A Non-iterative Method for Reconstruction of Phase from STFT Magnitude

Zdeněk Průša, Peter Balazs, *Senior Member, IEEE,* and Peter L. Søndergaard

***Abstract*—A non-iterative method for the construction of the Short-Time Fourier Transform (STFT) phase from the magnitude is presented. The method is based on the direct relationship between the partial derivatives of the phase and the logarithm of the magnitude of the un-sampled STFT with respect to the Gaussian window. Although the theory holds in the continuous setting only, the experiments show that the algorithm performs well even in the discretized setting (Discrete Gabor transform) with low redundancy using the sampled Gaussian window, the truncated Gaussian window and even other compactly supported windows like the Hann window. Due to the non-iterative nature, the algorithm is very fast and it is suitable for long audio signals. Moreover, solutions of iterative phase reconstruction algorithms can be improved considerably by initializing them with the phase estimate provided by the present algorithm. We present an extensive comparison with the state-of-the-art algorithms in a reproducible manner.**

***Index Terms*—STFT, Gabor transform, Phase reconstruction, Gradient theorem, Numerical integration**

I. Introduction

The phase retrieval problem has been actively investigated for decades. It was first formulated for the Fourier transform [1] and later for generic linear systems [2]. In this paper, we consider a particular case of the phase retrieval problem; the reconstruction from the magnitude of the Gabor transform coefficients obtained by sampling the STFT magnitude at discrete time and frequency points [3]. The need for an effective way to reconstruct the phase arises in audio processing applications such as source separation and denoising [4], [5], time-stretching/pitch shifting [6], channel mixing [7], and missing data imputation [8].

The problem has already been addressed by many authors. Among the iterative algorithms, the most widespread and influential is the algorithm introduced by Griffin and Lim [9] (GLA) which inspired several extensions [10], [11] (FleGLA) and [12], [13] (TF-RTISI-LA). See [14] for a detailed overview of the algorithms based on the idea of Griffin and Lim. A different approach was taken by Decorsiere et al. [15] (lBFGS). They expressed the problem as an unconstrained optimization problem and solve it using the limited memory Broyden-Flatcher-Goldfarb-Shanno algorithm. It is again an iterative algorithm and the computational cost of a single iteration is comparable to that of GLA.

Other approaches are based on reformulating the task as a convex optimization problem [16], [17], [18], [19]. The dimension of the problem however squares, which makes it unsuitable for long audio signals which typically consist of tens of thousands of samples per second. Eldar et al. [20] assume the signal to be sparse in the original domain, which is not realistic in the context of the audio processing applications mentioned above. An approach presented by Bouvrie and Ezzat [21] is based on solving a non-linear system of equations for each time frame. The authors proposed to use an iterative solver and initialize it with samples obtained from previous frames. The algorithm is, however, designed to work exclusively with a rectangular window, which is known to have bad frequency selectivity.

The common problem of the iterative state-of-the-art algorithms is that they require many relatively expensive iterations in order to produce acceptable results. A non-iterative algorithm proposed by Beauregard et al. [22] (SPSI) is based on the notion of *phase consistency* used in the phase vocoder [6]. Although the algorithm is simple, fast and it is directly suitable for the real-time setting, it relies on the fact that the signal consists of slowly varying sinusoidal components and fails for transients and broadband components in general. A similar algorithm was introduced by Magron et al. [23] (PU), which, in addition, tries to treat impulse-like components separately.

In this paper, we propose a non-iterative algorithm (Phase Gradient Heap Integration – PGHI). The theory behind PGHI has been known at least since 1979. It is based on the relationship between gradients of the Gaussian window-based STFT phase and log-magnitude first presented by Portnoff [24] and on the *gradient theorem*. More precisely, the phase gradient can be expressed using the STFT magnitude and the gradient theorem gives a prescription how to integrate the phase gradient field to recover the phase up to a global phase shift (or up to sign ambiguity in case of real signals). To our knowledge, no such algorithm has been published yet. In our previous work [25], we have presented a special case of PGHI adapted to the real-time setting. The present paper focuses on providing a complete mathematical treatment and on a thorough comparison with other algorithms in the offline setting. The aforementioned algorithms SPSI and PU are in fact close to the PGHI algorithm since they basically perform a simple integration of the estimate of *instantaneous frequency* and of the *local group delay* in case of PU, which are components of the STFT phase gradient. Their approach

Manuscript received April 19, 2005; revised August 26, 2015.

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

P. L. Søndergaard is with Oticon A/S, Kongebakken 9, 2765 Smørum, Denmark, email: peter@sonderport.dk

This work was supported by the Austrian Science Fund (FWF) START-project FLAME ("Frames and Linear Operators for Acoustical Modeling and Parameter Estimation"; Y 551-N13).

however cannot estimate the gradient at every time-frequency position and the estimation requires analysing the spectrogram content.

In the spirit of reproducible research, the implementation of the algorithms, audio examples, color version of the figures as well as scripts reproducing experiments from this manuscript are freely available at http://ltfat.github.io/notes/040. The code depends on our Matlab/GNU Octave [26] packages LTFAT [27], [28] and PHASERET available at http://ltfat.github.io and http://ltfat.github.io/phaseret, respectively.

The paper is organized as follows. Section II summarizes the necessary theory of the STFT and the Gabor analysis, Section III presents the theory behind the proposed algorithm, Section IV contains a detailed description of the numerical algorithm. Finally, in Section V we present an extensive evaluation of the proposed algorithm and comparison with the iterative and non-iterative state-of-the-art algorithms using the Gaussian window, the truncated Gaussian window, the Hann and the Hamming windows.

II. Gabor analysis

The short-time Fourier transform and its sampled version the Gabor transform are ubiquitous tools for audio analysis and processing. In this section, we define essential formulas for the analysis and the synthesis with respect to a generic window. We will further focus on the properties of the Gaussian window, which is essential for deriving the fundamental equations the PGHI algorithm is based on.

A. STFT

The *short-time Fourier transform* of a function $f \in L^2(\mathbb{R})$ with respect to a window $g \in L^2(\mathbb{R})$ can be defined as[1]

$$(\mathcal{V}_g f)(\omega, t) = \int_{\mathbb{R}} f(\tau + t) g(\tau) \mathrm{e}^{-\mathrm{i}2\pi\omega\tau} \,\mathrm{d}\tau, \quad \omega, t \in \mathbb{R}, \tag{1}$$

assuming both f, g are real valued. The magnitude and phase components can be separated by

$$M_g^f = \left|\mathcal{V}_g f\right| \quad \text{and} \quad \Phi_g^f = \arg\left(\mathcal{V}_g f\right), \tag{2}$$

assuming $\arg(\cdot)$ returns the principal value of the angle. Using the modulation $\left(\mathcal{E}_\omega f\right)(\tau) := \mathrm{e}^{\mathrm{i}2\pi\omega\tau} \cdot f(\tau)$ and translation $\left(\mathcal{T}_t f\right)(\tau) := f(\tau - t)$ we get the alternative representation $\left(\mathcal{V}_g f\right)(\omega, t) = \langle f, \mathcal{T}_t \mathcal{E}_\omega g \rangle$.

The *Gaussian function* is a particularly suitable window function as it possesses optimal time-frequency properties (achieves minimum time-frequency spread [3]) and it allows an algebraic treatment of the equations. It is defined by the following formula

$$\varphi_\lambda(t) = \left(\frac{\lambda}{2}\right)^{-\frac{1}{4}} \mathrm{e}^{-\pi\frac{t^2}{\lambda}} = \left(D_{\sqrt{\lambda}}\varphi_1\right)(t), \tag{3}$$

where $\lambda \in \mathbb{R}^+$ denotes the "width" or the time-frequency ratio of the Gaussian window and D_α is a dilation operator such that $(D_\alpha f)(t) = \frac{1}{\sqrt{|\alpha|}} f(t/\alpha)$, $\alpha \neq 0$. We will use the shortened notation $\varphi(t) = \varphi_1(t)$ in the following text.

[1]In the literature, two other STFT phase conventions can be found. The present one is the most common in the engineering community.

B. Discrete Gabor Transform – DGT

The discrete Gabor transform coefficients $\boldsymbol{c} \in \mathbb{C}^{M\times N}$ of a signal $\boldsymbol{f} \in \mathbb{R}^L$ with respect to a window $\boldsymbol{g} \in \mathbb{R}^L$ can be obtained as [29]

$$\boldsymbol{c}(m,n) = \sum_{l=0}^{L-1} \boldsymbol{f}(l + na)\boldsymbol{g}(l)\mathrm{e}^{-\mathrm{i}2\pi ml/M} \tag{4}$$

$$=: \boldsymbol{s}(m,n) \cdot \mathrm{e}^{\mathrm{i}\boldsymbol{\phi}(m,n)}, \tag{5}$$

for $m = 0, \ldots, M-1$ and $n = 0, \ldots, N-1$, $M = L/b$ is the number of frequency channels, $N = L/a$ number of time shifts, a is the length of the time shift or a hop size in samples in the time direction and b is a hop size in samples in the frequency direction and $(l + na)$ is assumed to be evaluated modulo L. $\boldsymbol{s}$ denotes magnitude of the coefficients and $\boldsymbol{\phi}$ denotes their phase. In the matrix notation, we can write $\boldsymbol{c}_{\text{vec}} = \mathbf{F}_{\boldsymbol{g}}^* \boldsymbol{f}$, where $\boldsymbol{c}_{\text{vec}} \in \mathbb{C}^{MN}$ denotes vectorized $\boldsymbol{c}$ such that $\boldsymbol{c}_{\text{vec}}(m + nM) = \boldsymbol{c}(m,n)$ and $\mathbf{F}_{\boldsymbol{g}}^*$ is a conjugate transpose of $L \times MN$ matrix $\mathbf{F}_{\boldsymbol{g}}$ (note that this matrix has a very particular block-structure [30]). The DGT can be seen as sampling of STFT (both of the arguments ω and t and the involved functions f and g themselves) of one period of L-periodic continuous signal f such that

$$\boldsymbol{c}(m,n) = \left(\mathcal{V}_g f\right)(bm, an) + \mathcal{A}(m,n), \tag{6}$$

for $m = 0, \ldots, M-1$, $n = 0, \ldots, N-1$ where $\mathcal{A}(m,n)$ models both the aliasing and numerical errors introduced by the sampling. The range of m can be shrunken to the first $\lfloor M/2 \rfloor + 1$ values as the remaining coefficients are complex conjugated. Moreover, the zero-frequency coefficients ($m = 0$) are always real and so are the Nyquist-frequency coefficients ($m = M/2$) if M is even.

Signal $\boldsymbol{f}$ can be recovered (up to a numerical precision error) using the following formula

$$\boldsymbol{f}(l) = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} \boldsymbol{c}(m,n)\widetilde{\boldsymbol{g}}(l - na)\mathrm{e}^{\mathrm{i}2\pi m(l-na)/M} \tag{7}$$

for $l = 0, \ldots, L-1$, where $\widetilde{\boldsymbol{g}}$ is the *canonical dual* window. In the matrix notation, we can write $\boldsymbol{f} = \mathbf{F}_{\widetilde{\boldsymbol{g}}}\boldsymbol{c}_{\text{vec}}$. The canonical dual window can be obtained as

$$\widetilde{\boldsymbol{g}} = \left(\mathbf{F}_{\boldsymbol{g}}\mathbf{F}_{\boldsymbol{g}}^*\right)^{-1}\boldsymbol{g}. \tag{8}$$

See e.g. [31] for conditions under which the product $\mathbf{F}_{\boldsymbol{g}}\mathbf{F}_{\boldsymbol{g}}^*$ is (easily) invertible and [32], [33] for efficient algorithms for computing (4), (7) and (8). In particular the block structure can be used for a pre-conditioning approach [30].

One period of the discretized and periodized Gaussian window is given by

$$\boldsymbol{\varphi}_\lambda(l) = \left(\frac{\lambda L}{2}\right)^{-\frac{1}{4}} \sum_{k\in\mathbb{Z}} \mathrm{e}^{-\pi\frac{(l+kL)^2}{\lambda L}}, \tag{9}$$

for $l = 0, \ldots, L-1$. We assume that L and λ are chosen such that the overlap of the window "tails" after periodization is numerically negligible and therefore it is sufficient to sum over

$k \in \{-1, 0\}$ in practice. The width of the Gaussian window at its relative height $h \in [0, 1]$ is given by

$$w_h = \sqrt{-\frac{4\log(h)}{\pi}\lambda L}. \tag{10}$$

The width is given in samples and it can be a non-integer number. This equation becomes relevant when working with truncated Gaussian window and when determining λ for non-Gaussian windows. For each window type used, we took λ of the closest Gaussian window in the least mean square error sense. The window was obtained via a simple heuristic search. Note that all windows used in this manuscript are odd symmetric, such that they have a unique center sample, and they are non-causal such that they introduce no delay. Finally, the discrete Fourier transform of such windows is real.

III. STFT Phase Reconstruction

The algorithm is based on the direct relationship between the partial derivatives of the phase and the log-magnitude of the STFT with respect to the Gaussian window. In this section, we derive such relations and show that, in theory, it is possible to reconstruct the phase from its gradient up to a constant global phase shift. We include a complete derivation since the relations for the STFT as defined in (1) have not appeared in the literature, as far as we know. Our approach is based on the properties of the *Bargmann transform* [3] which is closely related to the STFT with respect to the Gaussian window with $\lambda = 1$.

A. Phase-Magnitude Relationship

It is known that the Bargmann transform of $f \in L^2(\mathbb{R})$

$$(\mathcal{B}f)(z) = 2^{\frac{1}{4}} \int_{\mathbb{R}} f(\tau) \mathrm{e}^{2\pi\tau z - \pi\tau^2 - \frac{\pi}{2}z^2} \,\mathrm{d}\tau, \quad z \in \mathbb{C} \tag{11}$$

is an entire function [34] for all $z \in \mathbb{C}$ and that it relates to the STFT defined in (1) such that

$$(\mathcal{B}f)(z) = \mathrm{e}^{\pi \mathrm{i} t\omega + \pi\frac{|z|^2}{2}} (\mathcal{V}_\varphi f)(-\omega, t), \tag{12}$$

assuming f is real valued and $z = t + \mathrm{i}\omega$. Further, the logarithm of the Bargmann transform is an entire function (apart from zeros) and the real and imaginary parts of $\log(\mathcal{B}f)(z)$ can be written as

$$\log(\mathcal{B}f)(t + \mathrm{i}\omega) = u(\omega, t) + \mathrm{i}v(\omega, t) \tag{13}$$

$$u(\omega, t) = \pi(t^2 + \omega^2)/2 + \log M^f_\varphi(-\omega, t) \tag{14}$$

$$v(\omega, t) = \pi t\omega + \Phi^f_\varphi(-\omega, t) \tag{15}$$

and using the Cauchy-Riemann equations

$$\frac{\partial u}{\partial t}(\omega, t) = \frac{\partial v}{\partial \omega}(\omega, t), \quad \frac{\partial u}{\partial \omega}(\omega, t) = -\frac{\partial v}{\partial t}(\omega, t) \tag{16}$$

we can write (substituting $\omega' = -\omega$) that

$$\frac{\partial}{\partial \omega'}\Phi^f_\varphi(\omega', t) = -\frac{\partial}{\partial t}\log M^f_\varphi(\omega', t) \tag{17}$$

$$\frac{\partial}{\partial t}\Phi^f_\varphi(\omega', t) = \frac{\partial}{\partial \omega'}\log M^f_\varphi(\omega', t) + 2\pi\omega'. \tag{18}$$

A little more general relationships can be obtained for windows defined as $g = \mathcal{O}\varphi_1$ ($\mathcal{O}$ being a fixed bounded operator) and Proposition 1.

Proposition 1. *Let $\mathcal{O}, \mathcal{P}$ be bounded operators such that for all (ω, t) there exist differentiable, strictly monotonic functions $\eta(t)$ and $\xi(\omega)$, such that $\mathcal{T}_t\mathcal{E}_\omega\mathcal{O} = \mathcal{P}\mathcal{T}_{\eta(t)}\mathcal{E}_{\xi(\omega)}$ and let $g = \mathcal{O}\varphi_1$. Then*

$$\frac{\partial}{\partial \omega}\Phi^f_g(\omega, t) = -\frac{\partial}{\partial t}\log M^f_g(\omega, t) \cdot \frac{\xi'(\omega)}{\eta'(t)} \tag{19}$$

$$\frac{\partial}{\partial t}\Phi^f_g(\omega, t) = \frac{\partial}{\partial \omega}\log M^f_g(\omega, t) \cdot \frac{\eta'(t)}{\xi'(\omega)} + 2\pi\xi(\omega)\eta'(t). \tag{20}$$

Proof. Consider

$$\begin{aligned}\left(\mathcal{V}_g f\right)(\omega, t) &= \langle f, \mathcal{T}_t\mathcal{E}_\omega g\rangle = \langle f, \mathcal{T}_t\mathcal{E}_\omega\mathcal{O}\varphi_1\rangle \\ &= \left\langle \mathcal{P}^* f, \mathcal{T}_{\eta(t)}\mathcal{E}_{\xi(\omega)}\varphi_1 \right\rangle \\ &= \left(\mathcal{V}_{\varphi_1}\left(\mathcal{P}^* f\right)\right)\left(\xi(\omega), \eta(t)\right)\end{aligned}$$

and therefore

$$\begin{aligned}\frac{\partial}{\partial t}\Phi^f_g(\omega, t) &= \frac{\partial}{\partial t}\left[\Phi^{\mathcal{P}^* f}_{\varphi_1}\left(\xi(\omega), \eta(t)\right)\right] \\ &= \left[\frac{\partial}{\partial \eta}\Phi^{\mathcal{P}^* f}_{\varphi_1}\left(\xi(\omega), \eta(t)\right)\right] \cdot \eta'(t).\end{aligned}$$

Furthermore

$$\frac{\partial}{\partial \omega}\log M^f_g(\omega, t) = \left[\frac{\partial}{\partial \xi}\log M^{\mathcal{P}^* f}_{\varphi_1}\left(\xi(\omega), \eta(t)\right)\right] \cdot \xi'(\omega).$$

Combining this with (18) we obtain (20)

$$\begin{aligned}\frac{\partial}{\partial t}\Phi^f_g(\omega, t) &= \left[\frac{\partial}{\partial \eta}\Phi^{\mathcal{P}^* f}_{\varphi_1}\left(\xi(\omega), \eta(t)\right)\right] \cdot \eta'(t) \\ &= \left[\frac{\partial}{\partial \xi}\log M^{\mathcal{P}^* f}_{\varphi_1}\left(\xi(\omega), \eta(t)\right) + 2\pi\xi(\omega)\right] \cdot \eta'(t) \\ &= \frac{\partial}{\partial \omega}\log M^f_g(\omega, t) \cdot \frac{\eta'(t)}{\xi'(\omega)} + 2\pi\xi(\omega)\eta'(t).\end{aligned}$$

The other equality can be shown using the same arguments and (17). □

Choosing $\mathcal{O} = D_{\sqrt{\lambda}}$, $\xi(\omega) = \sqrt{\lambda}\omega$ and $\eta(t) = t/\sqrt{\lambda}$ leads to equations for dilated Gaussian window φ_λ

$$\frac{\partial}{\partial \omega}\Phi^f_{\varphi_\lambda}(\omega, t) = -\lambda\frac{\partial}{\partial t}\log M^f_{\varphi_\lambda}(\omega, t) \tag{21}$$

$$\frac{\partial}{\partial t}\Phi^f_{\varphi_\lambda}(\omega, t) = \frac{1}{\lambda}\frac{\partial}{\partial \omega}\log M^f_{\varphi_\lambda}(\omega, t) + 2\pi\omega. \tag{22}$$

The relations were already published in [24], [35], [36] in slightly different forms obtained using different techniques than we use here. The equations differ because the authors of the above mentioned papers use different STFT phase conventions. It should be noted that the relations for general windows were already studied [37], they however involve partial derivatives of the logarithm of the modified Bargmann transform and thus it seems they cannot be exploited directly. Moreover, the experiments presented in Section V show that the performance degradation is not too significant when using windows resembling the Gaussian window like the Hann, the Hamming or the Blackman window.

The STFT phase gradient of a signal f with respect to dilated Gaussian φ_λ will be further denoted as

$$\nabla\Phi^f_{\varphi_\lambda}(\omega, t) = \left[\frac{\partial}{\partial \omega}\Phi^f_{\varphi_\lambda}(\omega, t), \frac{\partial}{\partial t}\Phi^f_{\varphi_\lambda}(\omega, t)\right]. \tag{23}$$

Note that the derivative of the phase has a peculiar pole pattern around zeros [38].

B. Gradient Integration and the Phase Shift Phenomenon

Knowing the phase gradient, one can exploit the gradient theorem (see e.g. [39]) to reconstruct the original (unwrapped) phase $\Phi^f_{\varphi_\lambda}(\omega,t)$ such that

$$\Phi^f_{\varphi_\lambda}(\omega,t)-\Phi^f_{\varphi_\lambda}(\omega_0,t_0)=\int_0^1 \nabla\Phi^f_{\varphi_\lambda}\left(r\left(\tau\right)\right)\cdot\frac{\mathrm{d}r}{\mathrm{d}\tau}\left(\tau\right)\,\mathrm{d}\tau, \tag{24}$$

where $r(\tau)=[r_\omega(\tau),r_t(\tau)]$ is any curve starting at (ω_0,t_0) and ending at (ω,t) provided the phase at the initial point (ω_0,t_0) is known. When the phase is unknown completely, we consider $\Phi^f_{\varphi_\lambda}(\omega_0,t_0)=0$ and therefore the phase one obtains by (24) is

$$\widetilde{\Phi}^f_{\varphi_\lambda}(\omega,t)=\Phi^f_{\varphi_\lambda}(\omega,t)+\Phi_0 \tag{25}$$

where Φ_0 is a constant global phase shift.

The global phase shift of the STFT carries over to the global phase shift of the reconstructed signal through the linearity of the reconstruction. One must, however, treat real input signals with care as the phase shift breaks the complex conjugate relation of the positive and negative frequency coefficients. This relationship has to be either recovered or enforced because if one simply takes only the real part of the reconstructed signal the phase shift causes its amplitude attenuation or even causes the signal to vanish in the extreme case. To explain this phenomenon, consider the following example where we compare the effect of the phase shift on analytic and on real signals. We denote the constant phase shift as ψ_0 and define an analytic signal as $x_{\mathrm{an}}(t)=A(t)\mathrm{e}^{\mathrm{i}\psi(t)}$. The real part including the global phase shift ($\mathrm{e}^{\mathrm{i}\psi_0}$) is given as $\mathcal{R}(x_{\mathrm{an}}(t)\mathrm{e}^{\mathrm{i}\psi_0})=A(t)\cos(\psi(t)+\psi_0)$ which is what one would expect. Similarly, we define a real signal as $x(t)=\frac{A(t)}{2}\left(\mathrm{e}^{\mathrm{i}\psi(t)}+e^{-\mathrm{i}\psi(t)}\right)$ and the real part of such signal with the global phase shift ψ_0 amounts to $\mathcal{R}(x(t)e^{\mathrm{i}\psi_0})=A(t)\cos(\psi_0)\cos(\psi(t))$ which causes the signal to vanish when $\psi_0=\pi/2+k\pi$, $k\in\mathbb{Z}$.

In theory, the global phase shift of the STFT of a real signal can be compensated for, leaving only a global signal sign ambiguity. For real signals, it is clear that the following holds for $\omega\neq 0$

$$\widetilde{\Phi}^f_{\varphi_\lambda}(\omega,t)+\widetilde{\Phi}^f_{\varphi_\lambda}(-\omega,t)=2\Phi_0. \tag{26}$$

After the compensation, due to the phase wrapping, the phase shift is still ambiguous up to an integer multiple of π, which causes the aforementioned signal sign ambiguity.

IV. The Algorithm

In the discrete time setting (recall Section II-B; in particular (4) and (5)) the STFT phase gradient approximation $\widehat{\nabla\Phi_{\varphi_\lambda}}(bm,an):=\nabla\boldsymbol{\phi}(m,n)$ is obtained by numerical differentiation of $\boldsymbol{s}_{\log}(m,n):=\log\left(\boldsymbol{s}(m,n)\right)$ as

$$\nabla\boldsymbol{\phi}(m,n)=\left[\boldsymbol{\phi}_\omega(m,n),\boldsymbol{\phi}_t(m,n)\right]:= \tag{27}$$

$$\left[-\frac{\lambda}{a}(\boldsymbol{s}_{\log}\mathbf{D}_t)(m,n),\frac{1}{\lambda b}(\mathbf{D}_\omega\boldsymbol{s}_{\log})(m,n)+2\pi m/M\right] \tag{28}$$

where $\mathbf{D}_t,\mathbf{D}_\omega$ denote matrices performing the numerical differentiation of $\boldsymbol{s}_{\log}$ along rows (in time) and columns (in frequency) respectively. The matrices are assumed to be scaled such that the sampling step of the differentiation scheme they represent is equal to 1. The central (mid-point) finite difference scheme (see e.g. [40]) is the most suitable because it ensures the gradient components to be sampled at the same grid. The steps of the numerical integration will be done in either horizontal or vertical directions such that exclusively one of the components in $\frac{\mathrm{d}r}{\mathrm{d}\tau}$ from (24) is zero. Due to this property, the gradient can be pre-scaled using lengths of the steps (hop sizes a and b) such that

$$\nabla\boldsymbol{\phi}^{\mathrm{SC}}(m,n):=\left[b\boldsymbol{\phi}_\omega(m,n),a\boldsymbol{\phi}_t(m,n)\right]= \tag{29}$$

$$\left[-\frac{\lambda L}{aM}(\boldsymbol{s}_{\log}\mathbf{D}_t)(m,n),\frac{aM}{\lambda L}(\mathbf{D}_\omega\boldsymbol{s}_{\log})(m,n)+2\pi am/M\right]. \tag{30}$$

Note that the dependency on L can be avoided when (10) is used to express λL. This is useful e.g. when the signal length is not known in advance.

The numerical integration of the phase gradient is performed over the prominent contours of the spectrogram first in order to reduce accumulation of the error. The magnitude of the coefficients is used as a guide such that integration paths are chosen adaptively following the spectrogram ridges first. Such behavior is achieved by employing a heap data structure (from the heapsort algorithm [41]), which it is used for holding pairs (m,n) and it has the property of having (m,n) of the maximum $|\boldsymbol{c}(m,n)|$ always at the top. It is further equipped with efficient operations for insertion and deletion. The effect of the parameter tol is twofold. First, a random phase (uniformly distributed random values from the range $[0,2\pi]$) is assigned to coefficients small in magnitude for which the phase gradient is unreliable [38] and second, the integration is done only locally on "islands" with the max coefficient within the island serving as the zero phase reference. The randomization of the phase of the coefficients below the tolerance is chosen over the zero phase because in practice it helps to avoid the impulsive disturbances introduced by the small phase-aligned coefficients. The algorithm is summarized in Alg. 1 and a graphical step-by-step example can be found at the accompanying webpage.

After $\widehat{\boldsymbol{\phi}}(m,n)$ has been estimated by Alg. 1, it is combined with the target magnitude of the coefficients such that

$$\widehat{\boldsymbol{c}}(m,n)=\boldsymbol{s}(m,n)\mathrm{e}^{\mathrm{i}\widehat{\boldsymbol{\phi}}(m,n)} \tag{31}$$

and the signal is recovered by simply plugging these coefficients into (7).

A. Practical Considerations

In this section, we analyze the effect of the discretization on the performance of the algorithm. The obvious sources of error are the numerical differentiation and integration schemes. However, the aliasing introduced by subsampling in time and frequency domains is more serious. In the discrete time setting, since the signal is considered to be band-limited and periodic, the truly aliasing-free case occurs when $a=1,b=1$ ($M=L,N=L$) regardless of the time or the frequency effective supports of the window. DGT with such setting

Algorithm 1: Phase gradient heap integration – PGHI

Input: DGT phase gradient $\nabla\boldsymbol{\phi}^{\mathrm{SC}}(m,n) = \left(\boldsymbol{\phi}_{\omega}^{\mathrm{SC}}(m,n), \boldsymbol{\phi}_{t}^{\mathrm{SC}}(m,n)\right)$ obtained from (30), magnitude of DGT coefficients $\left|\boldsymbol{c}(m,n)\right|$, relative tolerance tol.

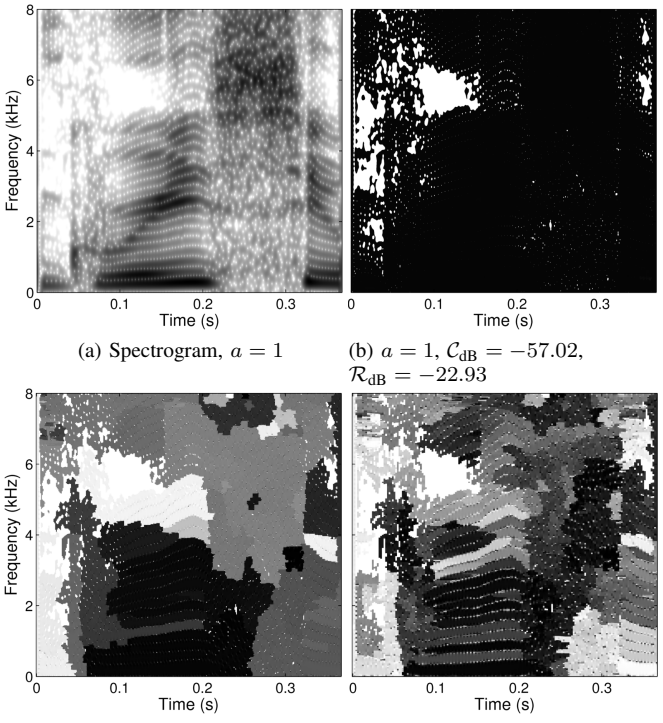
Output: Estimate of the DGT phase $\widehat{\boldsymbol{\phi}}(m,n)$.

Set $\mathcal{I} = \left\{(m,n) : \left|\boldsymbol{c}(m,n)\right| > tol \cdot \max\left(\left|\boldsymbol{c}(m,n)\right|\right)\right\}$;
Assign random values to $\widehat{\boldsymbol{\phi}}(m,n)$ where $(m,n) \notin \mathcal{I}$;
Construct a self-sorting *heap* for (m,n) pairs;
while $\mathcal{I}$ *is not* $\emptyset$ **do**
  if *heap is empty* **then**
    Insert $(m,n)_{\max} = \arg\max_{(m,n)\in\mathcal{I}}\left(\left|\boldsymbol{c}(m,n)\right|\right)$ into the *heap*;
    $\widehat{\boldsymbol{\phi}}(m,n)_{\max} \leftarrow 0$;
    Remove $(m,n)_{\max}$ from $\mathcal{I}$;
  end
  while *heap is not empty* **do**
    $(m,n) \leftarrow$ remove the top of the *heap*;
    if $(m+1,n) \in \mathcal{I}$ **then**
      $\widehat{\boldsymbol{\phi}}(m+1,n) \leftarrow \widehat{\boldsymbol{\phi}}(m,n) + \frac{1}{2}\left(\boldsymbol{\phi}_{\omega}^{\mathrm{SC}}(m,n) + \boldsymbol{\phi}_{\omega}^{\mathrm{SC}}(m+1,n)\right)$;
      Insert $(m+1,n)$ into the *heap*;
      Remove $(m+1,n)$ from $\mathcal{I}$;
    end
    if $(m-1,n) \in \mathcal{I}$ **then**
      $\widehat{\boldsymbol{\phi}}(m-1,n) \leftarrow \widehat{\boldsymbol{\phi}}(m,n) - \frac{1}{2}\left(\boldsymbol{\phi}_{\omega}^{\mathrm{SC}}(m,n) + \boldsymbol{\phi}_{\omega}^{\mathrm{SC}}(m-1,n)\right)$;
      Insert $(m-1,n)$ into the *heap*;
      Remove $(m-1,n)$ from $\mathcal{I}$;
    end
    if $(m,n+1) \in \mathcal{I}$ **then**
      $\widehat{\boldsymbol{\phi}}(m,n+1) \leftarrow \widehat{\boldsymbol{\phi}}(m,n) + \frac{1}{2}\left(\boldsymbol{\phi}_{t}^{\mathrm{SC}}(m,n) + \boldsymbol{\phi}_{t}^{\mathrm{SC}}(m,n+1)\right)$;
      Insert $(m,n+1)$ into the *heap*;
      Remove $(m,n+1)$ from $\mathcal{I}$;
    end
    if $(m,n-1) \in \mathcal{I}$ **then**
      $\widehat{\boldsymbol{\phi}}(m,n-1) \leftarrow \widehat{\boldsymbol{\phi}}(m,n) - \frac{1}{2}\left(\boldsymbol{\phi}_{t}^{\mathrm{SC}}(m,n) + \boldsymbol{\phi}_{t}^{\mathrm{SC}}(m,n-1)\right)$;
      Insert $(m,n-1)$ into the *heap*;
      Remove $(m,n-1)$ from $\mathcal{I}$;
    end
  end
end

is however highly redundant and only signals up to several thousands samples in length can be handled effectively.

In the subsampled case, the amount of aliasing and therefore the performance of the algorithm depends on the effective support of the window. Increasing a introduces aliasing in frequency and increasing b introduces aliasing in time. This property is illustrated by Figure 1, which shows that the algorithm performs very well in the aliasing-free case ($a = 1, b = 1$) and the performance becomes worse when longer hop sizes in time are introduced while keeping the effective width of the window constant. The length of the signal is 5888 samples and the time-frequency ratio of the Gaussian window is $\lambda = 1$. The hop size in frequency is $b = 1$ (i.e. $M = 5888$). Only the values for the 60 dB range of the highest coefficients are shown.

Even though the Gaussian window is in theory infinitely supported in both time and frequency, it decays exponentially and therefore aliasing might not significantly degrade the performance of the algorithm when choosing the hop sizes and the effective support carefully. Obviously the finer the hop sizes the higher the computational cost. The authors recommend to use 87.5% window overlap (redundancy 8), wich is also used in Section V. Since it is clear that the phase shift achieved by the algorithm is not constant, the conjugate symmetry of the DGT of real signals cannot be easily recovered. Therefore, when dealing with real signals, we reconstruct the phase only for the positive frequency coefficients and enforce the conjugate symmetry to the negative frequency coefficients.



(a) Spectrogram, $a = 1$

(b) $a = 1$, $\mathcal{C}_{\mathrm{dB}} = -57.02$, $\mathcal{R}_{\mathrm{dB}} = -22.93$

(c) $a = 16$, $\mathcal{C}_{\mathrm{dB}} = -28.17$, $\mathcal{R}_{\mathrm{dB}} = -2.18$

(d) $a = 32$, $\mathcal{C}_{\mathrm{dB}} = -24.06$, $\mathcal{R}_{\mathrm{dB}} = -1.21$

Fig. 1: Spectrogram of a spoken word *greasy* (a). The absolute phase differences of the STFT of the original and reconstructed signal in the range $[0, \pi]$ for varying time hop size a (b) (c) (d). The errors $\mathcal{C}_{\mathrm{dB}}$ and $\mathcal{R}_{\mathrm{dB}}$ are introduced in Section V.

B. Exploiting Partially Known Phase

In some scenarios, the true phase of some of the coefficients is available. In order to exploit such information, the proposed algorithm has to be adjusted slightly. First, we introduce a

mask to select the reliable coefficients and second, we select the border coefficients i.e. coefficients with at least one neighbor in the time-frequency plane with unknown phase. Then we simply initialize the algorithm with the border coefficients stored in the heap. Formally, Algorithm 1 will be changed such that steps summarized in Algorithm 2 are inserted after line 3. Note that the phase of the border coefficients can be used directly (i.e. no unwrapping is necessary). Depending on the situation, the phase might be propagated from more than one border coefficient, however the phases coming from distinct sources are never combined.

Algorithm 2: Initialization for partially known phase

Input: Set of indices of coefficients $\mathcal{M}$ with known phase $\boldsymbol{\phi}(m,n)$.

$\widehat{\boldsymbol{\phi}}(m,n) \leftarrow \boldsymbol{\phi}(m,n)$ for $(m,n) \in \mathcal{M}$;
for $(m,n) \in \mathcal{M} \cap \mathcal{I}$ **do**
  if $(m+1,n) \notin \mathcal{M}$ *or* $(m-1,n) \notin \mathcal{M}$ *or* $(m,n+1) \notin \mathcal{M}$ *or* $(m,n-1) \notin \mathcal{M}$ **then**
    Add (m,n) to the *heap*;
  end
end

C. Connections to Phase Vocoder

In this section we discuss some connections between the proposed algorithm and the phase vocoder [6] and consequently with algorithms SPSI [22] and PU [23]. The phase vocoder allows the signal duration to be changed by employing non-equal analysis and synthesis time hop sizes. A pitch change can be achieved by playing the signal at a sampling rate adjusted by the ratio of the analysis and synthesis hop sizes. In the synthesis, the phase must be kept *consistent* in order not to introduce artifacts. In the phase reconstruction task, the original phase is not available, but the basic phase behavior can be yet exploited. For example, it is known that for a sinusoidal component with a constant frequency the phase grows linearly in time for all frequency channels the component influences in the spectrogram. For these coefficients, the instantaneous frequency (STFT phase derivative with respect to time (22)) is constant and the local group delay (STFT phase derivative with respect to frequency (21)) is zero.

Algorithms SPSI and PU estimate the instantaneous frequency in each spectrogram column (time frame) from the magnitude by peak picking and interpolation. The instantaneous frequency determines phase increments for each frequency channel m such that

$$\boldsymbol{\phi}(m,n) = \boldsymbol{\phi}(m,n-1) + 2\pi a m_0/M, \tag{32}$$

where m_0 is the estimated, possibly non-integer instantaneous frequency belonging to the interval $\left[0, \lfloor M/2 \rfloor\right]$. This is exactly what the proposed algorithm does in case of constant sinusoidal components, except the instantaneous frequency is determined from the DGT log-magnitude. Integration in Alg. 1 performs nothing else than a cumulative sum of the instantaneous frequency in the time direction.

The algorithm PU goes further and employs an impulse model. The situation is reciprocal to sinusoidal components such that the phase changes linearly in frequency for all coefficients belonging to an impulse component but the rate is only constant for fixed n and it is inversely proportional to the local group delay $n_0 - n$ such as

$$\boldsymbol{\phi}(m,n) = \boldsymbol{\phi}(m-1,n) + 2\pi a (n-n_0)/M, \tag{33}$$

where an_0 is the time index of the impulse occurrence. Again, this is what the proposed algorithm does for coefficients corresponding to impulses.

The advantage of the proposed algorithm over the other two is that the phase gradient is computed from the DGT log-magnitude such that it is available at every time-frequency position without even analysing the spectrogram content. This allows an arbitrary integration path which combines both the instantaneous frequency and the local group delay according to the magnitude ridge orientation. In the other approaches, the phase time derivative can be only estimated in a vicinity of sinusoidal components and, vice versa, the frequency derivative only in a vicinity of impulse-like events. Obviously, such approaches will not cope well with deviations from the model assumptions although careful implementation can handle multiple sinusoidal components with slowly varying instantaneous frequencies and impulses with frequency-varying onsets. The difficulty of algorithm PU lies in detecting the onsets in the spectrogram and separating the coefficients belonging to the impulse-like component from the coefficients belonging to sinusoidal components.

Figure 2 shows phase deviations achieved by algorithms SPSI and PU and by the proposed algorithm PGHI. The phase difference at the transient coefficients is somewhat smoother for PU when compared to SPSI because of the involved impulse model. PGHI produces almost constant phase difference due to the adaptive integration direction. The setup used in the example is the following: the length of the signal is $L = 8192$ samples, time hop size $a = 16$, number of channels $M = 2048$, time-frequency ratio of the Gaussian window is $\lambda = aM/L$. Only the values for the 50 dB range of the highest coefficients are shown.

V. Experiments

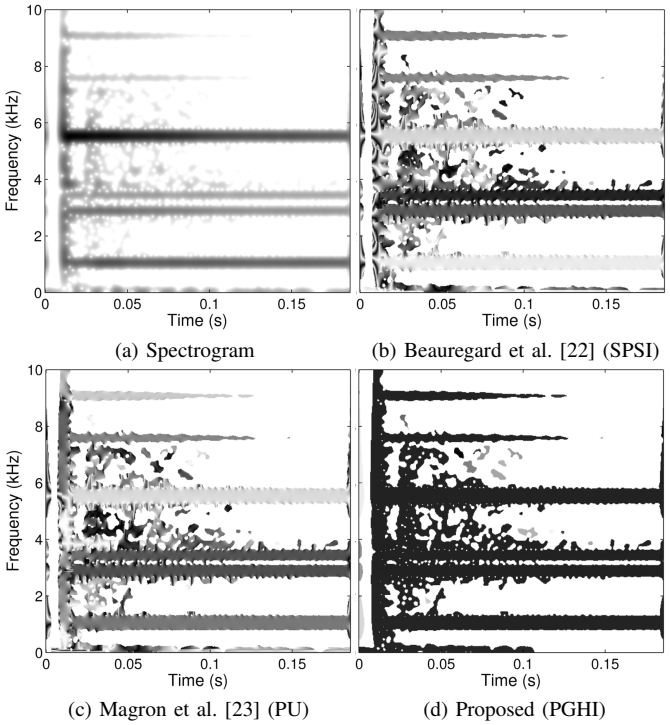
In the experiments, we use the following equation to measure the error

$$E(\boldsymbol{x},\boldsymbol{y}) = \frac{\|\boldsymbol{x}-\boldsymbol{y}\|_2}{\|\boldsymbol{x}\|_2}, \;\; E_{\text{dB}}(\boldsymbol{x},\boldsymbol{y}) = 20\log_{10} E(\boldsymbol{x},\boldsymbol{y}), \tag{34}$$

where $\|.\|_2$ denotes the standard energy norm. The *spectral convergence* [14] is defined as

$$\mathcal{C} = E\left(\boldsymbol{s}_{\text{vec}}, |\mathbf{P}\widehat{\boldsymbol{c}}_{\text{vec}}|\right), \;\; \mathcal{C}_{\text{dB}} = 20\log_{10}\mathcal{C}, \tag{35}$$

where $\mathbf{P} = \mathbf{F}_{\boldsymbol{g}}^{*}\mathbf{F}_{\widetilde{\boldsymbol{g}}}$ i.e. synthesis followed by analysis. Other authors proposed a slightly different measure $E(\widehat{\boldsymbol{c}}_{\text{vec}}, \mathbf{P}\widehat{\boldsymbol{c}}_{\text{vec}})^2$ termed *normalised inconsistency measure* [10] defining normalised energy lost by the reconstruction/projection. Such measures clearly do not accurately reflect the actual signal reconstruction error $\mathcal{R} = E(\boldsymbol{f}, \widehat{\boldsymbol{f}})$, but they are independent



(a) Spectrogram (b) Beauregard et al. [22] (SPSI)

(c) Magron et al. [23] (PU) (d) Proposed (PGHI)

Fig. 2: Spectrogram of an excerpt form the *glockenspiel* signal (a) and the absolute phase differences in the range $[0, \pi]$ for three different algorithms (b)(c)(d).

of the phase shift. Some other authors evaluate the algorithms using the signal to noise ratio, which they define as $SNR(\boldsymbol{x}, \boldsymbol{y}) = 1/E(\boldsymbol{x}, \boldsymbol{y})$ and $SNR_{\mathrm{dB}}(\boldsymbol{x}, \boldsymbol{y}) = -E_{\mathrm{dB}}(\boldsymbol{x}, \boldsymbol{y})$ respectively. Unfortunately, as the phase difference plots in Fig. 1 and Fig. 2 show, the phase difference is usually far from being constant when subsampling is involved (this holds for any algorithm, even the iterative ones). Therefore, the time-frames (i.e. individual short-time spectra) and even each frequency bin within the frame might have a different phase shift, causing the error $\mathcal{R}$ to be very high, even when the other error measures are low and the actual perceived quality is good. An interested reader can find sound examples demonstrating this phenomenon at the accompanying webpage.

The testing was performed on the speech corpus database MOCHA-TIMIT [42] consisting of recordings of 1 male and 1 female speakers (460 recordings for each, 61 minutes in total). The sampling rate of all recordings is 16 kHz. The Gabor system parameters used with this database (Fig 3 and 5) were: number of channels $M = 1024$, hop size $a = 128$, time-frequency ratio of the Gaussian window $\lambda = aM/L$, time support of the truncated Gaussian window and the other compactly supported windows was M samples.

Next, we used the EBU SQAM database of 70 test sound samples [43] recorded at 44.1 kHz. Only the first 10 seconds of the first channel was used from the stereophonic recordings to reduce the execution time to a reasonable value. The Gabor system parameters used with this database (Fig. 4 and 6) were the following: number of channels $M = 2048$, hop size $a = 256$, time-frequency ratio of the Gaussian window $\lambda = aM/L$, time support of the truncated Gaussian window and of the other compactly supported windows was M samples.

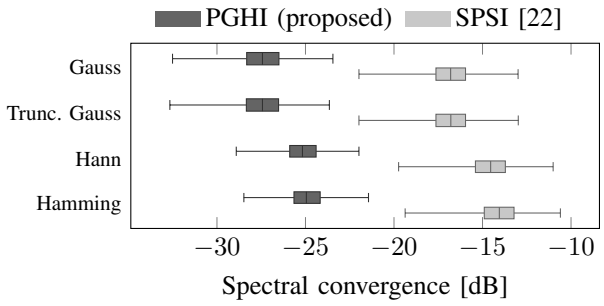


Fig. 3: Box plot of $\mathcal{C}$ in dB for the MOCHA-TIMIT database. The whiskers denote the minimum and maximum.

A. Comparison With Non-iterative Method

In this section, we evaluate the performance of the proposed algorithm and compare it to results obtained by the SPSI [22] algorithm. Unfortunately, we were not able to get good results with the PU [23] algorithm consistently due to the imperfect onset detection and due to the limitation of the impulse model and so did not include it here. The implementation of SPSI has been taken from http://anclab.org/software/phaserecon/ and it was modified to fit our framework. The most prominent change has been the removal of the alternating π and 0 phase modulation in the frequency direction which is not present when computing the transform according to (4).

The results for the proposed algorithm were computed via a two step procedure. In the first step Alg. 1 with $tol = 10^{-1}$ was used, and in the second step, the algorithm was run again with $tol = 10^{-10}$ including steps from Alg. 2 while using the result from the first step as known phase. This approach avoids error spreading during the numerical integration and improves the result considerably when compared to a single run with either of the thresholds.

Fig. 3 and 4 show box plots of $\mathcal{C}$ (converted to a value in dB) over whole databases for the SPSI and the proposed algorithm PGHI. The proposed algorithm very clearly outperforms the SPSI algorithm by a large margin. The performance of the proposed algorithm further depends on the choice of the window. While the Gaussian window truncation introduces only a negligible performance degradation, the choice of Hann or Hamming windows increase the error by about 2 dB. For a detailed comparison, please find the scores and sound examples for the individual files from the EBU SQAM database using the Gaussian window at the accompanying web page http://ltfat.github.io/notes/040.

We can only provide a rough timing for the algorithms as the actual execution time is highly signal dependent and our implementations might be suboptimal. In the setup used in the tests, the runtime of the proposed algorithm is about 6–8 times longer than of the implementation of the SPSI algorithm. In particular, the current implementation of the proposed algorithm is very slow for noise signals.

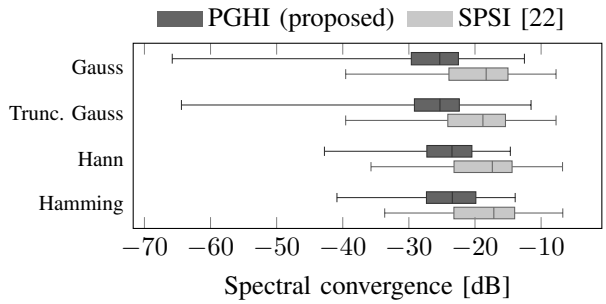


Fig. 4: Box plot of $\mathcal{C}$ in dB for the EBU SQAM database. The whiskers denote the minimum and maximum.

B. Comparison With Iterative Methods

We further compare the present algorithm with the following iterative algorithms:

- The Griffin-Lim algorithm [9] (GLA) as the baseline.
- A combination of Le Roux's modification of GLA [10] using the *on-the-fly truncated modified update* and of the fast version of GLA [11] with constant $\alpha = 0.99$ (FleGLA). The projection kernel was always truncated to size $2M/a - 1$ in both directions. This combination outperforms both algorithms [10] and [11] when used individually.
- The gradient descend-like algorithm by Decorsiere et al. [15] (lBFGS). Unfortunately, the lBFGS implementation we use (downloaded from [44]) fails in some cases.
- Time-Frequency domain Real-Time Iterative Spectrogram Inversion with Look-Ahead [13] (TF-RTISI-LA). The number of the look-ahead frames was always $M/a - 1$ and an asymmetric analysis window was used for the latest look-ahead frame.

Since all the iterative algorithms optimize a non-convex objective function, the result depends strongly on the initial phase estimate. In addition to the zero phase initialization, we also evaluate performance of the algorithms initialized with the phase computed using the proposed algorithm PGHI. We will denote such combinations as PGHI then GLA, PGHI then FleGLA and PGHI then lBFGS. Unfortunately, our tests showed that the TF-RTISI-LA algorithm does not benefit from phase initialization as it performs its own initial phase guess from the partially reconstructed signal. Note that the SPSI initialization is also possible, but, according to our tests, SPSI-initialized algorithms do not reach performance PGHI-initialized algorithms even after 200 iterations.

Figures 5 and 6 show average $\mathcal{C}$ in dB over the MOCHA-TIMIT and EBU SQAM databases respectively depending on the number of iterations without (solid lines) or with PGHI initialization (dashed lines). The horizontal dashed line is the average $\mathcal{C}$ in dB achieved by the proposed algorithm. In addition, the scores and sound examples for individual files from the EBU SQAM database using the Gaussian window can be found at the accompanying web page. Graphs for the truncated Gaussian window are not shown as they exhibit no visual difference from the graphs for the full-length Gaussian window. Further, the lBFGS algorithm has been excluded from the comparison using the EBU-SQAM database (Fig. 6); it failed to finish for a considerable number of the excerpts.

The graphs show that the proposed algorithm provides a suitable initial phase for the iterative algorithms i.e. the best overall results are obtained when PGHI is combined with the FleGLA and lBFGS algorithms. When considering the iterative algorithms without PGHI initialization, the crossing points indicating the number of iterations necessary to achieve the performance of the proposed algorithm can be clearly identified (if present). For the MOCHA TIMIT database (Fig. 5), the crossing point of the best algorithms is at about 20 iterations and the behavior is consistent for all the windows. The results for the EBU SQAM database (Fig. 6) are more erratic. First of all, the GLA algorithm never reaches the performance of the proposed algorithm even in 200 iterations. The crossing point of the FleGLA algorithm varies from 30 to 40 iterations depending on the window used. On the other hand, the TF-RTISI-LA algorithm gets close to the PGHI line only for 8 per-frame iterations using the Gaussian window (Fig. 6a) and it performs better than the proposed algorithm in all the other cases. The TF-RTISI-LA algorithm however does not perform favorably for sound excerpts like castanets (crossing point at 80–120 iterations), drums, cymbals and glockenspiel (crossing points 20–40 iterations).

In the tests, the execution time of the proposed algorithm was comparable to the execution time of 2–4 iterations of GLA with the Gaussian window and to the execution time of 4–10 iterations for the compactly supported windows.

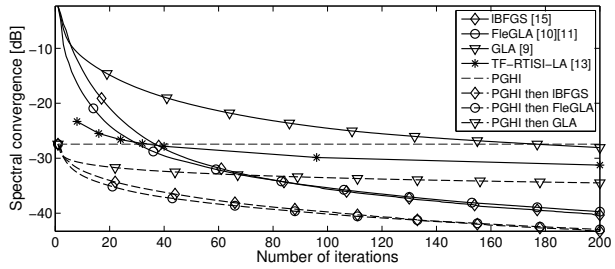
C. Modified Spectrograms

The main application area of the phase reconstruction algorithms is the reconstruction from the modified spectrograms. The spectrograms are modified in the complex-valued STFT domain. This could be done by multiplication which leads to so-called Gabor filters [45], [46] or by moving/copying of contents. In general, such a modified spectrogram is no longer a valid (consistent [10]) spectrogram, i.e. there is no signal having such spectrogram. Therefore the task is to construct rather than reconstruct a suitable phase. Unfortunately, it is neither clear for which spectrogram modifications the equations (21) and (22) still hold nor how it does affect the performance if they do not. Moreover, an objective comparison of the algorithms becomes difficult as the error measures chosen above become irrelevant.

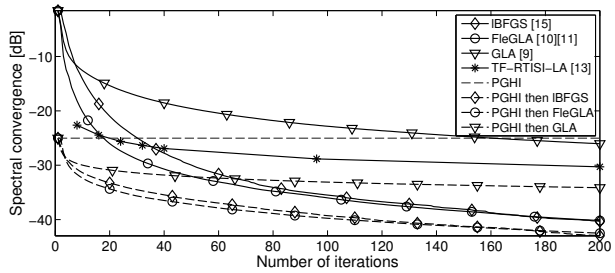
Nevertheless, in order to get the idea of the performance of the proposed algorithm acting on modified spectrograms, we implemented phase vocoder-like pitch shifting (up and down by 6 semitones) via changing the hop size [6], [47] using all the algorithms to rebuild the phase. The synthesis hop size $a = 256$ was fixed and the analysis hop size was changed accordingly to achieve the desired effect. Sound examples for the EBU SQAM database along with Matlab/GNU Octave script generating them can be found at the accompanying web page.

VI. Conclusion

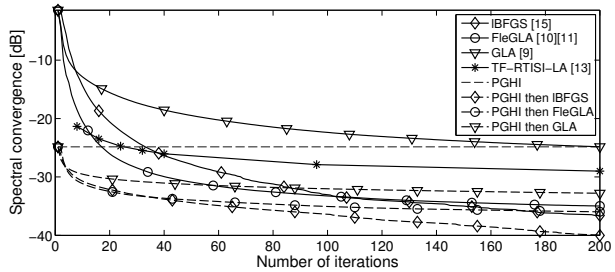
A novel, non-iterative algorithm for the reconstruction of the phase from the STFT magnitude has been proposed. The



(a) Gaussian window

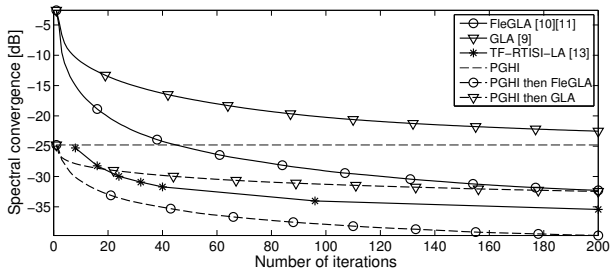


(b) Hann window

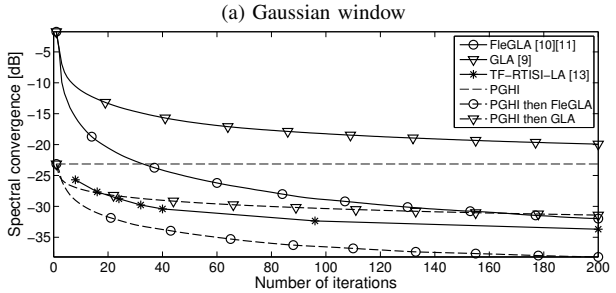


(c) Hamming window

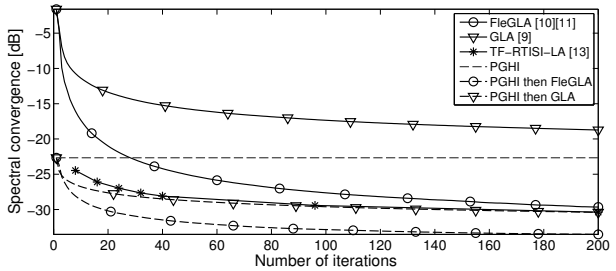
Fig. 5: Comparison with the iterative algorithms, MOCHA-TIMIT database.



(a) Gaussian window



(b) Hann window



(c) Hamming window

Fig. 6: Comparison with the iterative algorithms, EBU SQAM database.

algorithm is computationally efficient and its performance is competitive with the state-of-the-art algorithms. It can also provide a suitable initial phase for the iterative algorithms.

As a future work it would be interesting to investigate whether (simple) equations similar to (21) and (22) could be found for non-Gaussian windows. Moreover, the effect of the aliasing and spectrogram modifications on the phase-magnitude relationship should be systematically explored. For that we will extend Proposition 1 to a more general setting. Ideally, we hope that a similar result could be possible for α-modulation frames [48], [49] and warped time-frequency frames [50], [51].

From the practical point of view, a drawback of the proposed algorithm is the inability to run in real-time setting i.e. to process streams of audio data in a frame by frame manner. Clearly, the way how the phase is spread among the coefficients would have to be adjusted. This was done in [25] where we present a version of the algorithm introducing one or even zero frame delay.

Further, please note that equations (21) and (22) hold "in the other direction" as well; meaning they can be used to estimate the magnitude given the phase. This property might be useful in many applications since the phase-aware signal processing is a promising field of research [52], [53].

ACKNOWLEDGEMENTS

The authors thank Pavel Rajmic and anonymous reviewers for their valuable comments.

REFERENCES

[1] R. W. Gerchberg and W. O. Saxton, "A practical algorithm for the determination of the phase from image and diffraction plane pictures," *Optik*, vol. 35, pp. 237–246, 1972.

[2] E. J. Cands, T. Strohmer, and V. Voroninski, "Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming," *Communications on Pure and Applied Mathematics*, vol. 66, no. 8, pp. 1241–1274, 2013.

[3] K. Gröchenig, *Foundations of time-frequency analysis*, ser. Applied and numerical harmonic analysis. Boston, Basel, Berlin: Birkhäuser, 2001.

[4] D. Gunawan and D. Sen, "Iterative phase estimation for the synthesis of separated sources from single-channel mixtures," *IEEE Signal Processing Letters*, vol. 17, no. 5, pp. 421–424, May 2010.

[5] N. Sturmel and L. Daudet, "Iterative phase reconstruction of Wiener filtered signals," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, March 2012, pp. 101–104.

[6] J. Laroche and M. Dolson, "Improved phase vocoder time-scale modification of audio," *Speech and Audio Processing, IEEE Transactions on*, vol. 7, no. 3, pp. 323–332, May 1999.

[7] V. Gnann and M. Spiertz, "Comb-filter free audio mixing using STFT magnitude spectra and phase estimation," in *Proc. 11th Int. Conf. on Digital Audio Effects (DAFx-08)*, Sep. 2008.

[8] P. Smaragdis, B. Raj, and M. Shashanka, "Missing data imputation for time-frequency representations of audio signals," *Journal of Signal Processing Systems*, vol. 65, no. 3, pp. 361–370, 2011.

[9] D. Griffin and J. Lim, "Signal estimation from modified short-time Fourier transform," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 32, no. 2, pp. 236–243, Apr 1984.

[10] J. Le Roux, H. Kameoka, N. Ono, and S. Sagayama, "Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency," in *Proc. 13th Int. Conf. on Digital Audio Effects (DAFx-10)*, Sep. 2010, pp. 397–403.

[11] N. Perraudin, P. Balazs, and P. Søndergaard, "A fast Griffin-Lim algorithm," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), IEEE Workshop on*, Oct 2013, pp. 1–4.

[12] X. Zhu, G. T. Beauregard, and L. Wyse, "Real-time signal estimation from modified short-time Fourier transform magnitude spectra," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 5, pp. 1645–1653, July 2007.

[13] J. Le Roux, H. Kameoka, N. Ono, and S. Sagayama, "Phase initialization schemes for faster spectrogram-consistency-based signal reconstruction," in *Proceedings of the Acoustical Society of Japan Autumn Meeting*, no. 3-10-3, Mar. 2010.

[14] N. Sturmel and L. Daudet, "Signal reconstruction from STFT magnitude: A state of the art," *Proc. 14th Int. Conf. Digital Audio Effects (DAFx-11)*, pp. 375–386, 2011.

[15] R. Decorsiere, P. Søndergaard, E. MacDonald, and T. Dau, "Inversion of auditory spectrograms, traditional spectrograms, and other envelope representations," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, vol. 23, no. 1, pp. 46–56, Jan 2015.

[16] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase recovery, maxcut and complex semidefinite programming," *Math. Program.*, vol. 149, no. 1-2, pp. 47–81, 2015.

[17] E. J. Candes, Y. C. Eldar, T. Strohmer, and V. Voroninski, "Phase retrieval via matrix completion," *SIAM Review*, vol. 57, no. 2, pp. 225–251, 2015.

[18] D. L. Sun and J. O. Smith, III, "Estimating a signal from a magnitude spectrogram via convex optimization," in *Audio Engineering Society Convention 133*, Oct 2012.

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

[20] Y. Eldar, P. Sidorenko, D. Mixon, S. Barel, and O. Cohen, "Sparse phase retrieval from short-time Fourier measurements," *IEEE Signal Processing Letters*, vol. 22, no. 5, pp. 638–642, May 2015.

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

[22] G. T. Beauregard, M. Harish, and L. Wyse, "Single pass spectrogram inversion," in *Digital Signal Processing (DSP), IEEE International Conference on*, July 2015, pp. 427–431.

[23] P. Margon, R. Badeau, and B. David, "Phase reconstruction of spectrograms with linear unwrapping: application to audio signal restoration," in *Proc. 23rd European Signal Processing Conference (EUSIPCO 2015)*, Aug 2015.

[24] M. R. Portnoff, "Magnitude-phase relationships for short-time Fourier transforms based on Gaussian analysis windows," in *Acoustics, Speech, and Signal Processing (ICASSP), 1979 IEEE International Conference on*, vol. 4, Apr 1979, pp. 186–189.

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

[26] J. W. Eaton, D. Bateman, S. Hauberg, and R. Wehbring, *GNU Octave version 4.0.0 manual: A high-level interactive language for numerical computations*, 2015. [Online]. Available: http://www.gnu.org/software/octave/doc/interpreter

[27] P. L. Søndergaard, B. Torrésani, and P. Balazs, "The Linear Time Frequency Analysis Toolbox," *International Journal of Wavelets, Multiresolution Analysis and Information Processing*, vol. 10, no. 4, 2012.

[28] Z. Průša, P. L. Søndergaard, N. Holighaus, C. Wiesmeyr, and P. Balazs, "The Large Time-Frequency Analysis Toolbox 2.0," in *Sound, Music, and Motion*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 419–442.

[29] P. L. Søndergaard, "Gabor frames by Sampling and Periodization," *Adv. Comput. Math.*, vol. 27, no. 4, pp. 355 –373, 2007.

[30] P. Balazs, H. G. Feichtinger, M. Hampejs, and G. Kracher, "Double preconditioning for Gabor frames," *IEEE T. Signal. Proces.*, vol. 54, no. 12, pp. 4597–4610, December 2006.

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

[32] T. Strohmer, *Numerical Algorithms for Discrete Gabor Expansions*. Birkhäuser Boston, 1998, ch. 8, pp. 267–294.

[33] P. L. Søndergaard, "Efficient algorithms for the discrete Gabor transform with a long FIR window," *J. Fourier Anal. Appl.*, vol. 18, no. 3, pp. 456–470, 2012.

[34] J. B. Conway, *Functions of One Complex Variable I*, 2nd ed., ser. Graduate texts in Mathematics. Springer-Verlag New York, 1978, vol. 11.

[35] T. J. Gardner and M. O. Magnasco, "Sparse time-frequency representations," *Proceedings of the National Academy of Sciences*, vol. 103, no. 16, pp. 6094–6099, 2006.

[36] F. Auger, E. Chassande-Mottin, and P. Flandrin, "On phase-magnitude relationships in the short-time Fourier transform," *IEEE Signal Processing Letters*, vol. 19, no. 5, pp. 267–270, May 2012.

[37] E. Chassande-Mottin, I. Daubechies, F. Auger, and P. Flandrin, "Differential reassignment," *IEEE Signal Processing Letters*, vol. 4, no. 10, pp. 293–294, 1997.

[38] P. Balazs, D. Bayer, F. Jaillet, and P. Søndergaard, "The phase derivative around zeros of the short-time Fourier transform," *Applied and Computational Harmonic Analysis*, vol. 30, no. 3, pp. 610–621, May 2015, 2016.

[39] G. N. Felder and K. M. Felder, *Mathematical Methods in Engineering and Physics*. John Wiley & Sons, Inc., 2015.

[40] R. L. Burden and J. D. Faires, *Numerical Analysis*, 9th ed. Brooks/Cole, Cengage Learning, 2010.

[41] J. W. J. Williams, "Algorithm 232: Heapsort," *Communications of the ACM*, vol. 7, no. 6, p. 347348, 1964.

[42] A. Wrench, "MOCHA-TIMIT: Multichannel articulatory database," 1999. [Online]. Available: http://www.cstr.ed.ac.uk/research/projects/artic/mocha.html

[43] "Tech 3253: Sound Quality Assessment Material recordings for subjective tests," The European Broadcasting Union, Geneva, Tech. Rep., Sept. 2008. [Online]. Available: https://tech.ebu.ch/docs/tech/tech3253.pdf

[44] M. Schmidt, "minFunc: Unconstrained differentiable multivariate optimization in Matlab," 2005. [Online]. Available: http://www.cs.ubc.ca/%7Eschmidtm/Software/minFunc.html

[45] F. Hlawatsch, G. Matz, H. Kirchauer, and W. Kozek, "Time-frequency formulation, design, and implementation of time-varying optimal filters for signal estimation," *Signal Processing, IEEE Transactions on*, vol. 48, no. 5, pp. 1417 –1432, may 2000.

[46] P. Balazs, B. Laback, G. Eckel, and W. A. Deutsch, "Time-frequency sparsity by removing perceptually irrelevant components using a simple model of simultaneous masking," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 18, no. 1, pp. 34–49, 2010.

[47] U. Zoelzer, Ed., *DAFX: Digital Audio Effects*. New York, NY, USA: John Wiley & Sons, Inc., 2002.

[48] S. Dahlke and G. Teschke, "Coorbit theory, multi-alpha-modulation frames and the concept of joint sparsity for medical multi-channel data analysis," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, pp. Article ID 471 601, 19 pages, 2008.

[49] M. Speckbacher, D. Bayer, S. Dahlke, and P. Balazs, "The α-modulation transform: Admissibility, coorbit theory and frames of compactly supported functions," 2014, arXiv:1408.4971.

[50] N. Holighaus and C. Wiesmeyr, "Construction of warped time-frequency representations on nonuniform frequency scales, part I: Frames," 2015, arXiv:1409.7203, submitted.

[51] N. Holighaus, C. Wiesmeyr, and P. Balazs, "Construction of warped time-frequency representations on nonuniform frequency scales, part II: Integral transforms, function spaces, atomic decompositions and Banach frames," arXiv:1409.7203, submitted.

[52] P. Mowlaee, J. Kulmer, J. Stahl, and F. Mayer, *Single Channel Phase-Aware Signal Processing in Speech Communication: Theory and Practice*. John Wiley & Sons, Inc., 2016.

[53] P. Mowlaee, R. Saeidi, and Y. Stylianou, "Advances in phase-aware signal processing in speech communication," *Speech Communication*, vol. 81, pp. 1 – 29, 2016.