# LTFAT: A Matlab/Octave toolbox for sound processing

Zdeněk Průša, Peter L. Søndergaard, Nicki Holighaus, and Peter Balazs
Email:
{zdenek.prusa,peter.soendergaard,peter.balazs,nicki.holighaus}@oeaw.ac.at

Acoustics Research Institute, Austrian Academy of Sciences, Wohllebengasse 12–14,
1040 Vienna, Austria

**Abstract.** To visualize and manipulate musical signals time-frequency transforms have been used extensively. The Large Time Frequency Analysis Toolbox is an Octave/Matlab toolbox for modern signal analysis and synthesis. The toolbox provides a large variety of linear and invertible time-frequency transforms like Gabor, MDCT, constant-Q, filterbanks and wavelets transforms, and routines for modifying musical signal by manipulating coefficients by linear and non-linear methods. Combined with this the toolbox also supplies a framework for real-time processing of sound signals. It also provides demo scripts devoted either to demonstrating the main functions of the toolbox, or to exemplify their use in specific signal processing applications.

## 1 Introduction

Time-Frequency analysis has been used extensively in musical signal processing to visualize music signals and, if a reconstruction algorithm exists, to modify and manipulate them. A common tool is the phase vocoder [14], which for example can be used for time stretching or pitch shifting. This algorithm relies on the Short Time Fourier Transform (STFT) as signal processing background.

While the STFT is very useful for musical signal processing, for some application the rigid structure, resulting in a fixed time-frequency resolution, might not be optimal. Therefore several other time-frequency representation like the wavelet transform [17] or the non-stationary Gabor transform [5] have been used. In particular for the manipulation of musical signals directly in the analysis coefficient domain, for example amplifying or attenuating particular time-frequency regions, a reconstruction method is necessary. And, because if no modification is done, the original signal should be kept, perfect reconstruction is necessary. To guarantee that for adapted time-frequency transforms, the concept of frames has been proved to be very useful [3].

The concept of *frames* was introduced in [15], made popular by [10], and became a very active field of mathematics [8]. Frames allow redundant representations, i.e. having more coefficients than samples. Finding and constructing frames, given certain *a-priory* properties, is easier than for orthonormal basis

transforms (ONBs). This can readily be experienced in time-frequency analysis: The widely used Gabor transform [16] can be much better localized in the time-frequency domain if it constitutes a redundant frame rather than a basis. We note that even if it is impossible to find an ONB with certain properties, it is often possible to find a frame. Moreover, analysis with redundant frames can have the advantage that it is easier to directly interpret the coefficients, e.g. for Gabor sequences by the time-frequency localization. This is advantageous for many applications.

The Large Time Frequency Analysis Toolbox (LTFAT) is an Octave/Matlab toolbox built upon frames. By using frame theory as a unifying common language, it provides a plethora of signal transforms like Gabor frames, Wavelet bases and frames, filterbanks, non-stationary Gabor systems etc. using common interfaces.

In this paper we present a preview of the next major version (2.0) of LTFAT, the major linear transforms, the analysis and synthesis methods and the block-processing framework. In comparison to the first major version [36] the toolbox further includes wavelets, block processing and the object-oriented framework for frames.

Overall, LTFAT combines a large and well-documented mathematical knowledge with an easy to use programming language and a real-time sound sound processing framework. This allows students, researchers and musicians to learn the underlying mathematical concepts by reading the documentation, programming their own experiments in Octave and Matlab and getting immediate feedback while listening to the output of their experiments.

## 2 Frames

Formally, a frame is a collection of functions $\Psi = (\psi_\lambda)_{\lambda\in\Lambda}$ in a Hilbert space $\mathcal{H}$ such that $0 < A \leq B < \infty$ exist with $A\|f\|^2 \leq \sum_\lambda |\langle f, \psi_\lambda\rangle|^2 \leq B\|f\|^2$ for all $f \in \mathcal{H}$ and is called *tight*, if $A = B$. The basic operators associated with frames are the *analysis* and *synthesis operators* given by $(\mathbf{C}_\Psi f)[\lambda] = \langle f, \psi_\lambda\rangle$ and $\mathbf{D}_\Psi c = \sum_\lambda c_\lambda \psi_\lambda$, for all $f \in \mathcal{H}$ and $(c_\lambda) \in \ell^2(\mathbb{Z})$, respectively. Their concatenation $\mathbf{S}_\Psi = \mathbf{D}_\Psi \mathbf{C}_\Psi$ is referred to as the *frame operator*. Any frame admits a, possibly non-unique, dual frame, i.e. a frame $\Psi^d$ such that $\mathbf{I} = \mathbf{D}_{\Psi^d}\mathbf{C}_\Psi = \mathbf{D}_\Psi \mathbf{C}_{\Psi^d}$. The most widely used dual is the so called *canonical dual* that can be obtained by applying the inverse frame operator $\mathbf{S}_\Psi^{-1}$ to the frame elements $\psi_\lambda^d = \mathbf{S}_\Psi^{-1}\psi_\lambda$. When we prefer to have a tight system for both analysis and synthesis, we can instead use the *canonical tight frame* $\Psi^t = (\psi_\lambda^t)_\lambda$, defined by $\psi_\lambda^t = \mathbf{S}_\Psi^{-\frac{1}{2}}\psi_\lambda$ and satisfying $\mathbf{I} = \mathbf{D}_{\Psi^t}\mathbf{C}_{\Psi^t}$. For algorithmical purposes, like considered in this paper, sampled functions, i.e. $\mathcal{H} = \ell^2(\mathbb{Z})$ are considered, for the concrete computations finite dimensional signals are used, $\mathcal{H} = \mathbb{C}^L$, see e.g. [2].

### 2.1 Frames and Object Oriented Programming

The notion of a frame fits very well with the notion of a *class* in programming languages. A class is a collection of methods and variables that together forms

a logical entity. A class can be *derived* or *inherited* from another class, in such a case the derived class must define all the methods and variables of the original class, but may add new ones. In the framework presented in this paper, the frame class serves as the base class from which all other classes are derived.

A frame class is instantiated by the user providing information about which type of frame is desired, and any additional parameters (like a window function, the number of channels etc.) necessary to construct the frame object. This is usually not enough information to construct a frame for $\mathbb{C}^L$ in the mathematical sense, as the dimensionality $L$ of the space is not supplied. Instead, when the analysis operator of a frame object is presented with an input signal, it determines a value of $L$ larger than or equal to the length of the input signal and only at this point is the mathematical frame fully defined. The construction was conceived this way to simplify work with different length signals without the need for a new frame for each signal length.

Therefore, each frame type must supply the `framelength` method, which returns the next larger length for which the frame can be instantiated. For instance, a dyadic wavelet frame with $N$ levels only treats signal lengths which are multiples of $2^N$. An input signal is simply zero-extended until it has admissible length, but never truncated. Some frames may only work for a fixed length $L$.

The `frameaccel` method will fix a frame to only work for one specific space $\mathbb{C}^L$. For some frame types, this allows precomputing data structures to speed up the repeated application of the analysis and synthesis operators. This is highly useful for iterative algorithms, block processing or other types of processing where a predetermined signal length is used repeatedly.

Basic information about a frame can be obtained from the `framebounds` methods, returning the frame bounds, and the `framered` method returning the redundancy of the frame.

### 2.2 Analysis and Synthesis

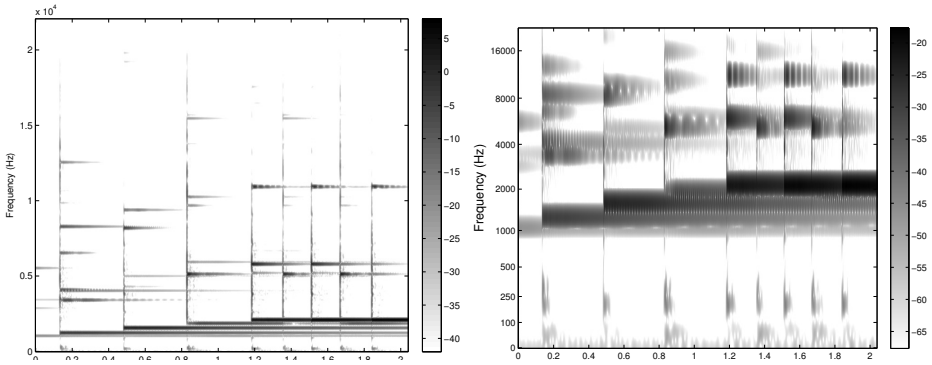
The workhorses of the framework are the `frana` and `frsyn` methods, providing the analysis $\mathbf{C}_\Psi$ and synthesis operators $\mathbf{D}_\Psi$ of the frame $\Psi$. These methods use a fast algorithm if available for the given frame. They are the preferred way of interacting with the frame when writing algorithms. However, if direct access to the operators are needed, the `framematrix` method returns a matrix representation of the synthesis operator.

For some frame types, e.g. `filterbank` and `nsdgt`, the canonical dual frame is not necessarily again a frame with the same structure, and therefore it cannot be realized with a fast algorithm. Nonetheless, analysis and synthesis with the canonical dual frame can be realized iteratively. The `franaiter` method implements iterative computation of the canonical dual analysis coefficients using the frame operator's self-adjointness via the equation $\langle f, \mathbf{S}^{-1}\phi_\lambda\rangle = \langle \mathbf{S}^{-1} f, \phi_\lambda\rangle$. More precisely, a conjugate gradients method (`pcg`) is employed to apply the inverse frame operator $\mathbf{S}^{-1}$ to the signal $f$ iteratively, such that the analysis coefficients can be computed quickly by the `frana` method. Note that each conjugate gradients iteration applies both `frana` and `frsyn` once. The method `frsyniter` works

in a similar fashion to provide the action of the inverse of the frame analysis operator. Furthermore, for some frame types the diagonal of the frame operator $\mathbf{S}$ can be used as a preconditioner, providing significant speedup whenever the frame operator is diagonally dominant.

While both methods `franaiter` and `frsyniter` are available for all frames, they are recommended only if no means of efficient, direct computation of the canonical dual frame exists or its storage is not feasible. Their performance is highly dependent on the frame bounds and the efficiency of `frana` and `frsyn` for the frame type used.

## 3 Filters and Filterbanks




(a) Spectrogram (b) ERBlet

Fig. 1: Two redundant signal representations of the same signal, an excerpt of the *glockenspiel* test signal. The figure on the left shows a classical spectrogram with a linear frequency scale, while the figure on the right shows an ERBlet transform, where the centre frequencies are equidistantly spaced on the ERB-scale and the bandwidth of the channels are constant if measured in ERB.

Filterbanks are more general constructions than Gabor frames, allowing for independent filters in each frequency channel. The output coefficients $c$ of an $M$-channel filterbank is given by

$$c_m(n) = \sum_{l=0}^{L-1} f(l)\, g(a(m)n - l), \tag{1}$$

If the same time-shift $a = a(m)$ is used across all channels, the filterbank is said to be *uniform* [7]. Uniform filterbank frames have the advantage that canonical dual and tight frames are again uniform filterbanks, making perfect reconstruction filter construction somewhat easier. On the other hand, the uniformity

usually means than too many coefficients are kept for subband channels with a small bandwidth.

Another approach to filterbank inversion is to construct the filterbank in such a way that it becomes a *painless* frame [10]. A painless frame has the property that its frame operator is a diagonal matrix. This makes it easy to find the canonical dual and tight frames, and in the case of painless filterbanks they are again painless filterbanks. A filterbank is painless if the filters are strictly bandlimited with a bandwidth (in radians) that is less than or equal to $2\pi/a(m)$.

In a general filterbank, the user must provide filters to cover the whole frequency axis, including the negative frequencies. For users working with real-valued signals only, a real filterbank construction exists in LTFAT. These constructions work as if the filterbank was extended with the conjugates of the given filters. They work entirely similar as the real valued Gabor frames.

ERBlets [28] is a family of perfect reconstruction filterbanks with a frequency resolution that follows the ERB-scale [19]. The ERBlets are included in LTFAT through a routine that generates the correct filters and downsampling rates. To aid researchers working with auditory signal processing, the toolbox contains a small collection of routines to generate the most common auditory scales, range compression and specialized auditory filters. An highly redundant ERBlet-representation of a common test signal is shown on Figure 1b, to create an auditory "spectrogram".

## 4 Gabor Analysis: Linear Frequency Scales

The Discrete Gabor Transform (DGT) with $M$ channels, time-shift of $a$ and window function $g \in \mathbb{C}^L$ is given by

$$c(m,n) = \sum_{l=0}^{L-1} f(l)\,\overline{g}(l-na)\,e^{-2\pi iml/M},$$

where $m = 0,\ldots,M-1$ and $n = 0,\ldots,L/a$. An overview of the theory of Gabor frames can be found in [21]. The toolbox supports two types of Gabor systems: the normal type `dgt` and a type `dgtreal` which only works for real-valued signals. This type of frame simply returns the coefficients of the positive frequencies in the time-frequency plane. For practical applications it is a convenient way of not having to deal with the redundant information in the negative frequencies. An highly redundant DGT-representation of a common test signal is shown on Figure 1a, this is simply a normal spectrogram.

### 4.1 The Discrete Wilson Transform and the MDCT

The `wilson` frame type represents a type of time-frequency basis known as a Wilson basis [12]. Wilson bases were proposed as substitutes for Gabor frames, because of the impossibility of constructing Gabor systems that would be simultaneously generated using well-behaved windows, and bases of the considered signal spaces.

A Wilson basis is formed by taking linear combinations of appropriate basis functions from a Gabor frame with redundancy 2, [6]. Essentially Gabor atoms of positive and negative frequencies are combined, with suitable fine tuning of their phases. This remarkable construction turns a tight Gabor frame into an real, orthonormal basis, or turns a non-tight Gabor frame into a Riesz basis (corresponding to a bi-orthogonal filterbank). In [25] this system is described as a "linear phase cosine modulated maximally decimated filter bank with perfect reconstruction".

The MDCT (modified discrete cosine transform) is another substitute for the non-existent well localised Gabor bases that has become extremely popular recently for its numerous applications, in audio coding for instance [27,31,30]. Both Wilson and MDCT bases are variations of the same construction, the notable difference being that the basis vectors of a Wilson basis with $M$ channels are centered on the $M$ roots of unity in frequency, while the MDCT basis functions are centered in between.

The coefficients $c \in \mathbb{C}^{M\times N}$ computed by the MDCT of $f \in \mathbb{C}^L$ are given by:
For $m+n$ even:

$$c\,(m,n) = \sqrt{2}\sum_{l=0}^{L-1} f(l)\cos\left(\frac{\pi}{M}\left(m+\frac{1}{2}\right)l+\frac{\pi}{4}\right)g(l-na). \tag{2}$$

For $m+n$ odd:

$$c\,(m,n) = \sqrt{2}\sum_{l=0}^{L-1} f(l)\sin\left(\frac{\pi}{M}\left(m+\frac{1}{2}\right)l+\frac{\pi}{4}\right)g(l-na). \tag{3}$$

MDCT coefficients of a common test signal are shown on Figure 2a.

### 4.2 Adaptable Time Scale

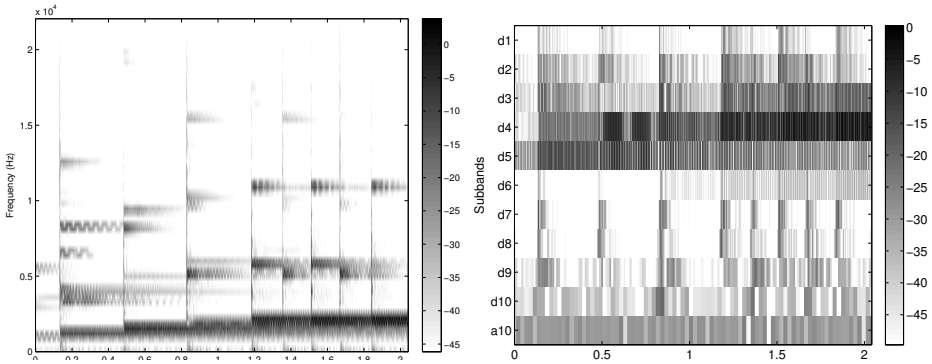
Non-stationary Discrete Gabor Systems (NSDGS) [5] is a generalization of Gabor frames, where window and time-shift are allowed to change over time, but the frequency channels are always placed on a linear scale (through the proper application of a Discrete Fourier Transform).

Similar to filterbanks, an NSDGT must be either uniform or painless to a have a fast linear reconstruction. A uniform NSDGS has the same frequency resolution for all time-shifts and a painless NSDGT always has a window length that is less than or equal to the corresponding number of channels. In these cases, the dual and tight systems are again NSDGTs. As for Gabor systems, the real-valued NSDGT provides only the positive frequencies of the DFT.

NSDGTs are usefull for adapting the time and frequency resolution over time, for instance for tracking the pitch changes in a voice or musical signal.

## 5 Wavelet Analysis: Logarithmic Frequency Scale

The newly added wavelet module extends the one-dimensional time-frequency signal processing capabilities of the toolbox. The module is intended to be in-




(a) MDCT (b) DWT

Fig. 2: Two non-redundant signal representations of the same signal, a piece of the *glockenspiel* test signal. The figure on the left shows the output from a Modified Discrete Cosine Transform with 64 channels, while the figure on the right shows a Discrete Wavelet transform with a depth of $J = 10$. The MDCT has a linear, while the DWT has a logarithmic frequency scale.

tuitive, self-contained (in a sense that all dependencies are within the LTFAT toolbox) and compatible with some of the existing routines. The term *wavelets* should be understood vaguely in this context, because in the discrete wavelet transform setting the routines in the Wavelet module are capable of calculating all transforms build upon and extending the basic iterated two-channel filterbank scheme (Mallat's algorithm [26]) such as Framelets [11], dual-tree $\mathbb{C}$WT [35], M-band wavelets [24] and even more complex constructions.

Note that the discrete wavelet transform routines can be directly used for framelet-type transforms with an arbitrary number of filters in the basic iteration filterbank. Building custom wavelet filterbank trees including any tree shape and different elementary filterbanks is also possible. A smooth transition between the custom wavelet filterbank trees and the non-uniform non-iterated identical filterbanks is another feature of the module.

### 5.1 The Discrete Wavelet Transform (DWT)

The DWT provides a multiresolution decomposition into $J$ octaves of the discrete signal $f \in \ell^2(\mathbb{Z})$ in terms of coordinates in a basis given by

$$f(n) = \sum_{j=1}^{J} \sum_{k \in \mathbb{Z}} d_j(k) \tilde{g}_j(n - 2^j k) + \sum_{k \in \mathbb{Z}} a_J(k) \tilde{h}_J(n - 2^J k), \tag{4}$$

where $\tilde{g}_j$ is the synthesis wavelet sequence and $\tilde{h}_J$ is the synthesis scaling sequence. The wavelet (detail) coefficients $d_j(k)$, for $j = 1, \ldots, J$ and scaling (ap-

proximation) coefficients $a_J(k)$ are given by

$$d_j(k) = \sum_n f(n) g_j^*(n - 2^j k) \tag{5}$$

and

$$a_J(k) = \sum_n f(n) h_J^*(n - 2^J k) \tag{6}$$

respectively, where $g_j^*(n)$ is the complex conjugate of the analysis wavelet sequence and $h_J^*(n)$ is the complex conjugate of the analysis scaling sequence. The analysis sequences are related in such a way that they are built from the two suitably designed half-band elementary filters $g_1(n)$ (high-pass, also referred to as the *discrete wavelet*) and $h_1(n)$ (low-pass) as follows

$$h_{j+1}(n) = \sum_k h_j(k) h_1(n - 2k), \tag{7}$$

$$g_{j+1}(n) = \sum_k g_j(k) h_1(n - 2k). \tag{8}$$

The same procedure holds for the synthesis sequences but with the different elementary filters $\tilde{g}_1(n)$ and $\tilde{h}_1(n)$. Perfect reconstruction is possible if the elementary filters have been suitably designed. The equations (7),(8) are in fact an enabling factor for the well-known Mallat's algorithm (also known as the fast wavelet transform). It comprises of an iterative application of the time-reversed elementary two-channel FIR filter bank followed by a factor of two subsampling

$$d_{j+1k}(k) = \left(a_j * g_1(\cdot - n)\right)_{\downarrow 2}(k), \tag{9}$$

$$a_{j+1}(k) = \left(a_j * h_1(\cdot - n)\right)_{\downarrow 2}(k), \tag{10}$$

make where $*$ is the convolution operation and $a_0 = f$. The iterative application of the elementary filterbank forms a tree-shaped filterbank, where just the low-pass output is iterated on. The signal reconstruction from the coefficients is then done by applying a mirrored filterbank tree using the synthesis filters $\tilde{g}_1(n)$ and $\tilde{h}_1(n)$.

In practice when $f \in \mathbb{C}^L$, the signal boundaries have to be taken into account. Usually the periodic extension is considered (which means the convolutions (9),(10) are circular), which requires $L$ to be an integer multiple of $2^J$. In this case, the number of coefficients is halved with each iteration and the overall coefficient count is equal to $L$. The approach to considering any other extension (symmetric, zero-padding, etc.) exploits the fact that the filters are FIR and thus the information about the signal extensions can be saved in the additional coefficients. The coefficients are obtained by the full linear convolution with (now causal) filters. The resulting wavelet representation is called *expansive* because the number of coefficients at level $j$ becomes $L_j = \lfloor 2^{-j} L + (1 - 2^{-j})(m - 1) \rfloor$ [34], where $m$ is the length of the filters, with no restrictions on $L$.

DWT coefficients of a common test signal are shown on Figure 2b.

### 5.2 General Filterbank Trees

The DWT is known to have several drawbacks. First, it is merely $2^J$-shift invariant, which becomes a burden in denoising schemes. Secondly, the critical subsampling introduces aliasing which is supposed to be cancelled by the synthesis filterbank. Provided some modification of the coefficients has been done, the aliasing may no longer be compensated for. Finally, the octave frequency resolution may not be enough for some applications. The first two shortcomings can be avoided by the undecimated DWT with a cost of a high redundancy and the frequency resolution may be improved by the use of the wavelet packets, where on the other hand the aliasing is an even greater issue. Several modifications of the DWT filterbank tree were proposed to avoid the mentioned shortcomings still maintaining the wavelet filter tree structure but using different numbers of the elementary filters, adding parallel wavelet filter trees, alternating different elementary filter sets etc. All these alternative constructions in both decimated and undecimated versions are incorporated in the Wavelet module by means of the general filterbank tree framework.

The framework also encompasses building custom wavelet packets and wavelet packet subtrees, which differ from the DWT-shaped trees by allowing further recursive decomposition of the high-pass filter output creating possibly a full tree filterbank. The wavelet packet coefficients are outputs of each of the nodes in the tree. Such a representation is highly redundant, but leaves of any admissible subtree form a basis. The best subtree (basis) search algorithm relies on comparing the *entropy* of the wavelet packet coefficient subbands.

### 5.3 CQT

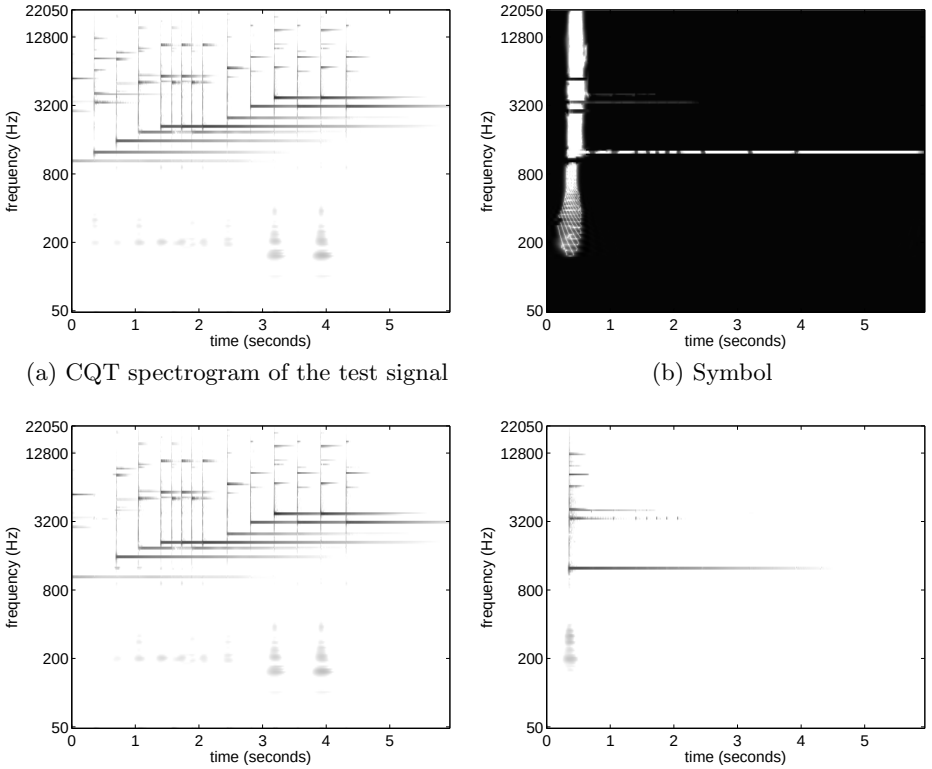
Additionally, LTFAT provides the method `cqt` for perfectly invertible constant-Q (CQ) filterbanks [22]. While conceptually reminiscent of Wavelet transforms, CQ techniques use a much higher number of channels per octave, resulting in a more detailed, redundant representation. The filters in a CQ transform are placed along the frequency axis with a constant ratio of center frequency to bandwidth, or Q-factor. Particularly interesting for acoustic signal processing, they provide a much finer frequency resolution than classical Wavelet techniques and harmonic structures are left invariant under a shift across frequency channels.

## 6 Operations on Coefficients

### 6.1 Frame multipliers

A *frame multiplier* [4] is an operator constructed by multiplying frame coefficients with a symbol $m$:

$$M_m f = \sum_{k=0}^{K-1} m_k \left\langle f, \Psi_k^a \right\rangle \psi_k^s,$$




(a) CQT spectrogram of the test signal

(b) Symbol

(c) Effect of the multiplier

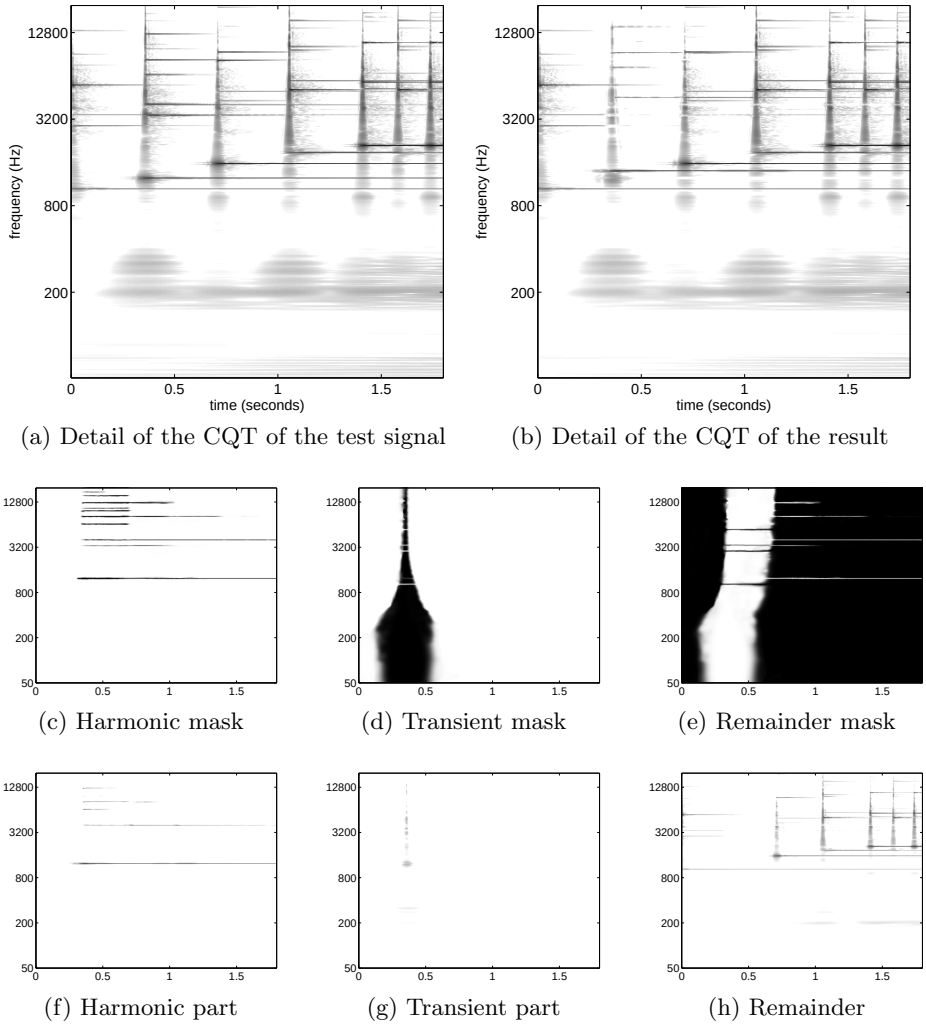
(d) Effect of the multiplier using the inverse symbol.

Fig. 3: Deleting/isolating object in a spectrogram using a frame multiplier. Values of the mask (symbol) on (b) are between 0 (white) and 1 (black). The results on (c) and (d) are obtained by an analysis of the outcome of the multiplier operator.

where $\Psi_k^a$ and $\Psi_k^s$ are simply the $k$th elements of the analysis and synthesis frames, respectively. The analysis and synthesis frames need not be of the same type, but they must have the exact same redundancy. The method `framemul` is the basic method that applies a frame multiplier, given an appropriate frames and a symbol. Its adjoint can be computed by `framemuladj`, useful for iterative algorithms.

Figure 3 shows an example of an effect of a frame multiplier on the *glockenspiel* test signal using the CQT frame (and its dual) producing coefficients as shown on Figure 3a and using symbol shown on Figure 3b and its inverse.

Figure 4 shows an example of editing the CQT spectrogram in order to isolate and transpose (two semitones up) a separate harmonic structure. Three separate masks are used: 4c to isolate transient part of the structure, 4d to isolate




(a) Detail of the CQT of the test signal (b) Detail of the CQT of the result

(c) Harmonic mask (d) Transient mask (e) Remainder mask

(f) Harmonic part (g) Transient part (h) Remainder

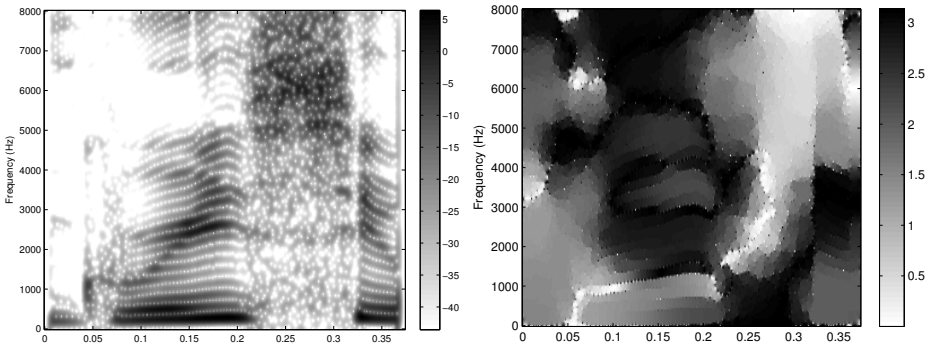
Fig. 4: Transposition of a single harmonic structure of the test signal *glockenspiel*.

harmonic part of the structure and 4e to remove the structure to be replaced with the transposed version. The result of the masking operation is shown on 4f, 4g and 4h respectively. The only modification done is a frequency shift of the harmonic part 4f by 8 bins upwards. The transient part is left as is to avoid phasing effects. The inverse transform is applied to the element-wise sum of the transient, the remainder and the modified harmonic coefficients layers. The CQT spectrogram of the result is shown on 4b.

The CQT used in both examples was defined for the frequency range 50 Hz – 20 kHz with 48 bins per octave.

Sound examples can be found at `http://ltfat.sourceforge.net/notes/022`.

### 6.2 Non-linear Analysis and Synthesis

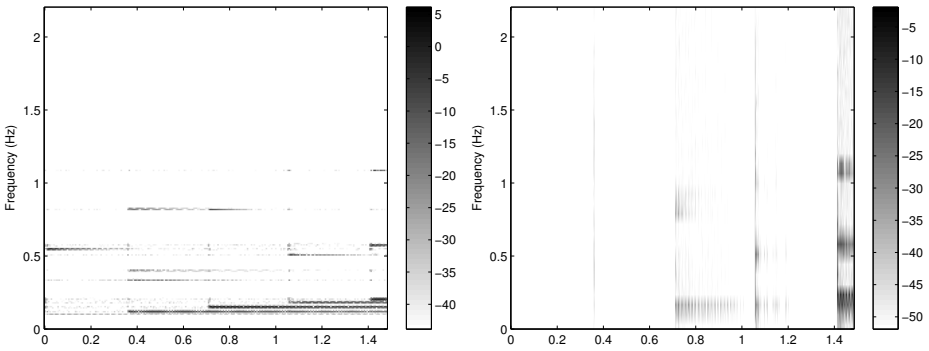



(a) Magnitude of coefficients (b) Phase difference

Fig. 5: The figure on the left shows a spectrogram of the test signal *greasy*. The figure on the right shows the difference between the phase of a STFT of the original signal, and the phase of the STFT of a reconstructed signal obtained by the Griffin-Lim algorithm.

**Reconstruction from magnitude only:** For a generic frame more than 4 times redundant, it has been shown in [1] that a signal can be reconstructed from the magnitude of its coefficients. A classical method for finding a solution to this problem is the Gerchberg–Saxton algorithm, [18] originally developed for image diffraction. For the short-time Fourier transform, a similar algorithm by Griffin and Lim was proposed in [20]. The `frsynabs` method attempts to reconstruct a signal from the magnitude of the given frame coefficients using the Griffin-Lim algorithm or more recent algorithms [13,29]. An example is shown on Figure 5. Theoretically, the algorithm should reproduce the original phase, up to a single, global phase shift, instead one obtains a pattern of local regions of constant phase shifts like the one visible on 5b. This phenomenon is due to the numerical limitations and the finite running time of the algorithm.

**Separation of tonal and transient parts:** Another nonlinear approach to analysis is searching for a sparse coefficient representation of the input signal. The `franalasso` achieves this by means of a LASSO method [9] to minimize the $l^1$-norm of the coefficients. Alternatively, the *group LASSO* method `franagrouplasso` [23] can be used to sparsify either transients or tonal com-




(a) Tonal part (b) Transient part

Fig. 6: Splitting of a piece of the *glockenspiel* test signal into transient and harmonic parts by use of the group LASSO method and and two MDCT systems with 256 and 32 channels, respectively. These figures are reproduced by `demo_audioshrink`.

ponents of a frame representation. An example is shown on Figure 6. For more information on that refer to [3].

## 7 Block-processing

The unified LTFAT block-processing framework allows supported transforms to be carried out on blocks of the input data. The output blocks can be assembled to generate the result. Using the chosen transform type, the input data block is first analyzed, producing the transform coefficients and then synthesized. During the process, the coefficients in the transform domain are free to be modified. However, block synthesis using the modified coefficients can introduce audible blocking artefacts due to the possible long/infinite analysis filter time-domain supports. Therefore, the transform calculations need to be done carefully or even modified to avoid or at least compensate for the blocking artefact. The general approach used in the framework exploits overlapped "slicing" windows introduced in [22], originally for the CQT transforms. The disadvantage of this approach is that the coefficient processing algorithms have to take into account the fact that the coefficients reflects the shape of the slicing window. The blocking artefacts can be avoided completely when working with transforms using finite filters such as DWT, DGT with finite-length windows and FIR filterbanks in general. The price to pay is an increased processing delay roughly equal to the longest filter length.

For a solution of this problem the SegDWT [33] algorithm employs an overlap-save principle for the analysis part and an overlap-add principle for the synthesis part. Simply put, prior to the analysis, it extends the segment from the left side using the previous samples. This extension ensures that after the analysis, the

obtained coefficients are exactly the ones one would get analyzing the whole input signal and picking up just those coefficients belonging to the processed block. Because of this feature, any coefficient processing algorithm can be applied with the same impact as if the same algorithm was applied to coefficients without dividing the data into blocks at all. The reconstructed segments have the length of the extended analyzed ones and the overlapping parts are simply added. As for the SegDWT algorithm itself, the left extension length required prior to the analysis of a given block is

$$L(S_n) = r(J) + (S_n \mod 2^J), \tag{11}$$

where $J$ stands for the number of the wavelet filterbank iterations, $S_n$ for first sample index of the segment $n$ in the global point of view and $r(J) = (2^J - 1)(m-2)$, where $m$ is the wavelet filter length. Note that the SegDWT algorithm accepts any block size $s$ (up to a minimum length $s = 2^J$) and the block sizes can even vary among each other. After processing the wavelet coefficients and application of the inverse Mallat's algorithm, the last $L(S_n + s)$ samples should be saved to be added to the respective reconstructed samples of the following block. In case $L(S_n + s) > s$, additional and more complex buffering have to be employed. The algorithm delay is $r(J)$ samples for block lengths restricted to values $s = k2^J, k = 1, 2, 3, \ldots$ and $r(J) + 2^J - 1$ otherwise.

The LTFAT block-processing framework combined with suitable open-source audio I/O libraries, like Portaudio `http://www.portaudio.com/` and Playrec `http://www.playrec.co.uk/` allows for the true real-time audio stream processing in Matlab/Octave. The libraries provide interfaces for the cross-platform non-blocking audio recording and playback. Such processing requires the transform routines to be fast enough to deliver the processed blocks on time to assure gapless playback. Not only for this purpose, many of the transforms included in LTFAT were implemented separately in C programming language.

An accompanying contribution [32], presented at this conference, demonstrates capabilities of the block-processing framework in the real-time setting.

## References

1. Balan, R., Casazza, P., Edidin, D.: On signal reconstruction without phase. Appl. Comput. Harmon. Anal. 20(3), 345–356 (2006)
2. Balazs, P.: Frames and finite dimensionality: Frame transformation, classification and algorithms. Applied Mathematical Sciences 2(41–44), 2131–2144 (2008)
3. Balazs, P., Dörfler, M., Kowalski, M., Torrésani, B.: Adapted and adaptive linear time-frequency representations: a synthesis point of view. IEEE Signal Processing Magazine (special issue: Time-Frequency Analysis and Applications) to appear, – (2013)
4. Balazs, P.: Basic definition and properties of Bessel multipliers. Journal of Mathematical Analysis and Applications 325(1), 571–585 (January 2007)
5. Balazs, P., Dörfler, M., Holighaus, N., Jaillet, F., Velasco, G.: Theory, implementation and applications of nonstationary Gabor frames. Journal of Computational and Applied Mathematics 236(6), 1481–1496 (2011)

6. Bölcskei, H., Feichtinger, H.G., Gröchenig, K., Hlawatsch, F.: Discrete-time Wilson expansions. In: Proc. IEEE-SP 1996 Int. Sympos. Time-Frequency Time-Scale Analysis (june 1996)
7. Bölcskei, H., Hlawatsch, F., Feichtinger, H.G.: Frame-theoretic analysis of oversampled filter banks. Signal Processing, IEEE Transactions on 46(12), 3256–3268 (2002)
8. Christensen, O.: Frames and Bases. An Introductory Course. Applied and Numerical Harmonic Analysis. Basel Birkhäuser (2008)
9. Daubechies, I., Defrise, M., De Mol, C.: An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. Communications in Pure and Applied Mathematics 57, 1413–1457 (2004)
10. Daubechies, I., Grossmann, A., Meyer, Y.: Painless non-orthogonal expansions. J. Math. Phys. 27, 1271–1283 (1986)
11. Daubechies, I., Han, B., Ron, A., Shen, Z.: Framelets: MRA-based constructions of wavelet frames. Applied and Computational Harmonic Analysis 14(1), 1 – 46 (2003)
12. Daubechies, I., Jaffard, S., Journé, J.: A simple Wilson orthonormal basis with exponential decay. SIAM J. Math. Anal. 22, 554–573 (1991)
13. Decorsière, R., Søndergaard, P.L., MacDonald, E.N., Dau, T.: Optimization Approach to the Reconstruction of a Signal from a Spectrogram-Like Representation. IEEE Trans. Acoust. Speech Signal Process. (submitted, 2013)
14. Dolson, M.: The phase vocoder: a tutorial. Computer Musical Journal 10(4), 11–27 (1986)
15. Duffin, R.J., Schaeffer, A.C.: A class of nonharmonic Fourier series. Trans. Amer. Math. Soc. 72, 341–366 (1952)
16. Feichtinger, H.G., Strohmer, T. (eds.): Gabor Analysis and Algorithms. Birkhäuser, Boston (1998)
17. Flandrin, P.: Time-Frequency/Time-Scale Analysis. Academic Press, San Diego (1999)
18. Gerchberg, R.W., Saxton, W.O.: A practical algorithm for the determination of the phase from image and diffraction plane pictures. Optik 35(2), 237–250 (1972)
19. Glasberg, B.R., Moore, B.: Derivation of auditory filter shapes from notched-noise data. Hearing Research 47(1-2), 103 (1990)
20. Griffin, D., Lim, J.: Signal estimation from modified short-time Fourier transform. IEEE Trans. Acoust. Speech Signal Process. 32(2), 236–243 (1984)
21. Gröchenig, K.: Foundations of Time-Frequency Analysis. Birkhäuser (2001)
22. Holighaus, N., Dörfler, M., Velasco, G.A., Grill, T.: A framework for invertible, real-time constant-Q transforms. IEEE Transactions on Audio, Speech and Language Processing 21(4), 775 –785 (2013)
23. Kowalski, M.: Sparse regression using mixed norms. Appl. Comput. Harmon. Anal. 27(3), 303–324 (2009)
24. Lin, T., Xu, S., Shi, Q., Hao, P.: An algebraic construction of orthonormal M-band wavelets with perfect reconstruction. Applied mathematics and computation 172(2), 717–730 (2006)
25. Lin, Y.P., Vaidyanathan, P.: Linear phase cosine modulated maximally decimated filter banks with perfectreconstruction. IEEE Trans. Signal Process. 43(11), 2525–2539 (1995)
26. Mallat, S.G.: A theory for multiresolution signal decomposition: The wavelet representation. IEEE Trans. Pattern Anal. Mach. Intell. 11(7), 674–693 (Jul 1989)
27. Malvar, H.S.: Signal Processing with Lapped Transforms. Artech House Publishers (1992)

28. Necciari, T., Balazs, P., Holighaus, N., Søndergaard, P.L.: The ERBlet transform: An auditory-based time-frequency representation with perfect reconstruction. In: Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2013). pp. 498–502. IEEE, Vancouver, Canada (May 2013)
29. Perraudin, N., Balazs, P., Søndergaard, P.L.: A fast Griffin-Lim algorithm. In: Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA) (2013)
30. Princen, J.P., Johnson, A.W., Bradley, A.B.: Subband/transform coding using filter bank designs based on time domain aliasing cancellation. Proceedings - ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing pp. 2161–2164 (1987)
31. Princen, J.P., Bradley, A.B.: Analysis/synthesis filter bank design based on time domain aliasing cancellation. IEEE Transactions on Acoustics, Speech, and Signal Processing ASSP-34(5), 1153–1161 (1986)
32. Průša, Z., Søndergaard, P.L., Balazs, P., Holighaus, N.: Real-Time Audio Processing in the Large Time Frequency Analysis Toolbox (2013), to be presented at 10th International Symposium on Computer Music Multidisciplinary Research (CMMR)
33. Prusa, Z.: Segmentwise Discrete Wavelet Transform. Ph.D. thesis, Brno University of Technology, Brno (2012)
34. Rajmic, P., Prusa, Z.: Discrete Wavelet Transform of Finite Signals: Detailed Study of the Algorithm. submitted (2013)
35. Selesnick, I., Baraniuk, R., Kingsbury, N.: The dual-tree complex wavelet transform. Signal Processing Magazine, IEEE 22(6), 123 – 151 (nov 2005)
36. Søndergaard, P.L., Torrésani, B., Balazs, P.: The Linear Time Frequency Analysis Toolbox. International Journal of Wavelets, Multiresolution Analysis and Information Processing 10(4) (2012)