Appl. Comput. Harmon. Anal. 35 (2013) 350–358

Contents lists available at SciVerse ScienceDirect

Applied and Computational Harmonic Analysis

www.elsevier.com/locate/acha

Letter to the Editor

Fast and memory-efficient algorithms for computing quadratic time–frequency distributions

J.M. O' Toole [a,*], B. Boashash [a,b]

[a] The University of Queensland, Centre for Clinical Research and Perinatal Research Centre, Royal Brisbane & Women's Hospital, Herston, QLD 4029, Australia
[b] College of Engineering, Qatar University, Doha, Qatar

ARTICLE INFO

Article history:
Received 3 April 2012
Revised 10 December 2012
Accepted 25 January 2013
Available online 4 February 2013
Communicated by Naoki Saito

Keywords:
Algorithms
Computational efficiency
Fast Fourier transform
Nonstationary signals
Time–frequency analysis
Time–frequency distributions
Wigner–Ville distribution

ABSTRACT

Algorithms for computing time–frequency distributions (TFDs) limit computation time by reducing numerical operations. But these *fast* algorithms do not reduce the memory load. This article presents four TFD algorithms to minimise both the computation and memory loads. Each algorithm is optimised for a specific kernel category. Three algorithms reduce memory by computing an exact TFD without oversampling; the fourth algorithm, for the nonseparable kernel, reduces memory by computing a decimated TFD. The separable-kernel algorithm, using a biomedical signal as an example, computes an exact TFD with only 12% of the computation load and 1% of the memory required by conventional algorithms.

1. Introduction

Most signals have spectra that changes over time. Time–frequency (t–f) domain methods, such as time–frequency distributions (TFDs), capture this *nonstationary* information by representing signals in both time and frequency simultaneously [1]. Applications of t–f methods are diverse and often use large data sets with long signals.

Yet TFD algorithms require significant computational resources. Typical algorithms use approximately $N^2 \log_2 N$ operations and N^2 sample points of memory [2]. Hence computing a TFD is problematic for a computer which cannot store N^2 sample points in memory [3]. Motivated by this memory problem, we present TFD algorithms that reduce both computation and memory loads.

An established and popular t–f representation is the class of quadratic TFDs [1]. Here, we present algorithms for this quadratic class only. There are two types of (quadratic) TFD algorithms. One approach computes an exact TFD and minimises computation load by taking advantage of the conjugate symmetry of the time-lag function [2]. (The time-lag function is the inverse Fourier transform of the TFD, where frequency is transformed to lag.) The other approach computes an approximation to the TFD [4–6]. The accuracy of the approximation is controlled by a parameter as decreasing computation load decreases accuracy. This second approach, unlike the first approach, can reduce the algorithm's memory load by computing a TFD decimated in the frequency direction [4,6]. The term *decimate* in this context describes the process of computing only a subset of the sample points for the TFD.

* Corresponding author. Now at: DeustoTech, University of Deusto, Bilbao 48007, Spain.
E-mail addresses: j.otoole@ieee.org (J.M. O' Toole), boualem@qu.edu.qa (B. Boashash).

1063-5203/$ – see front matter
http://dx.doi.org/10.1016/j.acha.2013.01.003

We extend these two existing approaches to compute an exact TFD with minimal computation and memory load. The first step was to extend the methods from the exact-TFD approach [2] to a new [7] discrete TFD definition [8]. We used this new definition because, unlike other definitions [2,9], the discrete TFD is alias free and satisfies all important mathematical properties [7], such as the frequency marginal. This discrete TFD, however, has a nonuniform sampling grid in the time-lag domain and thus requires a more complicated procedure to reduce computational load [7].

Here we present algorithms for four common kernel categories: the nonseparable, separable, lag-independent, and Doppler-independent kernel categories. Algorithms for three categories (separable, lag- and Doppler-independent kernels) have, depending on kernel parameters, computation loads that are less than or equal to the general nonseparable kernel form. Even better for these three kernel categories, however, is that the algorithm can reduce the size of the TFD array by not oversampling, thus reducing computation and memory loads. The algorithm for the nonseparable kernel computes the decimated TFD to reduce memory load.

2. Background: the Wigner–Ville distribution

Before forming the TFD, the real-valued $s(t)$ is first transformed to the analytic signal $z(t)$ [1]. The complex-valued $z(t)$ avoids cross-terms (artefacts) between positive and negative frequencies in the t–f domain. The quadratic class of TFDs, for $z(t)$, is written as

$$\rho_z(t,f) = W_z(t,f) \underset{t}{*}\underset{f}{*} \gamma(t,f) \tag{1}$$

where $*_t*_f$ represents t–f convolution, $W_z(t,f)$ represents the Wigner–Ville distribution (WVD), and $\gamma(t,f)$ is known as the t–f kernel [1]. The WVD is expressed as

$$W_z(t,f) = \int_{-\infty}^{\infty} z\left(t+\frac{\tau}{2}\right) z^*\left(t-\frac{\tau}{2}\right) \mathrm{e}^{-\mathrm{j}2\pi\tau f}\,\mathrm{d}\tau$$

where $z^*(t)$ is the complex conjugate of $z(t)$.

There is more than one way to define a discrete WVD (DWVD) [10–12]. The DWVD definition in [12] is selected as it satisfies all important mathematical properties, such as Moyal's formula and the frequency marginal, and can be computed more efficiently than other definitions [11,12]. It is defined as [12]

$$W[2n,k] = \sum_{m=0}^{N-1} K[2n,m]\mathrm{e}^{-\mathrm{j}2\pi mk/N}, \tag{2}$$

$$W[2n+1,k] = \mathrm{e}^{-\mathrm{j}\pi k/N} \sum_{m=0}^{N-1} K[2n+1,m]\mathrm{e}^{-\mathrm{j}2\pi mk/N} \tag{3}$$

for $0 \leqslant n,k \leqslant N-1$, thus the $W[n,k]$ array is of size $2N \times N$. The time-lag function $K[n,m]$ is

$$\begin{aligned} K[2n,m] &= z[n+m]z^*[n-m], \\ K[2n+1,m] &= z[n+m+1]z^*[n-m]. \end{aligned} \tag{4}$$

The complex-valued $z[n]$, of length $2N$, is the analytic associate of the real-valued signal $s[n]$ of length N [13]. Although $z[n]$ is length $2N$, $z[n] = 0$ for $N \leqslant n \leqslant 2N-1$ [13]. The notation for the time index n in the previous expressions (2)–(4) represents even–odd values of n: that is, for $W[p,k]$, even p values are $p = 2n$ and odd p values are $p = 2n+1$.

Algorithms to compute the DWVD therefore must first form the time-lag function $K[n,m]$ and second discrete Fourier transform (DFT) the time-lag function to the t–f domain. The DFT, using a fast Fourier transform (FFT) algorithm, accounts for most of the computational load of the algorithm. The time-lag function $K[2n,m]$ in (4) is conjugate symmetrical in lag (m). Algorithms take advantage of this conjugate symmetry property to reduce the computation load from $N \times$ FFT-N operations to just $N/2 \times$ FFT-N operations [10] (the notation FFT-N represents one FFT operation on a length-N signal).

But $K[2n+1,m]$, unlike $K[2n,m]$, is not conjugate symmetric and would require $N \times$ FFT-N operations. The following procedure reduces this computation load [12]. Because $W[2n+1,k]$ in (3) is real-valued, we can rewrite as

$$W[2n+1,k] = \Im\Big(\underset{m\to k}{\mathrm{DFT}}\big\{K[2n+1,m]\big\}\Big)\csc(\pi k/N) \tag{5}$$

for $k = 1,\ldots,N-1$, where $\Im(\cdot)$ is the imaginary part of a complex number and $\csc(\cdot) = 1/\sin(\cdot)$ is the cosecant function. The DFT$\{\cdot\}$ function, with $m \to k$, is the DFT from lag (m) to frequency (k).

Next, define the function $\hat{K}[n,m]$ so that

$$\underset{m\to k}{\mathrm{DFT}}\big\{\hat{K}[n,m]\big\} = \Im\Big(\underset{m\to k}{\mathrm{DFT}}\big\{K[2n+1,m]\big\}\Big) \tag{6}$$

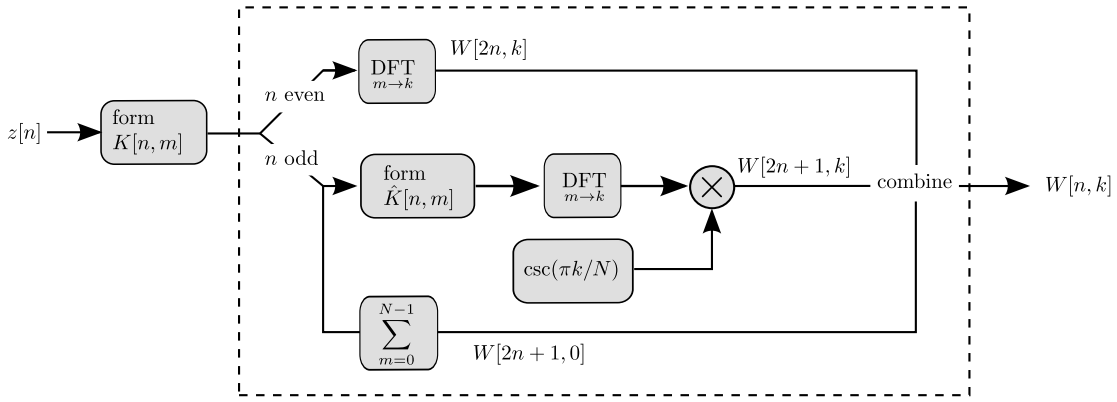

Fig. 1. Outline of Wigner–Ville distribution algorithm for analytic signal $z[n]$ of length $2N$. The process within the dotted-line box transforms the time-lag function $K[n,m]$ to the time–frequency domain $W[n,k]$.

and then substitute this expression into (5). Because $\hat{K}[n,m]$ is conjugate symmetric, this procedure costs $N/2 \times$ FFT-N operations [12]. When $k=0$, (5) is undefined, and we simply sum along the lag values:

$$W[2n+1,0] = \sum_{m=0}^{N-1} K[2n+1,m].$$

Fig. 1 outlines the algorithm. The algorithm uses $N \times$ FFT-N operations to compute the $2N \times N$ DWVD.

3. Kernel-specific algorithms

The discrete version of (1) is

$$\rho[n,k] = W[n,k] \underset{n}{\circledast} \underset{k}{\circledast} \gamma[n,k] \tag{7}$$

where $\rho[n,k]$ is of size $2N \times 2N$ [7]. But as $\rho[n,k]$ is periodic in N in frequency we need only compute the $2N \times N$ discrete TFD (DTFD). An algorithm computes the DTFD using $(3N+1) \times$ FFT-N operations [8]. The algorithm minimises computation by (i) forming the windowed time-lag function $R[n,m]$ for positive values of m only, an approach proposed for the so-called generalised DTFD (GDTFD) definition [2]; and (ii) by using the procedure in Fig. 1 to transform from the time-lag to the t–f domain.

This algorithm assumes the Doppler-lag kernel $g[l,m]$ is nonseparable. (The Doppler-lag kernel is the DFT of the t–f kernel $\gamma[n,k]$.) There are, however, other kernel categories: the separable kernel $g[l,m] = G_1[l]g_2[m]$; the Doppler-independent kernel $g[l,m] = g_2[m]$; and the lag-independent kernel $g[l,m] = G_1[l]$ [1]. The nonseparable kernel is the general form, encompassing all types.

Depending on the length of $G_1[l]$ and $g_2[m]$, and signal length (N), we can decrease the sampling rate of the DTFD in either t–f direction or both directions simultaneously. Reducing the sampling rate reduces the algorithm's computation and memory load. Note that these algorithms produce an exact DTFD which upholds the mathematical properties of their over-sampled counterparts. (See Appendix A for proofs of some properties.)

3.1. Separable kernel

This algorithm computes the $N_{\text{time}} \times N_{\text{freq}}$ DTFD using the kernel $g[l,m] = G_1[l]g_2[m]$. The parameters N_{time} and N_{freq} are set by the user to control the overall dimensions of the DTFD array and relate to the size of the kernel functions. The Doppler function $G_1[l]$ is length Q, with $Q \leqslant N$. The lag function $g_2[m]$ is length P, with $P \leqslant 2N$. The parameter N_{time} sets the size of the DTFD in the time direction; N_{time} is even and $2Q \leqslant N_{\text{time}} \leqslant 2N$. The parameter N_{freq} sets the length of the DTFD in the frequency direction; $\lceil (P+1)/2 \rceil \leqslant N_{\text{freq}} \leqslant N$ when $P < 2N$ or $N_{\text{freq}} = N$ when $P = 2N$.

When $N_{\text{time}} > 2Q$, or when $N_{\text{freq}} > \lceil (P+1)/2 \rceil$, the DTFD is oversampled. No information is lost when $N_{\text{freq}} < N$ because the length-P function $g_2[m]$ will zero-pad part of the discrete time-lag function, as $P < 2N$. The same is true when $N_{\text{time}} < 2N$.

The direct procedure to form the $N_{\text{time}} \times N_{\text{freq}}$ DTFD is as follows:

1. Form the time-lag function $K[n,m]g_2[m]$ of size $N \times 2N_{\text{freq}}$;
2. DFT to the Doppler-lag domain to get $A[l,m]$;
3. modulate for odd m values: $A[l,2m+1] = A[l,2m+1]\exp(\mathrm{j}\pi l/N)$;
4. window in the Doppler direction: $S[l,m] = A[l,m]G_1[l]$;

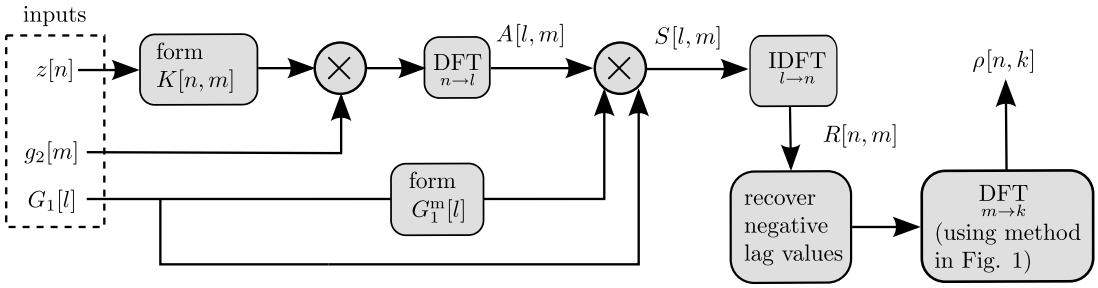

Fig. 2. Separable-kernel algorithm for length-$2N$ analytic signal $z[n]$, length-P lag-function $g_2[m]$, and length-Q Doppler-function $G_1[l]$.

5. resize the array $S[l,m]$ from $N\times 2N_{\mathrm{freq}}$ to $N_{\mathrm{th}}\times 2N_{\mathrm{freq}}$; we can do this because of the windowing of the Doppler-lag function by the length-Q $G_1[l]$ in the previous step, where $N_{\mathrm{th}}=N_{\mathrm{time}}/2$ and $N_{\mathrm{th}}\geqslant Q$;
6. modulate for odd m values: $S[l,2m+1]=S[l,2m+1]\exp(-\mathrm{j}\pi l/N_{\mathrm{th}})$;
7. DFT back to the time-lag domain to get $R[n,m]$;
8. and finally, DFT $R[n,m]$ to the t–f domain to get $\rho[n,k]$.

But this procedure can require significant memory to compute when N is large, as the functions $K[n,m]$, $A[l,m]$, $S[l,m]$, and $R[n,m]$ are of size $N\times 2N_{\mathrm{freq}}$. With some careful manipulation, the proposed algorithm can form one-dimensional arrays, and not the two-dimensional $N\times 2N_{\mathrm{freq}}$ arrays to compute the DTFD. Also, the proposed algorithm uses the procedure in Fig. 1 to help reduce the computational load.

The modulation terms in steps 3 and 6 correct for the nonuniform discrete grid of the time-lag functions [12]. When $N_{\mathrm{time}}=2N$ and thus $N_{\mathrm{th}}=N$, the modulation terms in step 3 and 6 cancel and are therefore not needed, which is why they are not present in the nonseparable-kernel DTFD algorithm [8]. When $N_{\mathrm{th}}<N$, we do need to modulate these terms. We can add these modulation terms in the Doppler function $G_1[l]$ so we do not need to modulate and demodulate for each odd m value, thus saving some computation.

Fig. 2 outlines the following algorithm. (We omit some details relating to indexing of the arrays to better present the concepts of algorithms; full details are in Ref. [12].)

- INPUT: $z[n]$, $g_2[m]$, $G_1[l]$, N_{time}, and N_{freq}.
- OUTPUT: $N_{\mathrm{time}}\times N_{\mathrm{freq}}$ DTFD array $\rho[n,k]$.

1. Include modulation term in G_1. Let ${G_1}^{\mathrm{mod}}[l]=G_1[l]\exp(-\mathrm{j}\pi l[1/N-1/N_{\mathrm{th}}])$.
2. For even values of m: iterate m_0 over positive lag values only:
   (a) For $0\leqslant n\leqslant N-1$, let: $K_{m0}[n]=z[n+m_0]z^*[n-m_0]g_2[2m_0]$;
   (b) DFT to the Doppler-lag domain:
   $$A_{m0}[l]=\underset{n\to l}{\mathrm{DFT}}\big\{K_{m0}[n]\big\};$$
   (c) window in the Doppler-lag domain: $S_{m0}[l]=A_{m0}[l]G_1[l]$;
   (d) resize $S_{m0}[l]$ from length N to N_{th};
   (e) IDFT back to the time-lag domain: $R[n,2m_0]=\mathrm{IDFT}_{l\to n}\{S_{m0}[l]\}$.
3. For odd values of m: iterate m_0 positive lag values only:
   (a) For $0\leqslant n\leqslant N-1$, let: $K'_{m0}[n]=z[n+m_0+1]z^*[n-m_0]g_2[2m_0+1]$;
   (b) Similar to procedure in step 2(b)–(e):
      i. $A'_{m0}[l]=\mathrm{DFT}_{n\to l}\{K'_{m0}[n]\}$,
      ii. $S'_{m0}[l]=A'_{m0}[l]{G_1}^{\mathrm{mod}}[l]$,
      iii. resize $S'_{m0}[l]$ to length N_{th},
      iv. and $R[n,2m_0+1]=\mathrm{IDFT}_{l\to n}\{S'_{m0}[l]\}$.
4. Recover the negative-lag values from the positive ones to form $R[n,m]$ of size $N_{\mathrm{th}}\times 2N_{\mathrm{freq}}$.
5. Transform to t–f domain, using a modified version of the method outlined in Fig. 1. Start by splitting n into even and odd values:
   (a) For $2n$ values: $\rho[2n,k]=\mathrm{DFT}_{m\to k}\{R[n,2m]\}$;
   (b) For $2n+1$ values,
      i. form $\hat{R}$, to satisfy relation (6), with $N_{\mathrm{fh}}=\lceil N_{\mathrm{freq}}/2\rceil$ as follows
      $$\hat{R}[n,0]=\Im\big(R[n,0]\big),$$
      $$\hat{R}[n,m]=\frac{1}{2\mathrm{j}}\big\{R[n,2m+1]-\big(R[n,2N_{\mathrm{freq}}-2m-1]\big)^*\big\},\quad 1\leqslant m\leqslant N_{\mathrm{fh}},$$

$$\hat{R}[n,m] = \big(\hat{R}[n, 2N_{\text{freq}} - 2m - 1]\big)^*, \qquad N_{\text{fh}} + 1 \leqslant m \leqslant N_{\text{freq}} - 1.$$

ii. DFT to the t–f domain,

$$\rho[2n+1,k] = \underset{m\to k}{\text{DFT}}\big\{\hat{R}[n,m]\big\}\csc(\pi k/N_{\text{freq}}).$$

iii. Do for frequency sample $k=0$: $\rho[2n+1,0] = \sum_{m=0}^{N_{\text{freq}}-1} R[n,2m+1]$.

The algorithm uses $(P_{\text{h}}+1)\times$ FFT-N plus $(P_{\text{h}}+1)\times$ FFT-N_{th} plus $(N_{\text{time}}/2)\times$ FFT-N_{freq} operations to compute the $N_{\text{time}}\times N_{\text{freq}}$ DTFD, with $P_{\text{h}} = \lfloor P/2\rfloor$ and $N_{\text{th}} = N_{\text{time}}/2$. When all parameters values are at their maximum (for $P = 2N$, $N_{\text{freq}} = N$, and $N_{\text{th}} = N$) the computation load for this algorithm is $(3N+2)\times$ FFT-N operations, which approximately equals the computation load for the nonseparable kernel [8].

3.2. Lag- or Doppler-independent kernels

The Doppler-independent algorithm computes the $2N\times N_{\text{freq}}$ DTFD with the kernel $g[l,m] = g_2[m]$. The lag-independent algorithm computes the $N_{\text{time}}\times N$ DTFD with the kernel $g[l,m] = G_1[l]$. Because these algorithms are special cases of the previous separable-kernel and the algorithms have considerable overlap [12], we omit the details here.

3.3. Decimated nonseparable kernel

The nonseparable kernel algorithm requires an array of size $2N\times N$, in contrast to the separable kernel array of size $N_{\text{time}}\times N_{\text{freq}}$. To reduce the memory load, the following algorithm for the nonseparable kernel computes the *decimated* DTFD of size $2N/a\times N/b$, where a and b are the integer decimation factors.

Appropriate folding of a signal in time decimates the signal in frequency and can be used to reduce computation load [14]. To compute the length-N spectral signal $X[k]$ from the time-domain signal $x[n]$, at a intervals only, fold $x[n]$ as follows:

$$x_{\text{fold}}[n] = \sum_{p=0}^{a-1} x[pL+n], \quad 0\leqslant n\leqslant L-1$$

and then take the DFT of this length-L signal, $X[ak] = \text{DFT}_{n\to k}\{x_{\text{fold}}[n]\}$. Thus $X[ak]$ requires only FFT-L operations, and not FFT-N needed to compute $X[k]$.

We apply this approach to computing the decimated DTFD, of the form $\rho[an,bk]$. The parameters a,b are the positive integer decimation factors which produces the $2N/a\times N/b$ DTFD array. (Both $2N/a$ and N/b must also be integers.) The proposed algorithm folds the Doppler-lag function after multiplication with the $N\times 2N$ kernel. Yet the algorithm uses only a $2N/a\times N/b$ array by folding one lag slice of the windowed Doppler-lag function at a time and then iterating over all lag values.

When a is odd, the time-lag function $R[n,2m+1]$ will have a time offset of $(a-1)/2$ because $R[n,2m]$ comes, in time, before $R[n,2m+1]$ [12]. Thus, the temporal order for the array $R[n,m]$ is $R[an,2m]$ followed by $R[an+(a-1)/2,2m+1]$. To enable the decimation with a nonzero offset for $R[an+(a-1)/2,2m+1]$, the algorithm uses a modulation of the folded function with a complex exponential.

- INPUT: $z[n]$, $g[l,m]$, a, and b.
- OUTPUT: $2N/a\times N/b$ DTFD $\rho[an,bk]$.

1. Separate time samples $\{n_i\} = 0,a,2a,\ldots,(2N/a-1)a$ into two sets: one for even values of n_i, $\{n_{ei}\}$ for $0\leqslant i\leqslant L_{\text{e}}-1$; and one set for odd values of n_i, $\{n_{oi}\}$ for $0\leqslant i\leqslant L_{\text{o}}-1$, where $L_{\text{e}}+L_{\text{o}} = 2N/a$. If a is even, then $L_{\text{o}} = 0$. Also, let $J = N/b$ and $J_{\text{h}} = \lceil J/2\rceil$.
2. Iterate the following over m_0 for positive lag values only:
   (a) Fold Doppler-lag function in lag, for $0\leqslant l\leqslant N-1$:

$$S_{m0}[l] = \sum_{p=0}^{b-1} A[l,\, pJ+m_0]\, g\big[l, 2(pJ+m_0)\big],$$

for $A[l,m] = \text{DFT}_{n\to l}\{z[n+m]z^*[n-m]\}$ and

$$S'_{m0}[l] = \sum_{p=0}^{b-1} A'[l,\, pJ+m_0]\, g\big[l, 2(pJ+m_0)+1\big],$$

for $A'[l,m] = \text{DFT}_{n\to l}\{z[n+m+1]z^*[n-m]\}$. If a is even, only compute $S_{m0}[l]$.

(b) Fold in the Doppler direction:

$$S_{m0}[l] = \sum_{q=0}^{a'-1} S_{m0}[qL_e + l], \quad 0 \leqslant l \leqslant L_e - 1,$$

$$S'_{m0}[l] = \sum_{q=0}^{a-1} S'_{m0}[qL_o + l] e^{j[2\pi(pL_o+l)(a-1)/2]/N}, \quad 0 \leqslant l \leqslant L_o - 1,$$

where $a' = a/2$ if a is even and $a' = a$ if a is odd. The exponential above is nonzero for $a \geqslant 3$; this is the offset for the folding process; for example, if $a = 5$ then we decimate the time-lag function starting at $R[n+2, 2m+1]$.

(c) Then, transform to the time-lag domain:

$$R[n_{ei}, m_0] = \underset{l\to n}{\mathrm{IDFT}}\{S_{m0}[l]\},$$

$$R[n_{oi}, m_0] = \underset{l\to n}{\mathrm{IDFT}}\{S'_{m0}[l]\}, \quad \text{if } a \text{ is odd.}$$

3. Recover negative-lag from the positive-lag values to form the $R[n, m]$ of size $2N/a \times N/b$.
4. Transform the time-lag function to the t–f domain for even–odd values of n_i:
   (a) for $n_{ei} = n_{e1}, n_{e2}, \ldots, n_{eL_e}$: $\rho[n_{ei}, bk] = \mathrm{DFT}_{m\to k}\{R[n_{ei}, m]\}$;
   (b) and if a is odd, for $n_{oi} = n_{o1}, n_{o2}, \ldots, n_{oL_o}$:
      i. let,

$$\hat{R}[n_{oi}, 0] = \Im\big(R[n_{oi}, 0]\big),$$

$$\hat{R}[n_{oi}, m] = \frac{1}{2j}\big\{R[n_{oi}, m] - \big(R[n_{oi}, J-m]\big)^*\big\}, \quad 1 \leqslant m \leqslant J_h,$$

$$\hat{R}[n_{oi}, m] = \big(\hat{R}[n_{oi}, J-m]\big)^*, \quad J_h + 1 \leqslant m \leqslant J - 1.$$

      ii. DFT to the t–f domain

$$\rho[n_{oi}, bk] = \underset{m\to k}{\mathrm{DFT}}\big\{\hat{R}[n_{oi}, m]\big\} \csc(\pi bk/N).$$

      iii. Finally, do for frequency sample $k = 0$: $\rho[n_{oi}, 0] = \sum_{m=0}^{J-1} R[n_{oi}, m]$.

The algorithm uses $(N/2) \times$ FFT-N plus $(N/2b) \times$ FFT-$2N/a$ plus $(N/a) \times$ FFT-N/b operations to compute the $2N/a \times N/b$ DTFD, assuming that a is even. When a is odd, the computation load is $N \times$ FFT-N plus $(N/2b) \times$ FFT-$2N/a$ plus $(N/a) \times$ FFT-N/b; that is, only the $N/2$ term changes to N. This decimated algorithm can be applied to the separable kernel algorithms in Section 3.1 to produce an $N_{\text{time}}/a \times N_{\text{freq}}/b$ DTFD [12].

3.4. Computation and memory

To quantify computation load, we count the number of FFT operations used by the algorithm as these operations account for the majority of the algorithm's computation load [2,12]. We assume that the FFT algorithm uses $cN \log_2 N$ real multiplications and real additions to compute a length-N FFT for a complex-valued signal; parameter c is a constant specific to the FFT algorithm [14].

Table 1 shows the computation load for each algorithm together with the nonseparable kernel and DWVD algorithm from Section 2. In addition, the table also provides the computation loads for two other existing DTFD definitions: the GDTFD definition [2] and the so-called alias-free GDTFD (AF-GDTFD) [9,12]. The computation load and array size is smaller for the GDTFD definition, comparative to the load for the nonseparable kernel DTFD without decimation, but the GDTFD does not satisfy all important mathematical properties, such as the frequency marginal [7].

As an example, we tested the algorithms with a heart-rate variability (HRV) signal of length $N = 465$. We used the separable-kernel algorithm, with $P = 127$ (length of lag window) and $Q = 31$ (length of Doppler window). We set $N_{\text{time}} = 64$ and $N_{\text{freq}} = 64$ to limit oversampling. Using the metrics from Table 1, the algorithm computes the DTFD using only 12% of the computation load and less than 1% of the memory required to compute the oversampled $2N \times N$ DTFD.

4. Conclusions

The presented algorithms compute DTFDs with minimal computation and memory loads. Because the DTFD of a length-N signal requires an array of size $2N \times N$, computer memory places an upper limit on how large N can be. This memory limit differs to computation load: an algorithm with a large computation load will always compute with more time, but an algorithm with not enough available memory will not compute.

Table 1
Computation load for the kernel-specific algorithms using the analytic signal of length $2N$. For comparison, other DTFD definitions are included here; these algorithms assume that the kernel is the general nonseparable kernel. The user-selected parameter $N_{\text{time}} \leqslant 2N$ controls the oversampling in the time direction; likewise $N_{\text{freq}} \leqslant N$ controls the oversampling in the frequency direction. The decimated-nonseparable kernel, unlike the nonseparable kernel algorithm, does not compute an exact distribution, as the decimation integers a and b control the level of decimation.

DTFD type	Computation load[b]	DTFD array size
DWVD	$cN^2 \log_2 N$	$2N \times N$
Nonseparable kernel	$c3N^2 \log_2 N$	$2N \times N$
Decimated-nonseparable kernel[a]	$c(N^2/2)\log_2 N + c(2N^2/ab)\log_2(2N/ab)$	$2N/a \times N/b$
Separable kernel	$cP_{\text{h}}N\log_2 N + cP_{\text{h}}N_{\text{th}}\log_2 N_{\text{th}} + cN_{\text{th}}N_{\text{freq}}\log_2 N_{\text{freq}}$	$N_{\text{time}} \times N_{\text{freq}}$
Doppler-independent kernel	$cNN_{\text{freq}}\log_2 N_{\text{freq}}$	$2N \times N_{\text{freq}}$
Lag-independent kernel	$c(N/2)N_{\text{time}}\log_2 N_{\text{time}}$	$N_{\text{time}} \times N$
Other DTFD definitions:		
GDTFD	$c3(N^2/2)\log_2 N$	$N \times N$
AF-GDTFD	$c8N^2 \log_2 2N$	$2N \times 2N$

[a] Assumes that a is even; when a is odd, replace $c(N^2/2)\log_2 N$ with $cN^2 \log_2 N$.
[b] $P_{\text{h}} = \lfloor P/2 \rfloor$, where P is the length of lag-window $g_2[m]$, and $N_{\text{th}} = N_{\text{time}}/2$.

The separable kernel DTFD for signal analysis is recommended for two reasons. First, the kernel can be applied to many different signal types and offers more flexibility than the other three kernel categories [1]. And second, the separable-kernel algorithm computes the exact DTFD using only $N_{\text{time}} \times N_{\text{freq}}$ sample points—important for long signals when $N_{\text{time}} \ll 2N$ and $N_{\text{freq}} \ll N$. For other TFDs, such as nonseparable kernels, then the decimated nonseparable algorithm can compute these DTFDs in a fast and memory efficient manner. The decimation will eliminate information from the DTFDs but this negative effect depends on the signal and the application and can be offset against computational gains. MATLAB code for these algorithms can be found at https://spideroak.com/browse/share/fast_DTFDs/fastTFDs.

Acknowledgments

This work was funded by the Qatar National Research Fund, grant number NPRP 09-626-2-243.

Appendix A. Proof of properties

Another way to write (7) is to include the sampling information in the arguments [7], so that

$$\rho\left(\frac{nT}{2}, \frac{k}{2NT}\right) = W\left(\frac{nT}{2}, \frac{k}{2NT}\right) \circledast_n \circledast_k \gamma\left(\frac{nT}{2}, \frac{k}{2NT}\right)$$

where $1/T$ is the sampling frequency. Assuming $T = 1$, the DWVD is expressed as

$$W\left(\frac{n}{2}, \frac{k}{2N}\right) = \frac{1}{2N}\sum_{m=1}^{2N-1} K\left(\frac{n}{2}, m\right) \mathrm{e}^{-\mathrm{j}2\pi mk/N} \tag{A.1}$$

where we define the time-lag function $K(n/2, m)$ as

$$K(n, 2m) = z(n+m)z^*(n-m), \qquad K(n+1/2, 2m) = 0,$$
$$K(n, 2m+1) = 0, \qquad K(n+1/2, 2m+1) = z(n+m+1)z^*(n-m).$$

The algorithm in Section 3.1 computes the $N_{\text{time}} \times N_{\text{freq}}$ DTFD, using the separable kernel $g(l/N, m) = G_1(l/N)g_2(m)$, where $G_1(l/N)$ is length $N_{\text{time}}/2$, $g_2(m)$ is length $2N_{\text{freq}}$, and $N_{\text{freq}} < N$ and $N_{\text{time}} < 2N$. Extending the work from Ref. [15], the following shows the key steps of the proofs.

A.1. Time marginal

The DTFD satisfies the time marginal only for the Doppler-independent kernel and when $g_2(0) = 1$. For the $2N \times N_{\text{freq}}$ DTFD, with $c = N_{\text{freq}}/2N$, then $\sum_{k=0}^{N_{\text{freq}}-1} \rho(n, k/2N_{\text{freq}}) = c|z(n)|^2$.

Proof. Using the identities $\sum_{k=0}^{2N_{\text{freq}}-1} \exp(-\mathrm{j}\pi mk/N_{\text{freq}}) = 2N_{\text{freq}}\delta(m)$ (the next two proofs also use this identity with different parameters) and $g_2(0) = 1$,

$$\sum_{k=0}^{2N_{\text{freq}}-1} \rho\left(n, \frac{k}{2N_{\text{freq}}}\right) = \frac{1}{2N}\sum_{k=0}^{2N_{\text{freq}}-1}\sum_{m=0}^{2N_{\text{freq}}-1} K(n, m)g_2(m)\mathrm{e}^{-\mathrm{j}\pi mk/N_{\text{freq}}} = 2cK(n, 0)g_2(0) = 2c\left|z(n)\right|^2.$$

And because the $2N \times N_{\text{freq}}$ DTFD is periodic in N_{freq} in the frequency direction, $\sum_{k=0}^{2N_{\text{freq}}-1} \rho(n, k/2N_{\text{freq}}) = 2\sum_{k=0}^{N_{\text{freq}}-1} \rho(n, k/2N_{\text{freq}})$, and thus $\sum_{k=0}^{N_{\text{freq}}-1} \rho(n, k/2N_{\text{freq}}) = c|z(n)|^2$. □

A.2. Frequency marginal

The DTFD satisfies the frequency marginal only for the lag-independent kernel and when $G_1(0) = 1$. That is, $\sum_{n=0}^{N_{\text{time}}-1} \rho(n/2, k/2N) = c|Z(k/2N)|^2$ for $k = 0, 1, \ldots, N-1$. Here, $Z(k/2N)$ is the Fourier transform of $z(n)$ (recall that $z(n)$ is of length $2N$) and $c = N_{\text{time}}/2N$.

Proof. The $N_{\text{time}} \times N$ lag-independent DTFD can be written as

$$\rho\left(\frac{n}{2}, \frac{k}{2N}\right) = \frac{1}{2N} \sum_{m=0}^{2N-1} \sum_{l=0}^{N_{\text{time}}-1} \sum_{p=0}^{2N-1} K\left(\frac{p}{2}, m\right) \mathrm{e}^{-\mathrm{j}\pi pl/N} G_1\left(\frac{l}{N}\right) \mathrm{e}^{\mathrm{j}2\pi ln/N_{\text{time}}} \mathrm{e}^{-\mathrm{j}\pi mk/N}.$$

And as $G_1(0/N) = 1$, then

$$\sum_{n=0}^{N_{\text{time}}-1} \rho\left(\frac{n}{2}, \frac{k}{2N}\right) = \frac{N_{\text{time}}}{2N} \sum_{m=0}^{2N-1} \sum_{p=0}^{2N-1} K\left(\frac{p}{2}, m\right) \mathrm{e}^{-\mathrm{j}\pi mk/N} = c \sum_{p=0}^{2N-1} W\left(\frac{p}{2}, \frac{k}{2N}\right) = c\left|Z\left(\frac{k}{2N}\right)\right|^2,$$

as the DWVD $W(n/2, k/2N)$ satisfies the marginal property [12]. □

A.3. Instantaneous frequency

The DTFD satisfies the instantaneous frequency (IF) property for the Doppler-independent kernel only. The first moment of the $2N \times N_{\text{freq}}$ DTFD equals the IF,

$$\arg\left[\sum_{k=0}^{N-1} \rho\left(\frac{n}{2}, \frac{k}{2N}\right) \mathrm{e}^{\mathrm{j}2\pi k/N}\right] = f(n)$$

where $f(n)$ is the IF for signal $z(n) = A(n)\exp[\varphi(n)]$, estimated by the central-finite-difference method as $f(n) = \varphi(n+1) - \varphi(n-1)$. The periodic IF is defined from $f(n)$ as $[f(n) \bmod 2\pi]/(4\pi)$ [7].

Proof. Similar to the proof for the time marginal,

$$\begin{aligned}\sum_{k=0}^{2N_{\text{freq}}-1} \rho\left(n, \frac{k}{2N_{\text{freq}}}\right) \mathrm{e}^{\mathrm{j}2\pi k/N_{\text{freq}}} &= \frac{1}{2N} \sum_{k=0}^{2N_{\text{freq}}-1} \sum_{m=0}^{2N_{\text{freq}}-1} K(n, m) g_2(m) \mathrm{e}^{-\mathrm{j}\pi k(m-2)/N_{\text{freq}}} \\ &= cz(n+1)z^*(n-1) = cA(n+1)A(n-1)\mathrm{e}^{\mathrm{j}[\varphi(n+1)-\varphi(n-1)]}\end{aligned}$$

where $c = g_2(2)N_{\text{freq}}/N$. The argument of this final expression is equal to $f(n)$, thus concluding the proof. □

Appendix B. Supplementary material

The online version of this article contains additional supplementary material.
Please visit http://dx.doi.org/10.1016/j.acha.2013.01.003.

References

[1] B. Boashash (Ed.), Time–Frequency Signal Analysis and Processing: A Comprehensive Reference, Elsevier, Oxford, UK, 2003.
[2] B. Boashash, A. Reilly, Algorithms for time–frequency signal analysis, in: B. Boashash (Ed.), Time–Frequency Signal Analysis: Methods and Applications, Wiley Press, Melbourne, 1992, pp. 163–181 (Ch. 7).
[3] K. Kim, Comparison and improvement of the computational efficiencies of two FFT-based iterative solution methods for the scalar multiple-scattering equation, Commun. Comput. Phys. 5 (1) (2009) 108–125.
[4] G.S. Cunningham, W.J. Williams, Fast implementations of generalized discrete time–frequency distributions, IEEE Trans. Signal Process. 42 (6) (1994) 1496–1508.
[5] T. Le, M. Glesner, A flexible and approximate computing approach for time–frequency distributions, IEEE Trans. Signal Process. 48 (4) (2000) 1193–1196.
[6] J.R. O'Hair, B.W. Suter, The Zak transform and decimated time–frequency distributions, IEEE Trans. Signal Process. 44 (5) (1996) 1099–1110.
[7] J.M. O' Toole, M. Mesbah, B. Boashash, Improved discrete definition of quadratic time–frequency distributions, IEEE Trans. Signal Process. 58 (2) (2010) 906–911.
[8] J.M. O' Toole, M. Mesbah, B. Boashash, Algorithms for discrete quadratic time–frequency distributions, WSEAS Trans. Signal Process. 4 (5) (2008) 310–315.

[9] J.C. O'Neill, W.J. Williams, Shift covariant time–frequency distributions of discrete signals, IEEE Trans. Signal Process. 47 (1) (1999) 133–146.
[10] T.A.C.M. Claasen, W.F.G. Mecklenbräuker, The Wigner distribution—a tool for time–frequency signal analysis. Part II: Discrete-time signals, Philips J. Res. 35 (1980) 276–300.
[11] F. Peyrin, R. Prost, A unified definition for the discrete-time, discrete-frequency, and discrete-time/frequency Wigner distributions, IEEE Trans. Acoust. Speech Signal Process. 34 (4) (1986) 858–866.
[12] J.M. O' Toole, Discrete quadratic time–frequency distributions: definition, computation, and a newborn electroencephalogram application, Ph.D. thesis, School of Medicine, The University of Queensland, Nov. 2009, http://espace.library.uq.edu.au/view/UQ:185537.
[13] J.M. O' Toole, M. Mesbah, B. Boashash, A new discrete analytic signal for reducing aliasing in the discrete Wigner–Ville distribution, IEEE Trans. Signal Process. 56 (11) (2008) 5427–5434.
[14] P. Duhamel, M. Vetterli, Fast Fourier transforms: a tutorial review and a state of the art, Signal Process. 19 (1990) 259–299.
[15] J.M. O' Toole, M. Mesbah, B. Boashash, Proofs for discrete time–frequency distribution properties, Tech. Rep. UQCCR-2009-06-09, UQ Centre for Clinical Research and Perinatal Research Centre, The University of Queensland, Australia, Jun. 2009, http://espace.library.uq.edu.au/view/UQ:178641.