\varphi}) = [-s_\varphi, s_\varphi]$, where $\hat{\varphi}$ is the Fourier transform of φ defined below,

$$\hat{\varphi}(k) = \frac{1}{2\pi}\int_{\mathbb{R}} \varphi(t)\,\mathrm{e}^{-\mathrm{i}kt}\,\mathrm{d}t.$$

For each $1 \le j \le M$, we assume that $\theta_j' > 0$. Since θ_j is a strictly monotonically increasing function, there is a one-to-one mapping between the physical time t and $\theta_j(t)$. Thus, we can use θ_j

as a new coordinate and represent the signal as a function of θ_j. Then we can define the following wavelet basis in the θ_j-coordinate:

$$\mathcal{A}_{\theta_j} = \{\psi_{l,n}(\theta_j)_{\substack{l,n\in\mathbb{Z},\\ 0<l\le l_0}}, \varphi_{l_0,n}(\theta_j)_{n\in\mathbb{Z}}\}, \tag{2.1}$$

where l_0 is a positive integer associated with the lowest frequency of the envelope. In practice, l_0 is usually determined by the time range of the signal. And

$$\psi_{l,n}(\theta_j) = \frac{1}{\sqrt{2^l s_\varphi}}\psi\left(\frac{\theta_j}{2^l s_\varphi} - n\right) \quad \text{and} \quad \varphi_{l,n}(\theta_j) = \frac{1}{\sqrt{2^l s_\varphi}}\varphi\left(\frac{\theta_j}{2^l s_\varphi} - n\right). \tag{2.2}$$

In this paper, we use the Meyer wavelet to construct the basis.

Then the corresponding IMF, $a_j \cos\theta_j$, can be represented using the following basis:

$$\mathcal{D}_{\theta_j} = \{(\psi_{l,n}(\theta_j)\cos\theta_j)_{\substack{l,n\in\mathbb{Z},\\ 0<l\le l_0}}, (\varphi_{l_0,n}(\theta_j)\cos\theta_j)_{n\in\mathbb{Z}}\}. \tag{2.3}$$

This, in turn, implies that the whole signal $f(t)$ given in ((1.1)) lies in the space spanned by the following basis:

$$\mathcal{D}_{\theta_1,\ldots,\theta_M} = \{\mathcal{D}_{\theta_1}, \ldots, \mathcal{D}_{\theta_M}\}. \tag{2.4}$$

We assume that the envelope is sufficiently smooth, $a_j \in C^\infty$. Thus, a_j has a sparse representation over the wavelet basis $\mathcal{A}_{\theta_j}$. Consequently, the signal $f(t)$ would have a sparse representation over $\mathcal{D}_{\theta_1,\ldots,\theta_M}$. Based on this observation, we propose to decompose the signal $f(t)$ in ((1.1)) by looking for the sparsest representation over $\mathcal{D}_{\theta_1,\ldots,\theta_M}$:

$$\min_{c_j,\theta_1\ldots,\theta_M} \sum_j |c_j|, \quad \text{subject to: } \sum_j c_j\phi_j(t) = f(t), \quad \phi_j(t) \in \mathcal{D}_{\theta_1,\ldots,\theta_M}, \tag{2.5}$$

where c_j, $j = 1, 2, \ldots$ are the coefficients of the signal $f(t)$ in the dictionary $\mathcal{D}_{\theta_1,\ldots,\theta_M}$.

In this problem, not only the coefficients are unknown, but the basis over which the signal has a sparse representation is also unknown *a priori*. The basis is determined by the phase functions $\theta_1, \ldots, \theta_M$. Both the coefficients c_j and the phase functions need to be determined in the optimization process.

The above formulation is presented in the continuous version. Next, we will introduce a discrete version. Suppose the signal $f(t)$ is sampled over N discrete points, $t_j = jh$, $j = 1, \ldots, N$, $h = T/N$ and $[0, T]$ is the sample interval. Here, we assume that the sample points are dense enough such that the signal can be interpolated to any other grids with high accuracy. And we use the bold font to denote the corresponding discrete samples of the continuous function, for instance, $\mathbf{f} = [f(t_1), \ldots, f(t_N)]$. Then the discrete version of (2.5) is given as

$$\min_{\mathbf{x},\boldsymbol{\theta}_1\ldots,\boldsymbol{\theta}_M} \|\mathbf{x}\|_1, \quad \text{subject to: } \boldsymbol{\Phi}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M} \cdot \mathbf{x} = \mathbf{f}, \tag{2.6}$$

where $\boldsymbol{\Phi}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M} = [\boldsymbol{\Phi}_{\boldsymbol{\theta}_1}, \ldots, \boldsymbol{\Phi}_{\boldsymbol{\theta}_M}]$ and $\boldsymbol{\Phi}_{\boldsymbol{\theta}_j}$, $j = 1, \ldots, M$ are matrices corresponding to the dictionaries $\mathcal{D}_{\theta_j}$, $j = 1, \ldots, M$, i.e. each column of $\boldsymbol{\Phi}_{\boldsymbol{\theta}_j}$ is discrete samples of basis function in $\mathcal{D}_{\theta_j}$. Here, we denote $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_M]^{\mathrm{T}}$ and $\mathbf{x}_j$, $j = 1, \ldots, M$ are the corresponding coefficients of $\mathbf{a}_j \cos\boldsymbol{\theta}_j$ over $\boldsymbol{\Phi}_{\boldsymbol{\theta}_j}$. Moreover, we define

$$\boldsymbol{\Pi}_{\boldsymbol{\theta}_j} = [\boldsymbol{\psi}_{l,n}(\boldsymbol{\theta}_j)_{\substack{l,n\in\mathbb{Z},\\ 0<l\le l_0}}, \boldsymbol{\varphi}_{l_0,n}(\boldsymbol{\theta}_j)_{n\in\mathbb{Z}}]. \tag{2.7}$$

To simplify the notation, we introduce an operator $\star$, which is defined as follows: for any $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^{n\times m}$,

$$\mathbf{x} \star \mathbf{y} = \mathrm{diag}(\mathbf{x}) \cdot \mathbf{y}. \tag{2.8}$$

3. An algorithm based on the Augmented Lagrangian Multiplier method

Inspired by the algorithm in [7], we propose an iterative algorithm, algorithm 1, to solve the nonlinear l_1 optimization problem (2.6). Algorithm 1 is essentially based on the Gauss–Newton iteration which is derived by performing a local linearization around the phase functions in the current step. Suppose $\theta^*(t)$ is the exact phase function and $\theta^n(t)$ is a good approximation. By keeping only the linear terms in the Taylor expansion, we obtain the following linearization around $\theta^n(t)$,

$$\cos\theta^*(t) \approx \cos\theta^n(t) - \Delta\theta^n(t)\sin\theta^n(t),$$

where $\Delta\theta^n(t) = \theta^*(t) - \theta^n(t)$. Both the matrix $\boldsymbol{\Psi}_{\boldsymbol{\theta}_1^n,\ldots,\boldsymbol{\theta}_M^n}$ in (3.1) and $\mathbf{b}_j$ in (3.2) come from this linearization procedure. The coefficients, $\mathbf{b}_j$, are just auxiliary functions in the iteration and will tend to zero as the iteration converges.

Algorithm 1. (Gauss–Newton-type iteration).

Input: Initial guess of phase functions $\boldsymbol{\theta}_j^0$, $j=1,\ldots,M$, $\eta = l_0$, where l_0 is same as that in (2.1).
Output: Phase functions and corresponding envelopes: $\boldsymbol{\theta}_j$, $\mathbf{a}_j$, $j=1,\ldots,M$.

```
1: while $\eta \geq 1$ do
2:   while $\sum_{j=1}^{M} \|\boldsymbol{\theta}_j^{n+1} - \boldsymbol{\theta}_j^n\|_2 > \epsilon_0$ do
3:     Solve the following $l_1$ optimization problem:
         $$\left(\widetilde{\mathbf{a}}^{n+1}, \widetilde{\mathbf{b}}^{n+1}\right) = \operatorname{argmin}_{\mathbf{x},\mathbf{y}}(\|\mathbf{x}\|_1 + \|\mathbf{y}\|_1), \quad (3.1)$$
         $$\text{subject to} \quad \boldsymbol{\Phi}_{\boldsymbol{\theta}_1^n,\ldots,\boldsymbol{\theta}_M^n}\cdot\mathbf{x} + \boldsymbol{\Psi}_{\boldsymbol{\theta}_1^n,\ldots,\boldsymbol{\theta}_M^n}\cdot\mathbf{y} = \mathbf{f},$$
       where $\boldsymbol{\Psi}_{\boldsymbol{\theta}_1^n,\ldots,\boldsymbol{\theta}_M^n} = [\boldsymbol{\Psi}_{\boldsymbol{\theta}_1^n},\ldots,\boldsymbol{\Psi}_{\boldsymbol{\theta}_M^n}]$ and
         $$\boldsymbol{\Psi}_{\boldsymbol{\theta}_j^n} = (\sin\boldsymbol{\theta}_j)\star\boldsymbol{\Pi}_{\boldsymbol{\theta}_j}, \quad j=1,\ldots,M.$$
4:     Calculate the envelopes $\mathbf{a}_j$, $\mathbf{b}_j$, $j=1,\ldots M$:
         $$\mathbf{a}_j = \boldsymbol{\Pi}_{\boldsymbol{\theta}_j^n}\cdot\widetilde{\mathbf{a}}_j^{n+1}, \quad \mathbf{b}_j = \boldsymbol{\Pi}_{\boldsymbol{\theta}_j^n}\cdot\widetilde{\mathbf{b}}_j^{n+1}, \quad (3.2)$$
5:     Update $\boldsymbol{\theta}_j^n$, $j=1,\ldots,M$:
         $$\Delta\boldsymbol{\theta}_j' = P_{V_\eta(\boldsymbol{\theta}_j^n)}\left(\frac{d}{dt}\left(\arctan\left(\frac{\mathbf{b}_j}{\mathbf{a}_j}\right)\right)\right),$$
         $$\Delta\boldsymbol{\theta}_j = \int \Delta\theta_j'(s)\mathrm{d}s, \quad \boldsymbol{\theta}_j^{n+1} = \boldsymbol{\theta}_j^n - \beta_j\Delta\boldsymbol{\theta}_j,$$
       where $\beta_j \in [0,1]$ is chosen to make sure that $\boldsymbol{\theta}_j^{n+1}$ is monotonically increasing:
         $$\beta_j = \max\left\{\alpha\in[0,1] : \frac{d}{dt}(\boldsymbol{\theta}_j^n - \alpha\Delta\boldsymbol{\theta}_j) \geq 0\right\}.$$
       Here $P_{V_\eta(\boldsymbol{\theta}_j^n)}$ is the projection operator to the space $V_\eta(\boldsymbol{\theta}_j^n)$ and
         $$V_\eta(\boldsymbol{\theta}) = \operatorname{span}[\psi_{l,n}(\boldsymbol{\theta})_{\substack{l,n\in\mathbb{Z},\\ \eta<l\leq l_0}}, \varphi_{l_0,n}(\boldsymbol{\theta})_{n\in\mathbb{Z}}].$$
       and $\frac{d}{dt}$ is approximated by a central difference scheme, the integral is approximated by the trapezoid rule.
6:   end while
7:   $\eta = \eta - 1$.
8: end while
```

In some sense, algorithm 1 is similar in spirit to the K-SVD algorithm [10]. The step to solve the l_1 optimization problem is similar to the sparse coding stage in the K-SVD method and the step to update the phase functions is similar to the codebook update stage. The main difference is in the codebook update stage. In our problem, the atoms of the dictionary are not arbitrary functions. They are parametrized by the phase functions. For this reason, we cannot use singular value decomposition (SVD) to update the dictionary directly. Instead, we use the Gauss–Newton algorithm to update the phase functions, which in turn updates the dictionary accordingly.

In the step to update the phase functions, we choose to update the instantaneous frequency (derivative of the phase function) instead of updating the phase function directly. This is because arctan $(\mathbf{b}_j/\mathbf{a}_j)$ could be discontinuous if we do not add or subtract 2π at some particular points. Here, β_j at step 5 is used to make sure that the instantaneous frequency being positive during the iteration to make sure that the phase function remains monotonically increasing. In other words, we update $\boldsymbol{\theta}_j$ along the direction of $\Delta\boldsymbol{\theta}_j$ under the constraint that $\boldsymbol{\theta}_j' > 0$.

The optimization problem we want to solve is non-convex. There are many local minima. If the initial guess is not good enough, the iteration may converge to a local minimum instead of the global minimum. However, we cannot guarantee to get a good initial guess unless we have obtained some *a priori* information about the signal. To abate the dependence on the initial guess, we introduce a projection operator $P_{V_\eta(\boldsymbol{\theta}_j^n)}$ and a parameter η. The value of η is gradually decreased during the iteration. When η is large, $\Delta\boldsymbol{\theta}'$ is confined to a small space. In this small space, the objective functional has fewer local minima. The iteration may find a good approximation for $\Delta\boldsymbol{\theta}'$. By gradually decreasing η, we enlarge the space for $\Delta\boldsymbol{\theta}'$ which allows for the small-scale structures of $\Delta\boldsymbol{\theta}'$ to develop gradually during the iterations. This procedure mimics a continuation method and works very well in practice.

In the numerical examples presented in §6, we obtain the initial guess using a simple approach based on the Fourier transform. More precisely, we obtain the initial guess by estimating the frequencies by which the Fourier coefficients have the largest energy.

In algorithm 1, the most expensive part is to solve the l_1 optimization problem (3.1) at step 3. In this paper, we use an ALM algorithm [15] to solve (3.1). In order to simplify the notations, we denote

$$\Theta_{\boldsymbol{\theta}_1^n,\ldots,\boldsymbol{\theta}_M^n} = [\Theta_{\boldsymbol{\theta}_1^n}, \ldots, \Theta_{\boldsymbol{\theta}_M^n}], \tag{3.3}$$

where

$$\Theta_{\boldsymbol{\theta}_j^n} = [\boldsymbol{\Phi}_{\boldsymbol{\theta}_j^n}, \boldsymbol{\Psi}_{\boldsymbol{\theta}_j^n}], \quad j = 1, \ldots, M.$$

The ALM method operates on the augmented Lagrangian

$$L(\mathbf{p}, \mathbf{q}) = \|\mathbf{p}\|_1 + \langle \mathbf{q}, \mathbf{f} - \Theta_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M}\mathbf{p}\rangle + \frac{\mu}{2}\|\mathbf{f} - \Theta_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M}\mathbf{p}\|_2^2. \tag{3.4}$$

A generic Lagrange multiplier algorithm [15] would solve (3.1) by repeatedly setting $\mathbf{p}^{k+1} = \operatorname{argmin}_{\mathbf{p}} L(\mathbf{p}, \mathbf{q}^k)$, and then updating the Lagrange multiplier via $\mathbf{q}^{k+1} = \mathbf{q}^k + \mu(\mathbf{f} - \Theta_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M}\mathbf{p}^k)$.

In this iteration, solving $\min_{\mathbf{p}} L(\mathbf{p}, \mathbf{q}^k)$ is also very time consuming. Note that the matrix $\Theta_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_M} = [\Theta_{\boldsymbol{\theta}_1}, \ldots, \Theta_{\boldsymbol{\theta}_M}]$ is the combination of M matrices with smaller size. It is natural to use the following sweeping algorithm, algorithm 2, to solve $\min_{\mathbf{p}} L(\mathbf{p}, \mathbf{q}^k)$ iteratively.

Theoretically, we need to run the above sweeping process several times until the solution converges, but in practical computations, in order to save the computational cost, we only run the sweeping process once. Combining this idea with the augmented Lagrange multiplier method, we obtain algorithm 3 to solve the l_1 optimization problem (3.1):

4. A fast algorithm based on the discrete wavelet transform

In this section, we propose an approximate solver to accelerate the computation of $\mathbf{p}_j^{k+1} = \operatorname{argmin}_{\mathbf{p}_j} \|\mathbf{p}_j\|_1 + (\mu/2)\|\mathbf{r}_j^k + \mathbf{q}^k/\mu - \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j\|_2^2$, which is the most expensive step in algorithm 1.

In this paper, we only consider the signal which is well resolved by the samples and the samples are uniformly distributed in time and the total number of the samples is N. Based on

Algorithm 2. (Sweeping).

```
Input: $\mathbf{p}_j^0 = 0,\ j = 1, \dots, M,\ \mu > 0$.
1: while not converge do
2:     for j=1:M do
3:         Compute $\mathbf{r}_j^m = \mathbf{f} - \sum_{l=1}^{j-1} \Theta_{\boldsymbol{\theta}_l} \mathbf{p}_l^{m+1} - \sum_{l=j+1}^{M} \Theta_{\boldsymbol{\theta}_l} \mathbf{p}_l^m$.
4:         Compute $\mathbf{p}_j^{m+1} = \operatorname{argmin}_{\mathbf{p}_j} \|\mathbf{p}_j\|_1 + \frac{\mu}{2}\|\mathbf{r}_j^m + \mathbf{q}^k/\mu - \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j\|_2^2$.
5:     end for
6: end while
```

Algorithm 3. (Sweeping ALM).

```
Input: $\mathbf{p}_j^0 = 0,\ j = 1, \dots, M,\ \mathbf{q}^0 = 0,\ \mu > 0$
1: while not converge do
2:     for j=1:M do
3:         Compute $\mathbf{r}_j^k = \mathbf{f} - \sum_{l=1}^{j-1} \Theta_{\boldsymbol{\theta}_l} \mathbf{p}_l^{k+1} - \sum_{l=j+1}^{M} \Theta_{\boldsymbol{\theta}_l} \mathbf{p}_l^k$.
4:         Compute $\mathbf{p}_j^{k+1} = \underset{\mathbf{p}_j}{\operatorname{argmin}} \|\mathbf{p}_j\|_1 + \frac{\mu}{2}\|\mathbf{r}_j^k + \mathbf{q}^k/\mu - \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j\|_2^2$.
5:     end for
6:     $\mathbf{q}^{k+1} = \mathbf{q}^k + \mu\left(\mathbf{f} - \sum_{j=1}^{M} \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j^{k+1}\right)$.
7: end while
```

these assumptions, we can approximate the continuous integral by the discrete summation and the integration error is negligible. This greatly simplifies our calculations.

Now, we turn to simplify the optimization problem. First, we replace the standard L^2 norm by a weighted L^2 norm which gives the following approximation:

$$\mathbf{p}_j^{k+1} = \underset{\mathbf{p}_j}{\operatorname{argmin}} \|\mathbf{p}_j\|_1 + \frac{\mu}{2}\left\|\mathbf{r}_j^k + \frac{\mathbf{q}^k}{\mu} - \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j\right\|_{2,\boldsymbol{\theta}_j}^2, \tag{4.1}$$

where $\|\mathbf{g}\|_{2,\boldsymbol{\theta}_j}^2 = \sum_i \mathbf{g}(t_i)^2 \boldsymbol{\theta}_j'(t_i)$, $\boldsymbol{\theta}_j'$ is the derivative of $\boldsymbol{\theta}_j$ which is computed by a central difference scheme, $\mathbf{g}(t_i)$ is the ith element of vector $\mathbf{g}$.

Using the fact that $\operatorname{supp}(\hat{\varphi}) = (-s_\varphi, s_\varphi)$, it is easy to check that the columns of the matrix $\Theta_{\boldsymbol{\theta}}$ are orthonormal under the weighted discrete inner product

$$\langle \mathbf{g}, \mathbf{h} \rangle_{\boldsymbol{\theta}} = \sum_{i=1}^{N} \mathbf{g}(t_i)\mathbf{h}(t_i)\boldsymbol{\theta}'(t_i) = \mathbf{g}^{\mathrm{T}} \cdot (\boldsymbol{\theta}' \star \mathbf{h}). \tag{4.2}$$

Using this property, it is easy to derive the following equality:

$$\|\Theta_{\boldsymbol{\theta}} \cdot \mathbf{x}\|_{2,\boldsymbol{\theta}} = \|\mathbf{x}\|_2. \tag{4.3}$$

We can also define the projection operator $P_{V(\boldsymbol{\theta})}$ to the $V(\boldsymbol{\theta})$ space. Here, $V(\boldsymbol{\theta})$ is the linear space spanned by the columns of the matrix $\Theta_{\boldsymbol{\theta}}$ and $P_{V(\boldsymbol{\theta})}$ is the projection operator to $V(\boldsymbol{\theta})$. Since the columns of the matrix $\Theta_{\boldsymbol{\theta}}$ are orthonormal under the weighted inner product (4.2), the projection $P_{V(\boldsymbol{\theta})}$ can be calculated as follows:

$$P_{V(\boldsymbol{\theta})}(\mathbf{r}) = \Theta_{\boldsymbol{\theta}} \cdot \hat{\mathbf{r}}, \quad \hat{\mathbf{r}} = \Theta_{\boldsymbol{\theta}}^{\mathrm{T}} \cdot [\boldsymbol{\theta}' \star \mathbf{r}]. \tag{4.4}$$

Now, we are ready to show that the optimization problem (4.1) can be solved explicitly by the shrinkage operator. To simplify the notation, we denote $\mathbf{w} = \mathbf{r}_j^k + \mathbf{q}^k/\mu$,

$$
\begin{aligned}
&\operatorname*{argmin}_{\mathbf{p}_j} \|\mathbf{p}_j\|_1 + \frac{\mu}{2}\left\| \mathbf{r}_j^k + \frac{\mathbf{q}^k}{\mu} - \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j \right\|_{2,\boldsymbol{\theta}_j}^2 \\
&= \operatorname*{argmin}_{\mathbf{p}_j} \|\mathbf{p}_j\|_1 + \frac{\mu}{2}\|\Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j - P_{V(\boldsymbol{\theta}_j)}(\mathbf{w})\|_{2,\boldsymbol{\theta}_j}^2 \\
&= \operatorname*{argmin}_{\mathbf{p}_j} \|\mathbf{p}_j\|_1 + \frac{\mu}{2}\|\Theta_{\boldsymbol{\theta}_j}\cdot[\mathbf{p}_j - \Theta_{\boldsymbol{\theta}_j}^{\mathrm{T}}\cdot[\boldsymbol{\theta}_j'\star\mathbf{w}]]\|_{2,\boldsymbol{\theta}_j}^2 \\
&= \operatorname*{argmin}_{\mathbf{p}_j} \|\mathbf{p}_j\|_1 + \frac{\mu}{2}\|\mathbf{p}_j - \Theta_{\boldsymbol{\theta}_j}^{\mathrm{T}}\cdot[\boldsymbol{\theta}_j'\star\mathbf{w}]\|_2^2 \\
&= \mathcal{S}_{\mu^{-1}}\left(\Theta_{\boldsymbol{\theta}_j}^{T}\cdot\left[\boldsymbol{\theta}_j'\left(\mathbf{r}_j^k + \frac{\mathbf{q}^k}{\mu}\right)\right]\right),
\end{aligned}
\tag{4.5}
$$

where $\mathcal{S}_\tau$ is the shrinkage operator defined below:

$$\mathcal{S}_\tau(x) = \operatorname{sgn}(x)\max(|x| - \tau, 0). \tag{4.6}$$

Note that the matrix vector product in (4.5) has the following structure by the definition of $\Theta_{\boldsymbol{\theta}_j}$ in (3.3):

$$\Theta_{\boldsymbol{\theta}_j}^{\mathrm{T}}\cdot(\boldsymbol{\theta}_j'\star\mathbf{r}) = [\boldsymbol{\Pi}_{\boldsymbol{\theta}_j}^{\mathrm{T}}\cdot(\cos\boldsymbol{\theta}_j\star(\mathbf{r}\star\boldsymbol{\theta}_j')), \boldsymbol{\Pi}_{\boldsymbol{\theta}_j}^{\mathrm{T}}\cdot(\sin\boldsymbol{\theta}_j\star(\mathbf{r}\star\boldsymbol{\theta}_j'))]^{\mathrm{T}}.$$

This is nothing but the wavelet transform of $\sin\boldsymbol{\theta}_j\star\mathbf{r}$ and $\cos\boldsymbol{\theta}_j\star\mathbf{r}$ in the $\boldsymbol{\theta}_j$-coordinate, since the columns of $\boldsymbol{\Pi}_{\boldsymbol{\theta}_j}$ are standard wavelet basis in the $\boldsymbol{\theta}_j$-coordinate. Then this product can be computed efficiently by interpolating $\cos\boldsymbol{\theta}_j\star\mathbf{r}$ and $\sin\boldsymbol{\theta}_j\star\mathbf{r}$ to the uniform grid in the $\boldsymbol{\theta}_j$ coordinate and employing the fast wavelet transform.

Summarizing the above discussion, we obtain algorithm 4 based on the fast wavelet transform to solve the optimization problem (3.1).

Remark 4.1. The final algorithm described above is based on employing the soft shrinkage operator iteratively. Essentially, it is a proximal splitting algorithm which shares many common ideas with some existing l_1 optimization methods, including the split Bregman method [16], proximity algorithms [17] and proximal splitting methods [18]. Inspired by these methods, we adopt the splitting technique to decompose the original problem to some easier subproblems and solve the original problem by solving these subproblems iteratively. In each step, we need to compute a proximity operator. The main novelty of our algorithm is in the computation of the proximity operator. Here, we use the special structure of the matrix $\Theta_{\boldsymbol{\theta}_j}$, i.e. the orthonormality of the columns in the $\boldsymbol{\theta}_j$-coordinate such that the proximity operator is just a simple soft shrinkage operator. This observation gives rise to a much more efficient algorithm which is designed specifically for our optimization problem. For our particular application, our method is much more efficient than the current l_1 optimization algorithms.

5. Generalization for signals with outliers

One advantage of the formulation (2.6) is that it can be generalized to deal with more complicated data with some minor modifications. In this section, we will give one generalization for signals with outliers. For outliers, we mean that at some sample points, the error of the measurement is large. In this paper, we assume that the number of outliers is small and distributed over the whole interval at random.

In order to deal with this kind of signal, we have to enlarge the dictionary since the outliers are not sparse over the time-frequency dictionary. Under the assumption that the number of outliers is small, we know that the outliers are sparse over the basis consisting of the impulses $\delta[n-i]$,

Algorithm 4. (Sweeping ALM accelerated by the fast wavelet transform).

```
Input: $a^0_{\boldsymbol{\theta}_j} = b^0_{\boldsymbol{\theta}_j} = 0,\ j = 1, \dots, M,\ \mathbf{q}^0 = 0$.
1: while not converge do
2:     for j=1:M do
3:         Compute
               $$\mathbf{r}^n_j = \mathbf{f} - \sum_{l=1}^{j-1} (\mathbf{a}^{n+1}_{\boldsymbol{\theta}_l} \star \cos\boldsymbol{\theta}_l + \mathbf{b}^{n+1}_{\boldsymbol{\theta}_l} \star \sin\boldsymbol{\theta}_l) - \sum_{l=j+1}^{M} (\mathbf{a}^{n}_{\boldsymbol{\theta}_l} \star \cos\boldsymbol{\theta}_l + \mathbf{b}^{n}_{\boldsymbol{\theta}_l} \star \sin\boldsymbol{\theta}_l).$$
4:         Interpolate $\mathbf{R} = \mathbf{r}^n_j + \mathbf{q}^n/\mu$ from $\{t_i\}_{i=1}^N$ in the physical space to a uniform mesh in the
           $\boldsymbol{\theta}_j$-coordinate to get $\mathbf{R}_{\boldsymbol{\theta}_j}$ and compute the wavelet representations:
               $$R_{\boldsymbol{\theta}_j,k} = \text{Interpolate}\ (\boldsymbol{\theta}_j(t_i), \mathbf{R}, \boldsymbol{\theta}_{j,k}),$$
           where $\boldsymbol{\theta}_{j,k},\ j = 0, \dots, N-1$ are uniformly distributed in the $\boldsymbol{\theta}_j$-coordinate, i.e. $\boldsymbol{\theta}_{j,k} =$
           $2\pi L_{\boldsymbol{\theta}_j} k/N$ and the interpolation is done by a cubic spline. And compute
               $$\widetilde{\mathbf{a}} = \sum_{k=1}^{N} \boldsymbol{\Pi}^T_{\boldsymbol{\theta}_j}(\boldsymbol{\theta}_{j,k}) \cdot \left[(\cos\boldsymbol{\theta}_{j,k}) \star R_{\boldsymbol{\theta}_j,k}\right], \tag{4.7}$$
               $$\widetilde{\mathbf{b}} = \sum_{k=1}^{N} \boldsymbol{\Pi}^T_{\boldsymbol{\theta}_j}(\boldsymbol{\theta}_{j,k}) \cdot \left[(\sin\boldsymbol{\theta}_{j,k}) \star R_{\boldsymbol{\theta}_j,k}\right]. \tag{4.8}$$
           This computation can be accelerated by the fast wavelet transform. Here we add $(\boldsymbol{\theta}_{j,k})$
           after the matrix (or vector) to emphasize it is evaluated over $\boldsymbol{\theta}_{j,k}$.
5:         Apply the shrinkage operator to compute $a_{\boldsymbol{\theta}_j}$ and $b_{\boldsymbol{\theta}_j}$ in the $\boldsymbol{\theta}_j$-coordinate:
               $$a^{n+1}_{\boldsymbol{\theta}_j}(\boldsymbol{\theta}_{j,k}) = \boldsymbol{\Pi}_{\boldsymbol{\theta}_j}(\boldsymbol{\theta}_{j,k}) \cdot \mathcal{S}_{\mu^{-1}}(\widetilde{\mathbf{a}}), \tag{4.9}$$
               $$b^{n+1}_{\boldsymbol{\theta}_j}(\boldsymbol{\theta}_{j,k}) = \boldsymbol{\Pi}_{\boldsymbol{\theta}_j}(\boldsymbol{\theta}_{j,k}) \cdot \mathcal{S}_{\mu^{-1}}(\widetilde{\mathbf{b}}). \tag{4.10}$$
           This step can also be accelerated by the wavelet reconstruction algorithm.
6:         Interpolate $a_{\boldsymbol{\theta}_j}$ and $b_{\boldsymbol{\theta}_j}$ back to the physical grid points $\{t_i\}_{i=1}^N$ by a cubic spline.
7:     end for
8:     Compute
           $$\mathbf{q}^{n+1} = \mu \left[\mathbf{f} - \sum_{j=1}^{M} (\mathbf{a}^{n+1}_{\boldsymbol{\theta}_j} \star \cos\boldsymbol{\theta}_j + \mathbf{b}^{n+1}_{\boldsymbol{\theta}_j} \star \sin\boldsymbol{\theta}_j)\right] + \mathbf{q}^n$$
9: end while
```

$i = 1, \dots, N$, where N is the number of samples and

$$\delta[n] = \begin{cases} 1, & n = 0, \\ 0, & n \neq 0. \end{cases} \tag{5.1}$$

If we enlarge the dictionary to include all the impulses, then the generalized formulation can be used to decompose signals with outliers.

Algorithm 5. (Gauss–Newton type iteration with outliers).

```
Input: Initial guess of phase functions $\boldsymbol{\theta}_j^0$, $j=1,\dots,M$, $\eta=l_0$, where $l_0$ is same as that in (2.1).
Output: Phase functions and the corresponding envelopes: $\boldsymbol{\theta}_j$, $\mathbf{a}_j$, $j=1,\dots,M$.
 1: while $\eta \geq 1$ do
 2:     while $\sum_{j=1}^{M} \|\boldsymbol{\theta}_j^{n+1}-\boldsymbol{\theta}_j^{n}\|_2 > \epsilon_0$ do
 3:         Solve the following $l_1$ optimization problem:
                $\left(\widetilde{\mathbf{a}}^{n+1}, \widetilde{\mathbf{b}}^{n+1}, \mathbf{z}^{n+1}\right) = \mathrm{argmin}_{\mathbf{x},\mathbf{y},\mathbf{z}}(\|\mathbf{x}\|_1 + \|\mathbf{y}\|_1 + \|\mathbf{z}\|_1)$,
                subject to: $\boldsymbol{\Phi}_{\boldsymbol{\theta}_1^n,\dots,\boldsymbol{\theta}_M^n} \cdot \mathbf{x} + \boldsymbol{\Psi}_{\boldsymbol{\theta}_1^n,\dots,\boldsymbol{\theta}_M^n} \cdot \mathbf{y} + \mathbf{z} = \mathbf{f}$.
 4:         Calculate the envelopes in the same way as we did in Step 4 of Algorithm 1.
 5:         Update $\boldsymbol{\theta}_j^n$, $j=1,\dots,M$ in the same way as we did in Step 5 of Algorithm 1.
 6:     end while
 7:     $\eta=\eta-1$.
 8: end while
```

Algorithm 6. (Sweeping ALM with outliers).

```
Input: $\mathbf{p}_j^0=0$, $j=1,\dots,M$, $\mathbf{q}^0=0$, $\mu>0$.
Output: Phase functions and the corresponding envelopes: $\boldsymbol{\theta}_j$, $\mathbf{a}_j$, $j=1,\dots,M$.
 1: while not converge do
 2:     for $j=1:M$ do
 3:         $\mathbf{r}_j^k = \mathbf{f} - \sum_{l=1}^{j-1} \Theta_{\boldsymbol{\theta}_l}\mathbf{p}_l^{k+1} - \sum_{l=j+1}^{M} \Theta_{\boldsymbol{\theta}_l}\mathbf{p}_l^{k} - \mathbf{z}^k$.
 4:         $\mathbf{p}_j^{k+1} = \underset{\mathbf{p}_j}{\mathrm{argmin}}\, \|\mathbf{p}_j\|_1 + \frac{\mu}{2}\|\mathbf{r}_j^k + \mathbf{q}^k/\mu - \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j\|_2^2$.
 5:     end for
 6:     $\mathbf{r}_0^k = \mathbf{f} - \sum_{l=1}^{M} \Theta_{\boldsymbol{\theta}_l}\mathbf{p}_l^{k+1}$.
 7:     $\mathbf{z}^{k+1} = \mathcal{S}_{\mu^{-1}}(\mathbf{r}_0^k + \mathbf{q}^k/\mu)$.
 8:     $\mathbf{q}^{k+1} = \mathbf{q}^k + \mu\left(\mathbf{f} - \sum_{j=1}^{M} \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j^{k+1} - \mathbf{z}^{k+1}\right)$.
 9: end while
```

More specifically, in this case, the optimization problem is formulated in the following way:

$$\min_{\mathbf{x},\boldsymbol{\theta}_1\dots,\boldsymbol{\theta}_M} \|\mathbf{x}\|_1 + \|\mathbf{z}\|_1, \quad \text{subject to: } \boldsymbol{\Phi}_{\boldsymbol{\theta}_1,\dots,\boldsymbol{\theta}_M} \cdot \mathbf{x} + \mathbf{z} = \mathbf{f}, \tag{5.2}$$

where $\boldsymbol{\Phi}_{\boldsymbol{\theta}_1,\dots,\boldsymbol{\theta}_M}$ is given in (2.4).

Using a linearization procedure similar to that of algorithm 1, we obtain algorithm 5 for the above optimization problem (5.2). Moreover, the l_1 optimization problem in the above iterative algorithm can be solved by a sweeping ALM method, algorithm 6. The computation of $\mathbf{p}_j^{k+1} = \mathrm{argmin}_{\mathbf{p}_j} \|\mathbf{p}_j\|_1 + (\mu/2)\|\mathbf{r}_j^k + \mathbf{q}^k/\mu - \Theta_{\boldsymbol{\theta}_j}\mathbf{p}_j\|_2^2$ can be accelerated by algorithm 4 in the previous section.

6. Numerical results

In this section, we present several numerical results to demonstrate the effectiveness of our time-frequency analysis methods. In our previous paper [7], we have performed extensive numerical experiments to demonstrate the effectiveness of our data-driven time-frequency analysis method for signals with good scale separations. To save space, we will not consider the examples with good scale separation in this paper and consider more challenging signals that do not have good scale separation property.

Example 6.1. In the first example, the signal contains two different components whose instantaneous frequencies intersect with each other. More specifically, the signal is generated by the formula below:

$$f = \cos\theta_1(t) + \cos\theta_2(t) + X(t), \quad t \in [0, 1], \tag{6.1}$$

where the phase function θ_1, θ_2 are given as follows:

$$\theta_1(t) = 39.2\pi t - 12\sin 2\pi t \quad \text{and} \quad \theta_2(t) = 85.4\pi t + 12\sin 2\pi t,$$

and $X(t)$ is white noise with zero mean and variance $\sigma^2 = 1$. The signal is sampled over 1024 grid points which are uniformly distributed over the interval $[0, 1]$. The original signal is shown in figure 1.

For this signal, the classical time-frequency analysis methods, such as the windowed Fourier transform, the wavelet transform give a poor result near the intersection where the two instantaneous frequencies cross each other. The EMD method and the data-driven time-frequency analysis method introduced in our previous paper [7] also have problem near the intersection.

The result given by the data-driven time-frequency analysis method proposed in this paper is shown in figure 1. In the computation, the initial guesses of the instantaneous frequencies are chosen to be $128\pi t$ and $32\pi t$, respectively, which are far from the ground truth. As we can see, even with these rough initial guesses, our iterative algorithm still can recover the instantaneous frequencies and the corresponding IMFs with reasonable accuracy, although the signal is polluted by noise. We note that the end-effect is more pronounced in this case due to the noise pollution.

Example 6.2. The next signal that we consider is polluted by outliers. We generate the signal by using the following formula:

$$f = \cos\theta_1(t) + \cos\theta_2(t) + \sigma(t), \tag{6.2}$$

where $\theta_1(t)$ and $\theta_2(t)$ are the same as in example 6.1. The signal is sampled over 1024 uniform grid points. Among these samples, there are 32 samples that are outliers. The locations of these outliers are selected randomly and the strengths satisfy the normal distribution. In figure 2, we present the results for the case without noise ($\sigma(t) = 0$) by using the algorithm given in §5. As we can see, both the IMFs and the outliers are captured very accurately.

We also test the signal with outliers and noise. The results are shown in figure 3. In this example, the noise and the outliers are added to the original signal together. The signal is sampled over 1024 uniform grid points. Among these samples, there are 32 samples that are outliers. The locations of these outliers are selected randomly and the strengths satisfy the normal distribution whose standard deviation is 1. The noise is Gaussian noise and the standard deviation is 0.1.

In this case, the instantaneous frequencies and the IMFs are still quite accurate. But the outliers are not captured as well as in the case with no noise. We would like to emphasize that it would be hard to distinguish the outliers from the noise when the amplitude of the outliers is small. For the outliers whose amplitude is large, they can be separated from the noise by a proper shrinkage operator. However, the shrinkage operator also kills the outliers whose amplitude is comparable to or smaller than the noise level.

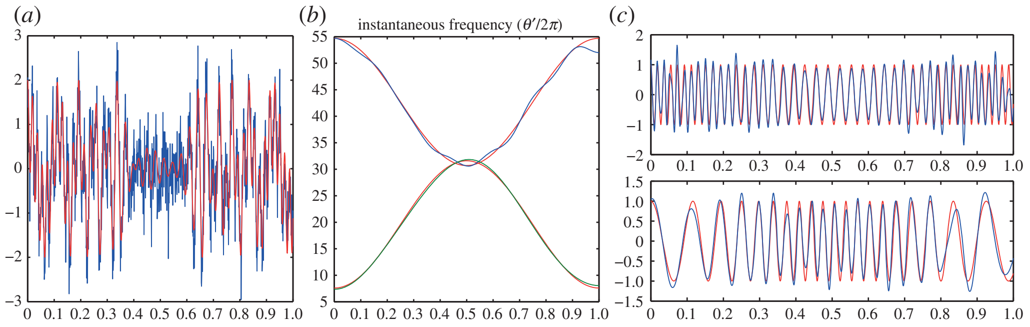

Figure 1. (*a*) Original signal in example 6.1, red curve is the clean signal without noise; (*b*) instantaneous frequencies; red: exact, blue: numerical; (*c*) corresponding IMFs, red: exact, blue: numerical. (Online version in colour.)

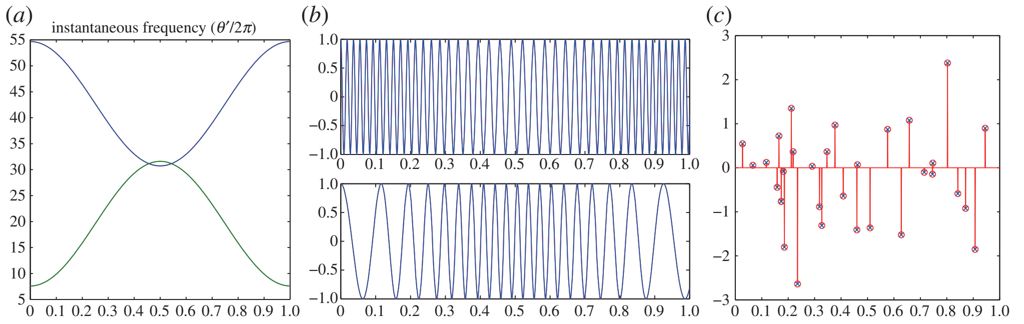

Figure 2. (*a*) Instantaneous frequencies; red: exact, blue: numerical; (*b*) corresponding IMFs; red: exact, blue: numerical; (*c*) outliers; red circle: exact, blue cross: numerical. (Online version in colour.)

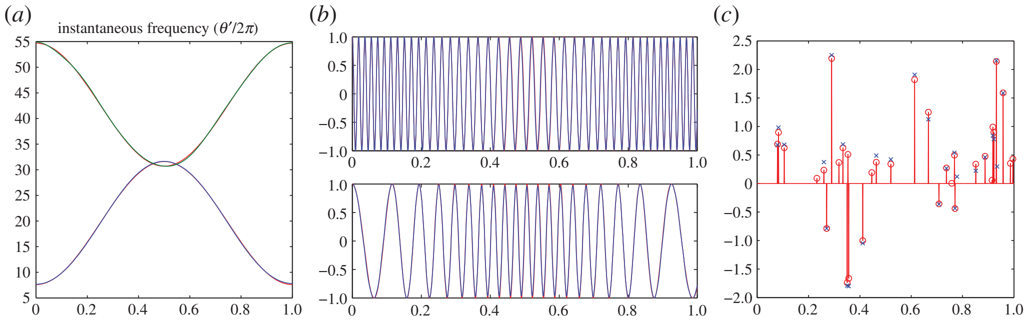

Figure 3. (*a*) Instantaneous frequencies; red: exact, blue: numerical; (*b*) corresponding IMFs; red: exact, blue: numerical; (*c*) outliers; red circle: exact, blue cross: numerical. (Online version in colour.)

Example 6.3. In our third example, we consider a real signal, a bat chirp signal. It is the digitized echolocation pulse emitted by the Large Brown Bat, Eptesicus Fuscus.[1] The signal includes 400 samples and the sampling period is 7 µs, so the total time span is 2.8 ms.

[1]The authors wish to thank Curtis Condon, Ken White and Al Feng of the Beckman Institute of the University of Illinois for the bat data and for permission to use it in this paper.

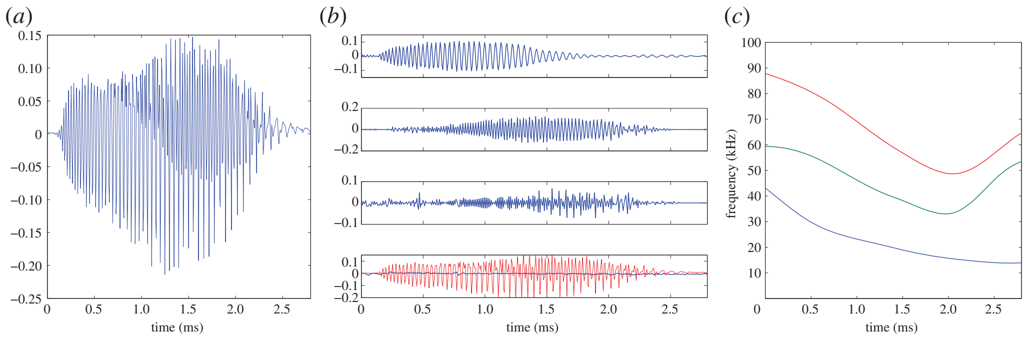

Figure 4. (*a*) Bat chirp signal; (*b*) IMFs; (*c*) instantaneous frequencies. (Online version in colour.)

The signal is shown in figure 4*a*. The IMFs and instantaneous frequencies obtained by our method are given in figure 4*b*,*c*. Our method could give precise instantaneous frequencies and also the sparse decomposition of the original signal. From figure 4, we can see that the signal is approximated very well by only three IMFs. Near the boundaries, the original signal and the IMFs are all very small. In these regions, the frequencies actually do not have any physical meaning. They are only auxiliary variables in the algorithm. In the region in which the amplitude of the signal is order one, the recovered instantaneous frequencies reveal some interesting patterns that have not been seen before using traditional time-frequency methods. The physical significance of these patterns need to be further investigated in the future.

Example 6.4. In the last example, we consider the data from an ODE system. We consider a multiple degree of freedom system.

$$\ddot{\mathbf{u}} + K(t)\mathbf{u} = 0, \tag{6.3}$$

where K is an $N \times N$ symmetric positive definite stiffness matrix and $\mathbf{u}$ is an $N \times 1$ vector, which typically represents the displacement of certain engineering structure. This kind of ODE system is widely used to model the movements of structures [19], such as buildings, bridges, etc. In many applications, we want to recover the stiffness matrix K (at least part of it) from incomplete measurement of the solution $\mathbf{u}$.

Here, we assume that $K(t)$ is slowly varying. Under this assumption, the solutions, $u_j(t), j = 1, \ldots, N$, have the following approximate expression:

$$u_j(t) = \sum_{k=1}^{N} \rho_{j,k}\, \mathrm{e}^{\mathrm{i}\boldsymbol{\theta}_k(t)}, \tag{6.4}$$

and $(\boldsymbol{\theta}'_k(t))^2, k = 1, \ldots, N$ are eigenvalues of $K(t)$. Using this formulation, we can see that the instantaneous frequencies could help us to retrieve the stiffness matrix.

As a test example, we consider a simple case where the degree of freedom is given by $N = 2$. The K matrix has the following form:

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 \\ -k_2 & k_2 + k_3 \end{bmatrix}, \tag{6.5}$$

$$k_1 = k_3 = 100\cos(0.2\pi t) + 500 \quad \text{and} \quad k_2 = 400\cos(0.2\pi t) + 400.$$

The initial conditions are

$$u_1(0) = 1, \quad u'_1(0) = 0, \quad u_2(0) = 2 \quad \text{and} \quad u'_2(0) = 0.$$

This system models the movement of two objects of equal masses connected by springs. And k_1, k_2, k_3 are stiffness values of springs.

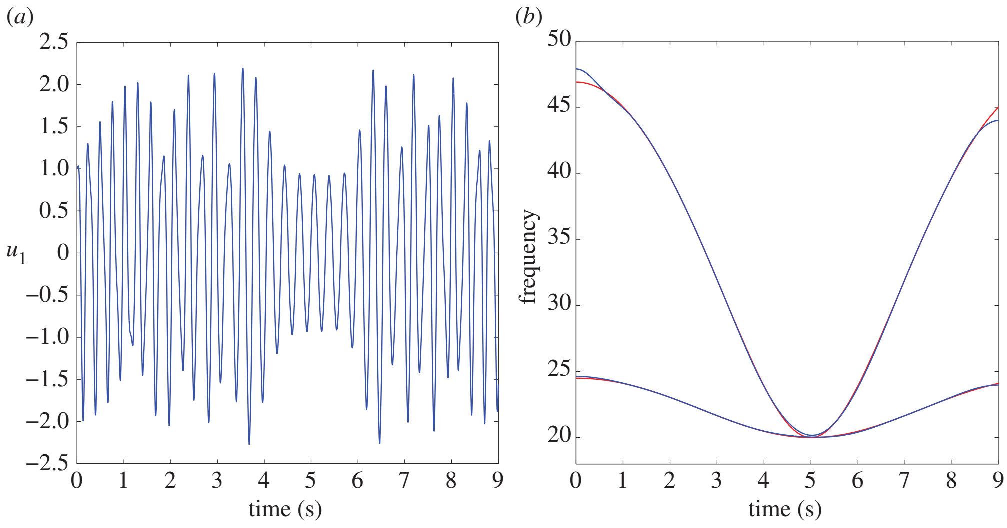

Figure 5. (*a*) Solution u_1 of the ODE system (6.3); (*b*) instantaneous frequencies; red: theoretical frequencies; blue: numerical results. (Online version in colour.)

The above system is solved up to $t = 9$. The initial guesses of the phase functions are $20t$ and $40t$. Here, we only analyse the first component of the solution $u_1(t)$, which is shown in the figure 5*a*. According to (6.4), the theoretical frequencies are $\sqrt{(k_1 + k_3)/2}$ and $\sqrt{(k_1 + k_3 + 4k_2)/2}$. In figure 5*c*, we compare the theoretical frequencies and the numerical results given by our method. As we can see, they match very well even near the intersection point. This toy example shows that our method indeed has the capability to retrieve some information of the physical process hidden within the signal. Now we are trying to apply our method to analyse the signals from real bridges.

7. Concluding remarks

In this paper, we have introduced a novel formulation to obtain sparse time-frequency decomposition and the corresponding instantaneous frequencies. We formulated the decomposition as a dictionary adaptation problem. The dictionary is parametrized by phase functions and the phase functions are determined by the signal itself. Based on our previous work and the methods of dictionary learning, we developed an iterative algorithm to determine the phase functions and the corresponding decomposition. By designing the dictionary carefully to make them orthogonal in the coordinate of the phase functions, we can accelerate the algorithm by using the fast wavelet transform. This makes our algorithm very efficient.

Another advantage of this method is that it can be easily generalized to deal with more complicated data that are not sparse over the time-frequency dictionary, such as data with outliers. For this kind of signal, we just need to enlarge the dictionary and follow a similar procedure to look for the sparsest decomposition over this enlarged dictionary. We presented several numerical examples to demonstrate the effectiveness of our method, including data that do not have scale separation and data that are polluted by noise and/or outliers. The results that we obtained seem to suggest that our method can offer an effective way to decompose multiscale data even with a poor scale separation property.

We remark that decomposing several IMFs simultaneously increases the complexity of the optimization problem. As a result, the robustness of this new method is not as good as the previous one that we introduced in [7]. Generally speaking, the more IMFs we try to decompose simultaneously, the less robust the method becomes. In the numerical tests, we found that a naive application of the method is not very stable if $M > 3$. This difficulty can be alleviated by knowing some information about the signal. To make our method more robust, we are now considering combining our method with other time-frequency analysis methods, such as synchrosqueezed wavelet transform [3], in our future work.

Another interesting problem that we are considering is to decompose data with intra-wave frequency modulation. This type of data is known to be very challenging. Naive application of traditional data analysis methods tends to introduce artificial harmonics. To deal with this kind of data, we have to extend the definition of the dictionary by replacing the cosine function by an unknown periodic shape function that we need to find as part of the optimization problem. This work will be reported in our future paper.

Data accessibility. Data in examples 6.1, 6.2 and 6.4 are synthetic data. They can be reproduced following the descriptions in the paper. Data used in example 6.3 can be downloaded at http://www.ece.rice.edu/dsp/software/bat.shtml.

Authors' contributions. Both authors, T.Y.H. and Z.S., worked together to propose the proper formulation for this problem. Z.S. carried out most of the work in the design and the implementation of the numerical algorithms. He also drafted the manuscript. T.Y.H. revised the manuscript and made sure that we have done a thorough study to demonstrate the accuracy and robustness of the proposed method. Both authors gave final approval for publication.

Competing interests. We declare we have no competing interests.

Funding. This research was supported in part by NSF grants nos. DMS-318377 and DMS-1159138, DOE grant no. DE-FG02-06ER25727, and AFOSR MURI grant no. FA9550-09-1-0613. The research of Z.S. was supported by a NSFC grant no. 11201257.

References

1. Huang NE, Shen Z, Long SR, Wu MLC, Shih HH, Zheng Q, Yen N-C, Tung CC, Liu HH. 1998 The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proc. R. Soc. Lond. A* **454**, 903–995. (doi:10.1098/rspa.1998.0193)
2. Wu Z, Huang NE. 2009 Ensemble empirical mode decomposition: a noise-assisted data analysis method. *Adv. Adapt. Data Anal.* **1**, 1–41. (doi:10.1142/S1793536909000047)
3. Daubechies I, Lu J, Wu H. 2011 Synchrosqueezed wavelet transforms: an empirical mode decomposition-like tool. *Appl. Comp. Harmon. Anal.* **30**, 243–261. (doi:10.1016/j.acha.2010.08.002)
4. Dragomiretskiy K, Zosso D. 2013 Varitational mode decomposition. *IEEE Trans. Signal Process.* **62**, 531–544. (doi:10.1109/TSP.2013.2288675)
5. Gilles J. 2013 Empirical wavelet transform. *IEEE Trans. Signal Process.* **61**, 3999–4010. (doi:10.1109/TSP.2013.2265222)
6. Hou TY, Shi Z. 2011 Adaptive data analysis via sparse time-frequency representation. *Adv. Adapt. Data Anal.* **3**, 1–28. (doi:10.1142/S1793536911000647)
7. Hou TY, Shi Z. 2013 Data-driven time-frequency analysis. *Appl. Comput. Harmon. Anal.* **35**, 284–308. (doi:10.1016/j.acha.2012.10.001)
8. Hou TY, Shi Z. 2013 Sparse time-frequency representation of nonlinear and nonstationary data. *Sci. China Math.* **56**, 2489–2506. (doi:10.1007/s11425-013-4733-7)
9. Hou TY, Shi Z, Tavallali P. 2014 Convergence of a data-driven time-frequency analysis method. *Appl. Comput. Harmon. Anal.* **37**, 235–270. (doi:10.1016/j.acha.2013.12.004)
10. Aharon M, Elad M, Bruckstein AM. 2006 The K-SVD: an algorithm for designing of overcomplete dictionaries for sparse representations. *IEEE Trans. Signal Process.* **54**, 4311–4322. (doi:10.1109/TSP.2006.881199)
11. Engan K, Skretting K, Husøy J. 2007 Family of iterative LS-based dictionary learning algorithms, ILS-DLA, for sparse signal representation. *Digit. Signal Process.* **17**, 32–49. (doi:10.1016/j.dsp.2006.02.002)
12. Lewicki M, Sejnowski T. 2000 Learning overcomplete representations. *Neural Comput.* **12**, 337–365. (doi:10.1162/089976600300015826)
13. Mairal J, Bach F, Ponce J, Sapiro G. 2009 Online dictionary learning for sparse coding. In *Proc. 26th Int. Conf. on Machine Learning* (*ICML*), *Montreal, Canada, 14–18 June*. Princeton, NJ: IMLS.

14. Skretting K, Engan K. 2010 Recursive least squares dictionary learning algorithm. *IEEE Trans. Signal Process.* **58**, 2121–2131. (doi:10.1109/TSP.2010.2040671)
15. Bertsekas DP. 1982 *Constrained optimization and lagrange multiplier method*. New York, NY: Academic Press.
16. Goldstein T, Osher S. 2009 The split Bregman method for L_1-regularized problems. *SIAM J. Imaging Sci.* **2**, 323–343. (doi:10.1137/080725891)
17. Micchelli CA, Shen L, Xu Y. 2011 Proximity algorithms for image models: denoising. *Inverse Probl.* **27**, 045009. (doi:10.1088/0266-5611/27/4/045009)
18. Combettes PL, Pesquet J-C. 2011 Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering* (eds HH Bauschke, R Burachik, PL Combettes, V Elser, DR Luke, H Wolkowicz), pp. 185–212. New York, NY: Springer.
19. Chopra AK. 1995 *Dynamics of structures: theory and applications to earthquake engineering*. Englewood Cliffs, NJ: Prentice-Hall.