The Phase Retrieval Toolbox

Zdeněk Průša

***Abstract*—A Matlab/GNU Octave toolbox for phase (signal) reconstruction from the short-time Fourier transform (STFT) magnitude is presented. The toolbox provides an implementation of various, conceptually different algorithms ranging from the well known Griffin-Lim algorithm and its derivatives to the very recent ones. The list includes real-time capable algorithms which are also implemented in real-time audio demos running directly in Matlab/GNU Octave. The toolbox is well-documented, open-source and it is available under the GPL3 license. In this paper, we give an overview of the algorithms contained in the toolbox and discuss their properties.**

I. Introduction

In many STFT-based audio processing applications, the direct synthesis from the STFT magnitude (or spectrogram) is unavoidable. Since the missing information is the phase of the complex coefficients, the task is also often called phase retrieval or phase reconstruction. To date, many algorithms have been proposed, but they have mostly been published without reference implementation or means for reproducing the results. Moreover, the algorithms have often been formulated and/or evaluated for a single STFT setting using a fixed window for the analysis and synthesis. In this paper, we present the Phase Retrieval Toolbox (PHASERET), which tries to rid the user of all hindrances mentioned above and others connected to implementation and evaluation of the algorithms. The phase reconstruction algorithms can be divided into two main classes. Algorithms in the first class require the knowledge of the magnitude component of the entire signal (offline only) while algorithms in the second class are capable of running in the real-time setting i.e. to process STFT frames in the frame-by-frame manner. Therefore, the paper is divided into two parts each devoted to one of the classes. The acronyms in the typewriter font used in the titles and in the text refer to the actual function names from the toolbox. For a detailed description of the functions, please refer to the documentation.

A. Toolbox Organization

The toolbox is mostly written in the Matlab scripting language compatible with GNU Octave. The main computational functions are written in the C language for greater speed and they are accessed trough MEX functions. The toolbox depends on the Large Time-Frequency Analysis Toolbox [1, 2] (LTFAT) and adopts its conventions and the GPLv3 license. The documentation is available at http://ltfat.github.io/phaseret/doc and the GitHub development page containing GIT repository, release packages and the issue tracker is available at http://github.com/ltfat/phaseret.

Z. Průša* is with the Acoustics Research Institute, Austrian Academy of Sciences, Wohllebengasse 12–14, 1040 Vienna, Austria, email: zdenek.prusa@oeaw.ac.at (corresponding address),

This work was supported by the Austrian Science Fund (FWF): Y 551–N13

II. Offline algorithms

In this section, we will only consider signals of finite length L with periodic boundaries. The STFT of such signals is also called the (finite) discrete Gabor transform (DGT) [3] and we will adopt such name in this paper. The DGT coefficient matrix $\mathbf{c} \in \mathbb{C}^{M\times N}$ of a signal $\mathbf{f} \in \mathbb{C}^L$ with respect to a window $\mathbf{g} \in \mathbb{C}^L$ can be obtained as [4]

$$\mathbf{c}(m,n) = \sum_{l=0}^{L-1} \mathbf{f}(l+na)\overline{\mathbf{g}(l)}\mathrm{e}^{-\mathrm{i}2\pi ml/M} \tag{1}$$

for $m = 0,\dots,M-1$ and $n = 0,\dots,N-1$, M is the number of frequency channels, $N = L/a$ number of time shifts, a is the hop size in samples. The overline denotes complex conjugation. Index $(l+na)$ is assumed to be evaluated modulo L; effectively introducing the aforementioned periodic boundaries. In other words, the signal $\mathbf{f}$ and the window $\mathbf{g}$ are considered to be single periods of L-periodic signals. Note that in order not to introduce undesirable effects caused by the phase of the window itself, we assume the original, nonperiodized window to be whole-point symmetric and centered around the origin. The complex coefficients can be expressed in polar coordinates as

$$\mathbf{s}(m,n) = \left|\mathbf{c}(m,n)\right| \quad \phi(m,n) = \arg\left(\mathbf{c}(m,n)\right), \tag{2}$$

$\mathbf{s}$ denoting magnitude and ϕ denoting their phase. The log of the magnitude $\mathbf{s}_{\log}(m,n) = \log(\mathbf{s}(m,n))$ is usually visualised in the form of a spectrogram. Using matrices, we can write $\mathbf{c}_{\text{vec}} = \mathbf{F}_{\mathbf{g}}^*\mathbf{f}$, where $\mathbf{c}_{\text{vec}} \in \mathbb{C}^{MN}$ denotes vectorized $\mathbf{c}$ such that $\mathbf{c}_{\text{vec}}(p) = \mathbf{c}(m,n)$ for $p = m + nM$ and $\mathbf{F}_{\mathbf{g}}^* \in \mathbb{C}^{MN\times L}$ is a conjugate transpose of a "fat" ($MN > L$) matrix $\mathbf{F}_{\mathbf{g}}$ with the columns in the form of modulated and circularly shifted versions of $\mathbf{g}$ (see Fig. 1)

$$\mathbf{g}_{m,n}(l) = \mathbf{g}(l-na)\mathrm{e}^{\mathrm{i}2\pi m(l-na)/M},$$

where $(l-na)$ is again evaluated modulo L. Equation (1) can be therefore equivalently expressed using inner products

$$\mathbf{c}(m,n) = \left\langle \mathbf{f}, \mathbf{g}_{m,n}\right\rangle \tag{3}$$

$$= \sum_{l=0}^{L-1} \mathbf{f}(l)\overline{\mathbf{g}(l-na)}\mathrm{e}^{-\mathrm{i}2\pi m(l-na)/M}. \tag{4}$$

As long as the matrix $\mathbf{F}_{\mathbf{g}}$ has full row rank (linearly independent rows), the signal can be recovered using $\mathbf{f} = \mathbf{F}_{\widetilde{\mathbf{g}}}\mathbf{c}_{\text{vec}}$ where $\mathbf{F}_{\widetilde{\mathbf{g}}} = \left(\mathbf{F}_{\mathbf{g}}\mathbf{F}_{\mathbf{g}}^*\right)^{-1}\mathbf{F}_{\mathbf{g}}$ is the left inverse of $\mathbf{F}_{\mathbf{g}}^*$ such that $\mathbf{F}_{\widetilde{\mathbf{g}}}\mathbf{F}_{\mathbf{g}}^* = \mathbf{F}_{\mathbf{g}}\mathbf{F}_{\widetilde{\mathbf{g}}}^* = \mathbf{I}$. The fundamental property of invertible Gabor systems is that $\mathbf{F}_{\widetilde{\mathbf{g}}}$ has the same structure as $\mathbf{F}_{\mathbf{g}}$ with $\widetilde{\mathbf{g}}$ being the synthesis window, defined by

$$\widetilde{\mathbf{g}} = \left(\mathbf{F}_{\mathbf{g}}\mathbf{F}_{\mathbf{g}}^*\right)^{-1}\mathbf{g}. \tag{5}$$

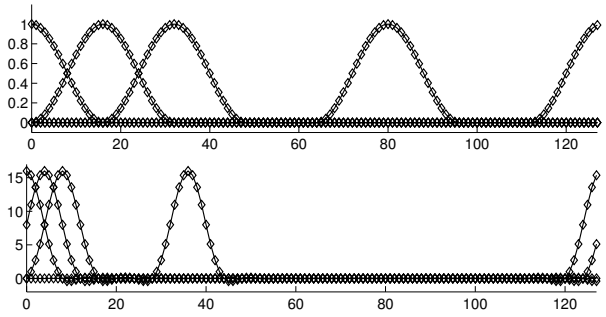

Fig. 1: Examples of $\mathbf{g}_{m,n}$ using Hann window (32 samples long) and $a = 16, M = 32$ and $L = 128$. Circular time shifts $\mathbf{g}_{0,n}$ for $n \in \{0, 1, 2, 5\}$ in the time domain (upper) and modulations $\mathbf{g}_{m,0}$ for $m \in \{0, 1, 2, 9\}$ in the frequency domain (lower).

Listing 1: Code example of a typical workflow

```
% Load the glockenspiel test signal (262144
    samples)
% Sampling rate fs = 44100
[f,fs] = gspi;
a = 256;  % Time step
M = 2048; % Number of frequency channels
g = 'blackman';     % Blackman window
gtilde = {'dual',g}; % Synthesis window
% Compute the DGT coefficients c, Ls=numel(f)
[c,Ls] = dgtreal(f,g,a,M,'timeinv');
% Obtain the magnitude
s = abs(c);
% Reconstruct the phase
chat = phase_rec_func(s,...
% Reconstruct the signal
fhat = idgtreal(chat,gtilde,a,M,Ls,'timeinv');
```

Note that the only requirement is the invertibility of $\mathbf{F_g F_g^*}$. In particular, it allows the support of the window $\mathbf{g}$ to be as long as the signal even when the number of frequency channels M is significantly lower than the signal length L.

The LTFAT toolbox contain implementation of fast algorithms for computing the equations mentioned in this section, therefore no matrix is explicitly inverted or created. Depending on the length of the support of the window, the computational functions employ either the classical Portnoff algorithm [5] or the algorithm based on the Walnut factorization of the matrices [6]. In LTFAT, an implementation of (1) is available as functions `dgt` and `dgtreal`, where the latter one accepts only real input signals and returns only half of the coefficient. The condition number of the matrix inverted in (5) can be obtained from `gabframebounds` and the synthesis window itself from `gabdual`. Functions doing the inverse transform are called `idgt` and `idgtreal` respectively. Note that it is necessary to pass additional `'timeinv'` flag to all transform functions in order to follow (1) exactly. See http://ltfat.github.io/doc for more details.

Listing 1 shows how to obtain the DGT coefficients and their magnitude from a signal and how to reconstruct it using functions from LTFAT. The function phase_rec_func is only used as a placeholder and it can be replaced by any function doing phase reconstruction mentioned in this paper.

A. *Griffin-Lim Algorithm – `gla`*

The Griffin-Lim algorithm [7] (GLA) is arguably the most generic algorithm for phase reconstruction. It proceeds by projecting the coefficients iteratively onto two subsets of $\mathbb{C}^{MN}$ denoted as $\mathcal{C}_1$ and $\mathcal{C}_2$. The set $\mathcal{C}_1$ is identical with the range (column) space of $\mathbf{F_g^*}$ i.e.

$$\mathcal{C}_1 = \left\{ \mathbf{c}_{\text{vec}} | \mathbf{c}_{\text{vec}} = \mathbf{F_g^*} \mathbf{f} \quad \text{for all} \quad \mathbf{f} \in \mathbb{C}^L \right\}. \tag{6}$$

Its orthogonal complement is the null space of $\mathbf{F_g}$ defined as

$$\mathcal{C}_1^{\perp} = \left\{ \mathbf{c}_{\text{vec}} | \mathbf{F_g} \mathbf{c}_{\text{vec}} = 0 \right\}. \tag{7}$$

Any $\mathbf{c}_{\text{vec}} \in \mathbb{C}^{MN}$ can be written as a sum of components in these two spaces as

$$\mathbf{c}_{\text{vec}} = \mathbf{c}_{\text{vec}}^{\mathcal{C}_1} + \mathbf{c}_{\text{vec}}^{\mathcal{C}_1^{\perp}} \tag{8}$$

and there exists an unique orthogonal projection onto $\mathcal{C}_1$ (see e.g. [8])

$$\mathbf{P}_{\mathcal{C}_1} \mathbf{c}_{\text{vec}} = \mathbf{F_g^*} \left(\mathbf{F_g F_g^*} \right)^{-1} \mathbf{F_g} \mathbf{c}_{\text{vec}} = \mathbf{F_g^*} \mathbf{F}_{\widetilde{\mathbf{g}}} \mathbf{c}_{\text{vec}} = \mathbf{c}_{\text{vec}}^{\mathcal{C}_1}, \tag{9}$$

which amounts to synthesis using the window $\widetilde{\mathbf{g}}$ obtained by (5) followed by analysis using the original window $\mathbf{g}$. The magnitude of $\mathbf{c}_{\text{vec}}^{\mathcal{C}_1}$ is also often called a valid or *consistent* spectrogram [9] since it is guaranteed that these exists a signal with such spectrogram. The set $\mathcal{C}_2$ is a set of coefficients with magnitude equal to the target magnitude $\mathbf{s}_{\text{vec}}$

$$\mathcal{C}_2 = \left\{ \mathbf{c}_{\text{vec}} | \left| \mathbf{c}_{\text{vec}}(p) \right| = \mathbf{s}_{\text{vec}}(p) \right\}. \tag{10}$$

Since it is not a convex set, the projection is substituted simply by forcing the magnitude of the coefficients to the target magnitude

$$\left(\mathbf{P}_{\mathcal{C}_2} \mathbf{c}_{\text{vec}} \right)(p) = \mathbf{s}_{\text{vec}}(p) \mathbf{c}_{\text{vec}}(p) / \left| \mathbf{c}_{\text{vec}}(p) \right|. \tag{11}$$

The composition of the projections then gives i-th iteration of GLA

$$\mathbf{c}_{\text{vec},i} = \mathbf{P}_{\mathcal{C}_2} \mathbf{P}_{\mathcal{C}_1} \mathbf{c}_{\text{vec},i-1}. \tag{12}$$

given $\mathbf{c}_{\text{vec},0}(p) = \mathbf{s}_{\text{vec}}(p) \mathrm{e}^{\mathrm{i}\phi_{\text{vec},0}(p)}$ for all p, where $\mathbf{s}_{\text{vec}}$ is the target magnitude and $\phi_{\text{vec},0}$ is the initial phase. A common approach is to choose zeros or random values, but it has been reported that the performance of the algorithm can be improved significantly by choosing suitable initial phase [10]. The algorithm has been also used for other transforms as well [11] and, furthermore, the nature of the algorithm allows creating ad-hoc algorithms by imposing additional constraints in time or time-frequency domains [12, 13, 14].

Perraudin et al. [15] proposed a fast version of GLA (FGLA) by introducing an acceleration step which combines the current and the previous iterations with weight α_k as shown in Alg. 1. The optimal sequence α_i for the FGLA remains an open question. Constant $\alpha_i = 0.99$ was suggested by the authors. Note that $\alpha_i = 0$ reverts the algorithm back to the regular GLA.

Algorithm 1: (Fast) Griffin-Lim algorithm

Input: Initial coefficients $\mathbf{c}_{\text{vec},0}$, initial α_0.
Output: Coefficients with new phase $\mathbf{c}_{\text{vec},i}$.

$\mathbf{t}_{\text{vec},0} = \mathbf{c}_{\text{vec},0}$;
for $i = 1, 2, \ldots$ **do**
  $\mathbf{t}_{\text{vec},i} = \mathbf{P}_{\mathcal{C}_2}\mathbf{P}_{\mathcal{C}_1}\mathbf{c}_{\text{vec},i-1}$;
  $\mathbf{c}_{\text{vec},i} = \mathbf{t}_{\text{vec},i} + \alpha_{i-1}(\mathbf{t}_{\text{vec},i} - \mathbf{t}_{\text{vec},i-1})$;
  Update α_i;
end

B. Le Roux's Modifications of GLA – `legla`

Le Roux et al. [9] proposed several modifications of the Griffin-Lim algorithm. First, they recognized a structure in the projection (9) and proposed its approximation using a truncated kernel. Second, since this way the projection (9) can be done coefficient-wise and so can be done (11), the authors proposed to do *on-the-fly* updates i.e. to reuse the just updated coefficient for computing others within the same iteration. Third, they proposed a *modified* phase update (omitting contribution of the coefficient being updated). The projection (9) can be written in a form of (non-commutative) circular twisted convolution [16]

$$\begin{aligned}&\left(\mathbf{c}_{\text{vec}} \natural \mathbf{h}_{\text{vec}}\right)(m + nM) = \\ &\sum_{j,k=0}^{M-1,N-1} \mathbf{c}(j,k)\mathbf{h}(m-j, n-k)\mathrm{e}^{\mathrm{i}2\pi(n-k)ja/M}\end{aligned} \tag{13}$$

where the kernel is obtained as $\mathbf{h}_{\text{vec}} = \mathbf{F}_{\mathbf{g}}^{*}\widetilde{\mathbf{g}}$. Clearly, there is only $\operatorname{lcm}(a, M)/a$ (least common multiple) unique kernel modulations which can be precomputed. Typically, the kernel $\mathbf{h}$ is essentially supported around the origin and it can be truncated with only a minor loss of precision. Further, the kernel is conjugate symmetric, which can be exploited to reduce the number of multiplications. Note that the *modified* phase update can be obtained simply by setting $\mathbf{h}(0,0) = 0$. The algorithm is summarized as Alg. 2. Note that the acceleration technique from [15] can be used here as well.

Algorithm 2: Le Roux's version of GLA

Input: Initial coefficients $\mathbf{c}_{\text{vec},1}$.
Output: Coefficients with new phase $\mathbf{c}_{\text{vec},i}$.

Initialize and order array $\mathcal{I}$ of indices p to be processed;
for $i = 1, 2, \ldots$ **do**
  for *all* $p \in \mathcal{I}$ **do**
    $\mathbf{c}_{\text{vec},i}(p) \leftarrow$
    $\mathbf{s}_{\text{vec}}(p)\left(\mathbf{c}_{\text{vec},i} \natural \mathbf{h}_{\text{vec}}\right)(p) / \left|\left(\mathbf{c}_{\text{vec},i} \natural \mathbf{h}_{\text{vec}}\right)(p)\right|$;
  end
end

C. Unconstrained Optimization – `decolbfgs`

Decorsiere et al. [17] proposed to take a direct optimization approach. Writing the objective function as

$$\mathcal{G}(\mathbf{f}) = \left\| \left|\mathbf{F}_{\mathbf{g}}^{*}\mathbf{f}\right|^{q} - \mathbf{s}_{\text{vec}}^{q} \right\|^{2} \tag{14}$$

(the q-th power is performed elementwise) and expressing its gradient explicitly as

$$\begin{aligned}&\nabla\mathcal{G}(\mathbf{f}) = \\ &2q\mathcal{R}\left(\mathbf{F}_{\mathbf{g}}\left[\left(\left|\mathbf{F}_{\mathbf{g}}^{*}\mathbf{f}\right|^{q} - \mathbf{s}_{\text{vec}}^{q}\right) \cdot \left|\mathbf{F}_{\mathbf{g}}^{*}\mathbf{f}\right|^{\frac{q}{2}-1} \cdot \mathbf{F}_{\mathbf{g}}^{*}\mathbf{f}\right]\right)\end{aligned} \tag{15}$$

($\mathcal{R}$ denotes the real-part operator) allows usage of generic algorithms for unconstrained minimization. The authors suggest the l-BFGS algorithm as it does not require explicit second order derivatives. Further, they suggested to pick $q = 2/3$. Note that the function `decolbfgs` requires the implementation of lBFGS from the minFunc package [18].

D. Phase Gradient Heap Integration – `pghi`

Průša et al. [10] proposed a phase reconstruction method based on the relationship between the magnitude and the phase gradients. Assuming the Gaussian window is used, the phase gradient can be estimated using the magnitude gradient and the phase is then recovered by employing an adaptive integration procedure. The estimate of the phase derivative in the frequency direction $\phi_\omega(m,n)$ and in the time direction $\phi_t(m,n)$ expressed solely from the magnitude (pre-scaled for the integration) can be written as

$$\begin{aligned}&\phi_\omega(m,n) = \\ &-\frac{\gamma}{2aM}\left(\mathbf{s}_{\log}(m,n+1) - \mathbf{s}_{\log}(m,n-1)\right) \\ &\phi_t(m,n) = \\ &\frac{aM}{2\gamma}\left(\mathbf{s}_{\log}(m+1,n) - \mathbf{s}_{\log}(m-1,n)\right) + 2\pi am/M\end{aligned} \tag{16}$$

with γ being the "time-frequency" ratio of the Gaussian window. Given phase at a single point $\phi(m,n)$, the phase of its four neighbors is computed as

$$\begin{aligned}\phi(m+1,n) &\leftarrow \phi(m,n) + \frac{1}{2}\left(\phi_\omega(m,n) + \phi_\omega(m+1,n)\right), \\ \phi(m,n+1) &\leftarrow \phi(m,n) + \frac{1}{2}\left(\phi_t(m,n) + \phi_t(m,n+1)\right), \\ \phi(m-1,n) &\leftarrow \phi(m,n) - \frac{1}{2}\left(\phi_\omega(m,n) + \phi_\omega(m-1,n)\right), \\ \phi(m,n-1) &\leftarrow \phi(m,n) - \frac{1}{2}\left(\phi_t(m,n) + \phi_t(m,n-1)\right).\end{aligned}$$

Further phase spreading is guided by the coefficient magnitude such that the phase of significant coefficients along spectrogram ridges is computed first. For non-Gaussian windows, the approximate γ can be obtained from `pghi_findgamma`.

III. Real-time algorithms

When dealing with real-time algorithms, the finite length signal model from the previous section is no longer suitable. In order to model streams of audio data, it is a common practice to work with infinite length discrete time signals. In this section, we will reformulate STFT for such signals and use it in the description of the algorithms. Given a discrete time signal f

and a compactly supported window g, the STFT at time index na denoted as c_n is given by

$$c_n(m) = \sum_{l \in \mathcal{I}} f(l + na)\overline{g(l)}\mathrm{e}^{-\mathrm{i}2\pi ml/M} \tag{17}$$

for $m = 0, \ldots, M-1$ and any $n \in \mathbb{Z}$ and where $\mathcal{I}$ denotes the compact support of the window g. We again assume the window is symmetric around the origin. We will further also use the following operator notation

$$\left(\mathcal{V}_g f\right)_n = c_n \tag{18}$$

and denote the magnitude of the coefficients of the n-th frame as s_n, the logarithm of the coefficients as $s_{\log,n}$ and the phase as ϕ_n. An individual *time frame* f_n can be reconstructed using

$$f_n(l) = \widetilde{g}(l) \sum_{m=0}^{M-1} c_n(m)\mathrm{e}^{\mathrm{i}2\pi ml/M} \tag{19}$$

for $l \in \mathcal{I}$ and zero elsewhere. If $a < \mathrm{len}(g) \leq M$ ($\mathrm{len}(g)$ being the length of the window support), the synthesis window $\widetilde{g}$ is given by

$$\widetilde{g}(l) = \frac{1}{M} \frac{g(l)}{\sum_{n \in \mathbb{Z}} g(l - na)^2}, \tag{20}$$

for $l \in \mathcal{I}$ and is zero elsewhere. Partially reconstructed signal up to frame N is given by

$$\widetilde{f}_N(l) = \sum_{n=-\infty}^{N} f_n(l - na). \tag{21}$$

In the following text, the function names will refer to the offline implementation of the algorithms. The true real-time implementation of the algorithms is available in `demo_blockproc_phaseret` and `demo_blockproc_phaseret2`.

A. Single Pass Spectrogram Inversion – `spsi`

Beauregard et al. [19] proposed a phase vocoder-like technique for phase reconstruction. It employs peak picking followed by quadratic interpolation of the magnitude in order to precisely identify the instantaneous frequency. The inst. frequency gives the increment of the peak phase between two frames. The peak phase is also assigned to all coefficients within its region of influence. The steps of the algorithm are summarized in Alg. 3.

B. Real-Time Phase Gradient Heap Integration – `rtpghi`

Průša and Søndergaard [20] proposed an adaptation of the PGHI algorithm to the real-time setting (RTPGHI). The core of the algorithm was kept, with the exception that the phase is spread only from the previous to the current frame. From (16), it is clear that the algorithm requires access to $s_{\log,n+1}$ i.e. it requires one look-ahead frame. Employing a causal differentiation scheme yields even zero look-ahead frames.

Algorithm 3: SPSI

Input: Magnitude of n-th frame s_n, phase of the previous frame ϕ_{n-1}.
Output: Phase of the current frame ϕ_n.

Identify peaks $s_n(m_{\text{peak}})$ and their regions of influence $\text{ROI}_{m_{\text{peak}}}$.;
for *all* m_{peak} **do**
  $m_{\text{truepeak}} \leftarrow m_{\text{peak}} + \frac{1}{2}\frac{s_{\log,n}(m_{\text{peak}}-1) - s_{\log,n}(m_{\text{peak}}+1)}{s_{\log,n}(m_{\text{peak}}-1) - 2s_{\log,n}(m_{\text{peak}}) + s_{\log,n}(m_{\text{peak}}+1)}$;
  $\phi_n(m_{\text{peak}}) \leftarrow \phi_{n-1}(m_{\text{peak}}) + 2\pi a m_{\text{truepeak}}/M$;
  $\phi_n(m) \leftarrow \phi_n(m_{\text{peak}})$ for all $m \in \text{ROI}_{m_{\text{peak}}}$;
end

C. Real-Time Iterative Spectrogram Inversion With Look-Ahead – `rtisila`

Zhu et al. [21] proposed a real-time iterative algorithm based on the idea of GLA. It employs N_{LA} look-ahead frames and proceeds by doing GLA-type iterations on individual frames in the order from the most recent one to the current one and repeats this procedure for a pre-determined number of iterations. The fundamental problem of the approach is the inherent asymmetry introduced by windowing the partially reconstructed signal. In order to compensate this problem, the authors proposed to use two additional asymmetric analysis windows $g_{N_{\text{LA}},-1}$ and $g_{N_{\text{LA}},0}$ to be used with the most recent look-ahead frame and to use the regular analysis window for other frames. The additional windows are obtained as

$$g_{N_{\text{LA}},-1}(l) = M \sum_{n=-N_{\text{LB}}}^{-1} g(na - l)\widetilde{g}(na - l), \tag{22}$$

$$g_{N_{\text{LA}},0}(l) = M \sum_{n=-N_{\text{LB}}}^{0} g(na - l)\widetilde{g}(na - l), \tag{23}$$

for $l \in \mathcal{I}$ and zero elsewhere, where $N_{\text{LB}} = \left\lceil \mathrm{len}(g)/a \right\rceil - 1$ is the number of "look-back" frames. Window $g_{N_{\text{LA}},-1}$ is only used in the very first iteration, while $g_{N_{\text{LA}},0}$ is used for all the following iterations. In the algorithm listing in Alg. 4 the analysis windows g_k are equal to g except for $k = N_{\text{LA}}$ for which the above applies.

D. Gnann and Spiertz's RTISI-LA – `gsrtisila`

Gnann and Spiertz [22] proposed an alternative way of dealing with the asymmetry of the partially reconstructed signal. They proposed to use $N_{\text{LA}} + 1$ additional analysis windows $g_0, \ldots, g_{N_{\text{LA}}}$ computed as

$$g_k(l) = M \frac{g(l)}{g_{\text{sum}}(l + ka)} \tag{24}$$

for $l \in \mathcal{I}$ and zero elsewhere and

$$g_{\text{sum}}(l) = \sum_{n=-N_{\text{LB}}}^{N_{\text{LA}}} g(l - na)\widetilde{g}(l - na). \tag{25}$$

For this version of RTISI-LA, the listing in Alg. 4 applies without changes.

A combination of GSRTISI-LA with RTPGHI proved to give excellent results [23].

Algorithm 4: (GS)RTISI-LA

Input: Number of look-ahead frames N_{LA}, magnitude of the current and the look-ahead frames $s_n, \ldots, s_{n+N_{\mathrm{LA}}}$

Output: Time frame f_n and its coefficients c_n.

Initialize $f_{n+N_{\mathrm{LA}}}$ to zeros.;
for $i = 1, 2, \ldots$ **do**
  for $k = N_{\mathrm{LA}}, \ldots, 0$ **do**
    Compute $\widetilde{f}_{n+N_{\mathrm{LA}}}$ using (21).;
    $t \leftarrow \left(\mathcal{V}_{g_k} \widetilde{f}_{n+N_{\mathrm{LA}}}\right)_{n+k}$;
    $c_{n+k}(m) \leftarrow s_{n+k}(m)\, t(m) / \left|t(m)\right|$;
    Compute f_{n+k} using (19);
  end
end

IV. Other algorithms

The toolbox is primarily focused on processing audio signals which consist of tens of thousands of samples per second. We have intentionally not included implementation of algorithms which cannot efficiently handle signals at least few seconds long or which have some other serious drawback. In this section, we mention other approaches and give a reason why they are not practical for processing audio signals.

A popular approach is reformulating the problem as a convex, matrix rank minimization problem [24, 25, 26]. Doing so however squares the size of the original problem and only very short signals can be handled efficiently. The authors themselves evaluate the algorithms on signals shorter than 100 samples.

An approach by Bouvrie and Ezzat [27] is based on solving a non-linear system of equations for each time frame. It however relies on using a rectangular window for analysis which is known to have bad frequency selectivity.

Worth mentioning is an algorithm by Eldar et. al. [28] which is only applicable to signals which are sparse in the original domain. Such assumption is not realistic when considering real world audio signals.

V. Acknowledgement

This work was supported by the Austrian Science Fund (FWF): Y 551–N13.

References

[1] Søndergaard, P. L., Torrésani, B., and Balazs, P., "The Linear Time Frequency Analysis Toolbox," *International Journal of Wavelets, Multiresolution Analysis and Information Processing*, 10(4), pp. 1–27, 2012, doi: 10.1142/S0219691312500324.

[2] Průša, Z., Søndergaard, P. L., Holighaus, N., Wiesmeyr, C., and Balazs, P., "The Large Time-Frequency Analysis Toolbox 2.0," in *Sound, Music, and Motion*, Lecture Notes in Computer Science, pp. 419–442, Springer International Publishing, 2014, ISBN 978-3-319-12975-4, doi:{10.1007/978-3-319-12976-1_25}.

[3] Søndergaard, P. L., *Finite discrete Gabor analysis*, Ph.D. thesis, Technical University of Denmark, 2007, available from: http://ltfat.github.io/notes/ltfatnote003.pdf.

[4] Søndergaard, P. L., "Gabor frames by Sampling and Periodization," *Adv. Comput. Math.*, 27(4), pp. 355 –373, 2007, doi:10.1007/s10444-005-9003-y.

[5] Portnoff, M. R., "Implementation of the digital phase vocoder using the fast Fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(3), pp. 243–248, 1976, ISSN 0096-3518, doi:10.1109/TASSP.1976.1162810.

[6] Søndergaard, P. L., "Efficient Algorithms for the Discrete Gabor Transform with a long FIR window," *J. Fourier Anal. Appl.*, 18(3), pp. 456–470, 2012, doi: 10.1007/s00041-011-9210-5.

[7] Griffin, D. and Lim, J., "Signal estimation from modified short-time Fourier transform," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 32(2), pp. 236–243, 1984, ISSN 0096-3518, doi:10.1109/TASSP.1984.1164317.

[8] Strang, G., *Introduction to Linear Algebra*, Wellesley-Cambridge Press, 5 edition, 2016, ISBN 978-09802327-7-6.

[9] Le Roux, J., Kameoka, H., Ono, N., and Sagayama, S., "Fast Signal Reconstruction from Magnitude STFT Spectrogram Based on Spectrogram Consistency," in *Proc. 13th Int. Conf. on Digital Audio Effects (DAFx-10)*, pp. 397–403, 2010.

[10] Průša, Z., Balazs, P., and Søndergaard, P. L., "A Noniterative Method for Reconstruction of Phase from STFT Magnitude," *IEEE/ACM Trans. on Audio, Speech, and Lang. Process.*, 25(5), 2017, doi:10.1109/TASLP.2017.2678166.

[11] Nakamura, T. and Kameoka, H., "Fast signal reconstruction from magnitude spectrogram of continuous wavelet transform based on spectrogram consistency," *Proc. 17th Int. Conf. Digital Audio Effects DAFx (DAFx-14)*, 2014, pp. 129–135, 2014.

[12] Dittmar, C. and Müller, M., "Towards transient restoration in score-informed audio decomposition," *Proc. 18th Int. Conf. Digital Audio Effects DAFx (DAFx-15)*, 2015, pp. 145–152, 2015.

[13] Sturmel, N. and Daudet, L., "Iterative phase reconstruction of Wiener filtered signals," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pp. 101–104, 2012, ISSN 1520-6149, doi: 10.1109/ICASSP.2012.6287827.

[14] Sturmel, N. and Daudet, L., "Informed Source Separation Using Iterative Reconstruction," *IEEE Transactions on Audio, Speech, and Language Processing*, 21(1), pp. 178–185, 2013, ISSN 1558-7916, doi:10.1109/TASL.2012.2215597.

[15] Perraudin, N., Balazs, P., and Søndergaard, P. L., "A fast Griffin-Lim algorithm," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), IEEE Workshop on*, pp. 1–4, 2013, ISSN 1931-1168, doi: 10.1109/WASPAA.2013.6701851.

[16] Werther, T., Matusiak, E., Eldar, Y. C., and Subbana, N. K., "A Unified Approach to Dual Gabor Windows," *IEEE*

Transactions on Signal Processing, 55(5), pp. 1758–1768, 2007, ISSN 1053-587X, doi:10.1109/TSP.2006.890908.

[17] Decorsiere, R., Søndergaard, P. L., MacDonald, E. N., and Dau, T., "Inversion of Auditory Spectrograms, Traditional Spectrograms, and Other Envelope Representations," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 23(1), pp. 46–56, 2015, ISSN 2329-9290, doi:10.1109/TASLP.2014.2367821.

[18] Schmidt, M., "minFunc: Unconstrained differentiable multivariate optimization in Matlab," 2005, available at: http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html.

[19] Beauregard, G. T., Harish, M., and Wyse, L., "Single Pass Spectrogram Inversion," in *Digital Signal Processing (DSP), IEEE International Conference on*, pp. 427–431, 2015, doi:10.1109/ICDSP.2015.7251907.

[20] Průša, Z. and Søndergaard, P. L., "Real-Time Spectrogram Inversion Using Phase Gradient Heap Integration," in *Proc. Int. Conf. Digital Audio Effects (DAFx-16)*, pp. 17–21, 2016.

[21] Zhu, X., Beauregard, G. T., and Wyse, L., "Real-Time Signal Estimation From Modified Short-Time Fourier Transform Magnitude Spectra," *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(5), pp. 1645–1653, 2007, ISSN 1558-7916, doi:10.1109/TASL.2007.899236.

[22] Gnann, V. and Spiertz, M., "Improving RTISI Phase Estimation with Energy Order and Phase Unwrapping," in *Proc. 13th Int. Conf. Digital Audio Effects (DAFx-10)*, pp. 1–5, 2010.

[23] Průša, Z. and Rajmic, P., "Towards High Quality Real-Time Signal Reconstruction from STFT Magnitude," *IEEE Signal Processing Letters*, 2017, doi:10.1109/LSP.2017.2696970.

[24] Balan, R., "On signal reconstruction from its spectrogram," in *Information Sciences and Systems (CISS), 44th Annual Conference on*, pp. 1–4, 2010, doi:10.1109/CISS.2010.5464828.

[25] Sun, D. L. and Smith, J. O., III, "Estimating a Signal from a Magnitude Spectrogram via Convex Optimization," in *Audio Engineering Society Convention 133*, 2012.

[26] Jaganathan, K., Eldar, Y. C., and Hassibi, B., "Recovering signals from the Short-Time Fourier Transform magnitude," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3277–3281, 2015, ISSN 1520-6149, doi:10.1109/ICASSP.2015.7178577.

[27] Bouvrie, J. V. and Ezzat, T., "An incremental algorithm for signal reconstruction from short-time Fourier transform magnitude." in *Spoken Language Processing (INTERSPEECH), 9th International Conference on*, ISCA, 2006.

[28] Eldar, Y., Sidorenko, P., Mixon, D., Barel, S., and Cohen, O., "Sparse Phase Retrieval from Short-Time Fourier Measurements," *IEEE Signal Processing Letters*, 22(5), pp. 638–642, 2015, ISSN 1070-9908, doi:10.1109/LSP.2014.2364225.