SoftwareX 10 (2019) 100275

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

OptFROG — Analytic signal spectrograms with optimized time–frequency resolution

O. Melchert [a,b,c,*], B. Roth [b,c], U. Morgner [a,b], A. Demircan [a,b,c]

[a] Institute of Quantum Optics (IQO), Leibniz Universität Hannover, 30167 Hannover, Germany
[b] Cluster of Excellence PhoenixD (Photonics, Optics, and Engineering - Innovation Across Disciplines), Hannover, Germany
[c] Hannover Centre for Optical Technologies (HOT), 30167 Hannover, Germany

ARTICLE INFO

Article history:
Received 10 April 2019
Received in revised form 27 June 2019
Accepted 27 June 2019

Keywords:
Spectrogram
Short-time Fourier analysis
Analytic signal
Optics
Ultrashort pulse propagation

ABSTRACT

A Python package for the calculation of spectrograms with optimized time and frequency resolution for application in the analysis of numerical simulations on ultrashort pulse propagation is presented. Gabor's uncertainty principle prevents both resolutions from being optimal simultaneously for a given window function employed in the underlying short-time Fourier analysis. Our aim is to yield a time–frequency representation of the input signal with marginals that represent the original intensities per unit time and frequency similarly well. As a use-case, we demonstrate the implemented functionality for the analysis of simulations on ultrashort pulse propagation in a nonlinear waveguide.

Code metadata

Current code version	1.0.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2019_130
Code Ocean compute capsule	https://doi.org/10.24433/CO.3464313.v1
Legal Code License	MIT License
Code versioning system used	none
Software code languages, tools, and services used	Python, GitHub
Compilation requirements, operating environments & dependencies	The `optfrog` package requires Python, numpy and scipy. The installation process requires Pythons setuptool package and the provided use-cases need Pythons matplotlib for figure generation.
If available Link to developer documentation/manual	Documentation provided within code
Support email for questions	melchert@iqo.uni-hannover.de

1. Motivation and significance

The spectrogram provides a particular time–frequency representation of signals that vary in time (for example, see [1]). It represents an essential tool in the analysis of the characteristics of ultrashort optical pulses. Spectrograms are employed in the analysis of data retrieved from experiments [2–5], where it is referred to as frequency resolved optical gating (FROG) analysis, and numerical simulations [6–8], carried out to complement experiments and to provide a basis for the interpretation of the observed effects. This highlights the relevance of signal processing in the field of nonlinear optics and demonstrates the need to be able to compute such spectrograms in the first place. Here, we consider the issue of obtaining *optimal* time–frequency representations of signals for the interpretation of numerical experiments on ultrashort pulse propagation in nonlinear waveguides.

In principle, a spectrogram measures the properties of the signal under scrutiny as well as those of a user-specified window function for localizing parts of the signal during analysis. Exhibiting features of both, the interpretation of the spectrogram is strongly affected by the particular function used for windowing. Different window functions estimate different signal properties,

* Corresponding author at: Institute of Quantum Optics (IQO), Leibniz Universität Hannover, 30167 Hannover, Germany.
E-mail address: melchert@iqo.uni-hannover.de (O. Melchert).

https://doi.org/10.1016/j.softx.2019.100275
2352-7110/

e.g., if a given function achieves a good approximation of the intensities per unit time of the underlying signal, its approximation of the intensities per unit frequency might be bad. Consequently, the spectrogram might suffer from distortion yielding an unreasonable characterization of the time–frequency features of the signal under scrutiny. The usual approach for deciding on a particular window function is by trial-and-error and guided by the liking and experience of the individual.

Here we present a software tool, called OptFROG, that aims at minimizing the mismatch between the signals actual intensities per unit time and frequency and their corresponding estimates obtained from the spectrogram. Within OptFROG, the latter is constructed for a user-supplied, parameterized window function. The resulting spectrograms are "optimal" in the sense that their visual inspection exhibits a minimal amount of distortion and thus allow for a reliable interpretation of the time–frequency composition of the input signal. Such an approach was previously shown to result in a reasonable characterization of the underlying time–frequency features [9]. It is further independent of the experience of the individual user and thus yields reproducible results. To demonstrate the advantage of our approach we address the highly complex propagation dynamics given by the supercontinuum generation process [10], which requires a sophisticated choice of a parameter setting used for signal windowing [6].

2. Software description

OptFROG facilitates the construction of spectrograms for the real-valued optical field $E(t)$, included in the analytic signal (AS) $\mathcal{E}(t)$ [11]. In the Fourier domain, the angular frequency components of both are related via $\hat{\mathcal{E}}(\omega) = [1+\mathrm{sgn}(\omega)]\,\hat{E}(\omega)$ [12]. Due to its one-sided spectral definition the time-domain representation of the AS is complex. Its definition further implies the relation $E(t) = \mathrm{Re}[\mathcal{E}(t)]$ (for example, see [12]). The construction of an AS spectrogram relies on the repeated calculation of the spectrum of the modified signal $\mathcal{E}(t)h(t-\tau)$ at different delay times τ in terms of the short-time Fourier transform

$$S_\tau(\omega) = \frac{1}{\sqrt{2\pi}} \int \mathcal{E}(t)h(t-\tau)\exp\{-i\omega t\}\,\mathrm{d}t, \quad (1)$$

wherein $h(t)$ specifies a narrow window function centered at $t = 0$ and decaying to zero for increasing $|t|$. The latter allows to selectively filter parts of the AS and to estimate its local frequency content. Scanning over a range of delay times then yields the spectrogram as $P_S(\tau,\omega) = |S_\tau(\omega)|^2$, providing a joint time–frequency distribution of both, the AS and the window function [13]. For assessing the approximation quality of P_S, we utilize its time and frequency marginals

$$\mathsf{P}_1(\tau) = \int P_S(\tau,\omega)\,\mathrm{d}\omega, \quad \text{and} \quad (2)$$

$$\mathsf{P}_2(\omega) = \int P_S(\tau,\omega)\,\mathrm{d}\tau. \quad (3)$$

Note that in the limit where $h(t)$ approaches a delta function, the time marginal will approach the intensity per unit time $|\mathcal{E}(t)|^2$ but the frequency marginal will represent the intensity per unit frequency $|\hat{\mathcal{E}}(\omega)|^2$ only poorly. As a result, time resolution will be good and frequency resolution will be bad, see the discussion in Section 3 below. The time–frequency uncertainty principle prevents both resolutions from being optimal simultaneously [1].

The aim of the presented package is to obtain a time–frequency representation of the input signal for which the integrated absolute error (IAE) between its normalized marginals and the original intensities per unit time and frequency are minimal. We consider a single parameter window function $h(t,\sigma)$, e.g. a Gaussian function with mean t and root-mean-square (rms) width σ, and solve for

$$\sigma^\star = \arg\min_\sigma Q(\sigma,\alpha) \quad (4)$$

wherein the objective function Q is defined by

$$Q(\sigma,\alpha) \equiv (1+\alpha)\mathrm{IAE}_1 + (1-\alpha)\mathrm{IAE}_2 \quad (5)$$

with the integrated absolute errors

$$\mathrm{IAE}_1 \equiv \int \left| |\mathcal{E}(\tau)|^2 - \frac{\mathsf{P}_1^{(\sigma)}(\tau)}{E_S} \right| \mathrm{d}\tau, \quad (6)$$

$$\mathrm{IAE}_2 \equiv \int \left| |\hat{\mathcal{E}}(\omega)|^2 - \frac{\mathsf{P}_2^{(\sigma)}(\omega)}{E_S} \right| \mathrm{d}\omega. \quad (7)$$

Above, the underlying spectrogram is computed via $h(t,\sigma)$, indicated by the superscript σ on the marginals, and we assume normalization to $\int |\mathcal{E}(t)|^2\,\mathrm{d}t = 1$ and a total signal energy $E_S = \iint P_S(\tau,\omega)\,\mathrm{d}\tau\,\mathrm{d}\omega$ in terms of the spectrogram. For a good agreement of the marginals and the original intensities, the objective function Q assumes a small value. The additional parameter α might be adjusted to give more weight to frequency resolution ($\alpha < 0$) or time resolution ($\alpha > 0$) if appropriate. The particular choice $\alpha = 0$ yields a balanced time–frequency representation, see the example provided in Section 3. The optimized spectrogram is then computed by using $h(t) \equiv h(t,\sigma^\star)$ for windowing. For the minimization of the scalar function $Q(\sigma,\alpha)$ in the variable σ, the `scipy` native function `scipy.optimize.minimize_scalar` is employed in `bounded` mode. As reasonable bounds the values $\sigma_{\mathrm{min}} = 0$ and $\sigma_{\mathrm{max}} = T/100$, with T specifying the full temporal period of the underlying signal, are considered. The algorithm then proceeds to find a local minimum $\sigma_{\mathrm{min}} < \sigma^\star < \sigma_{\mathrm{max}}$ of Q. For all of our use-cases, including more intricate propagation scenarios than those presented here, this resulted in a satisfactory performance.

2.1. Software architecture

OptFROG, following the naming convention [14] for Python packages implemented as `optfrog`, uses the Python programming language [15] and depends on the functionality of `numpy` and `scipy` [16]. It further follows a procedural programming paradigm.

2.2. Software functionalities

The current version of `optfrog` comprises five software units having the subsequent responsibilities:

vanillaFrog Compute a standard spectrogram $P_S(\tau,\omega)$ for the normalized time-domain analytic signal for a particular window function $h(t,\sigma)$.

optFrog Compute a time–frequency resolution optimized spectrogram for the normalized time-domain analytic signal using the window function $h(t,\sigma^\star)$ that minimizes the total IAE of both marginals.

timeMarginal Compute the marginal distribution in time P_1 based on the spectrogram.

frequencyMarginal Compute the marginal distribution in frequency P_2 based on the spectrogram.

totalEnergy Compute the total energy E_S provided by the spectrogram approximation of the time–frequency characteristics of the signal.

For a more detailed description of function parameters and return values we refer to the documentation provided within the code [17].

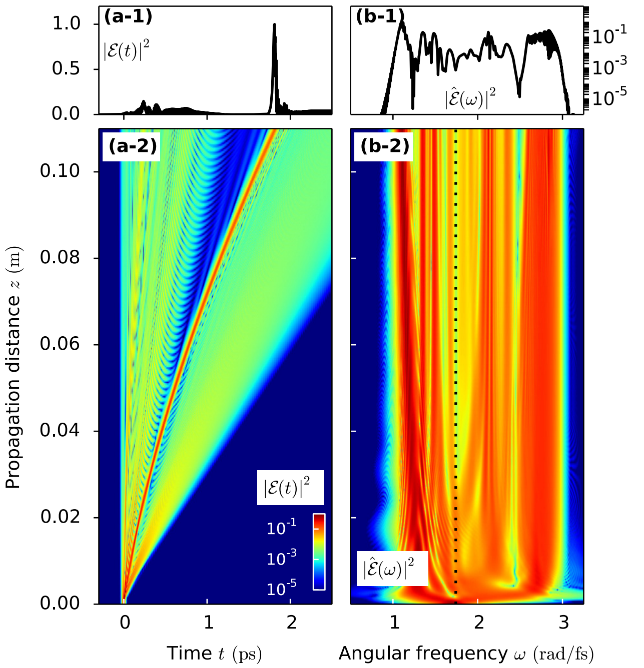

Fig. 1. Evolution of the analytic signal for a $t_0 = 7\,$fs soliton pulse of order $N_s = 8$ and center frequency $\omega = 1.7\,$rad/fs. (a-1) Normalized squared modulus at propagation distance $z = 0.11\,$m, and, (a-2) full evolution in the time domain. (b-1) Normalized squared modulus spectrum at $z = 0.11\,$m, and, (b-2) propagation characteristics.

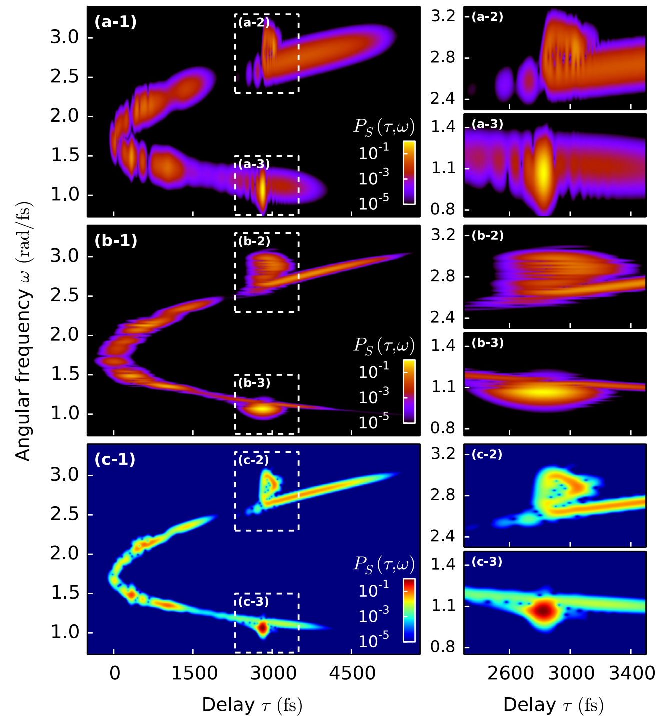

Fig. 2. Analytic signal spectrograms allowing for the time–frequency characterization of a real optical field obtained from the numerical propagation of an ultrashort pulse in an ESM photonic crystal fiber. (a-1) `vanillaFrog`-trace for a Gaussian window function with rms-width $\sigma = 10\,$fs and close-up views of an interacting dispersive wave (a-2) and soliton (a-3) (dashed boxes in (a-1)). (b-2) `vanillaFrog`-trace for $\sigma = 140\,$fs and close-ups (b-2) and (b-3) as in (a). (c-1) Balanced `optFrog`-trace for $\sigma^\star = 39.1\,$fs and close-ups (c-2) and (c-3) as in (a).

2.3. Sample code snippet

In our research work we use `optfrog` mainly in script mode. An exemplary data postprocessing script, reproducing Fig. 4 discussed in Section 3 below, is shown in listing 1. Therein, after importing the functionality of `numpy`, `optfrog`, and a custom figure generating routine in lines 1–3, the location of the input data (line 5) and filter options for the spectrogram output-data (lines 6p) are specified. Note that the user defined window function (lines 9p) does not need to be normalized. After loading the input data (lines 12p) the routine `optFrog` is used to compute an optimized spectrogram in line 15. Finally, a visual account of the latter is prepared by the routine `spectrogramFigure` in line 17.

Listing 1: Exemplary Python script using `optfrog` for the calculation of a time–frequency resolution optimized spectrogram.

```
1  import numpy as np
   from optfrog import optFrog
3  from figure import spectrogramFigure

5  fName = './data/exampleData_pulsePropagation.npz'
   tPars = (-500.0, 5800.0, 10)
7  wPars = (  0.75,    3.25,  3)

9  def wFunc(s0):
       return lambda x: np.exp(-x**2/2/s0/s0)
11
   data = np.load(fName)
13 t, Et = data['t'], data['Et']

15 res = optFrog(t,Et,wFunc,tLim=tPars,wLim=wPars)

17 spectrogramFigure((t,Et),res)
```

3. Illustrative examples

So as to demonstrate the functionality of `optfrog` we consider the supercontinuum generation process in nonlinear fibers, a scheme commonly used nowadays. It provides an example where the complex temporal and spectral evolution cannot easily be resolved, in both, theory and experiment. This results in difficulties to characterize the time–frequency relationships as a plethora of different optical effects are involved [10,18]. An example is shown in Fig. 1, exemplifying the numerical propagation of a short and intense few-cycle optical pulse in presence of the refractive index profile of an "endlessly single mode" (ESM) photonic crystal fiber [19,20]. The underlying unidirectional propagation model includes the Kerr effect and a delayed Raman response of Hollenbeck–Cantrell type [21]. For the preparation of the initial condition we considered a single soliton with duration $t_0 = 7\,$fs, i.e. approximately 3.8 cycles, and soliton order $N_s = 8$, prepared at a center frequency $\omega = 1.7\,$rad/fs. See Refs. [22,23] for a detailed account of the propagation model and Ref. [24] for a more thorough discussion of the particular problem setup. Note that the chosen parameters relate to values in regions that are very demanding with respect to an adequate post processing, e.g., features have to be resolved that allow for a correct interpretation of the complicated correlation of time and frequency dynamics.

In Fig. 2 we illustrate such an example at a certain propagation distance ($z = 0.12\,$m; cf. Fig. 1). The time–frequency characteristics are illustrated by using a Gaussian window function $h(t, \sigma)$ centered at $t = 0$ and having rms-width σ. Note that the delay time τ has to be interpreted as being relative to the origin of a co-moving frame of reference in which the soliton is initially at rest. Figs. 2(a-1,b-1) demonstrate an inevitable drawback of a trial-and-error choice of a window function used for calculating a spectrogram. As discussed earlier, the properties of the window implies a trade-off in resolution that might be achieved.

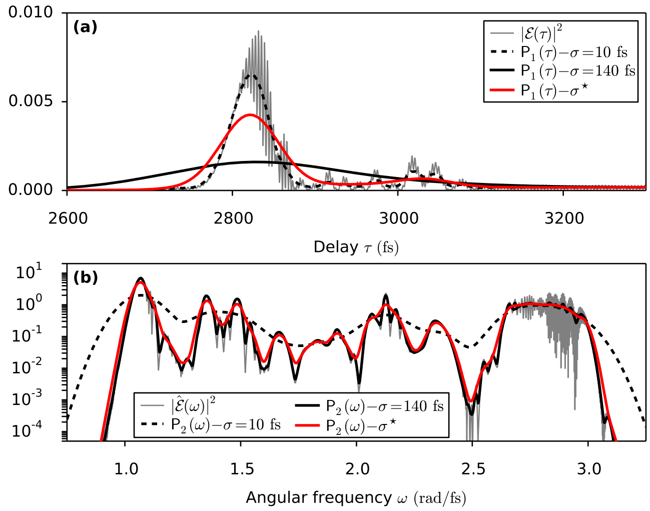

Fig. 3. Assessment of the approximation quality. (a) Comparison of the squared magnitude analytic signal to time marginals obtained from `vanillaFrog`-traces using rms-width $\sigma = 10$ fs (black dashed line), $\sigma = 140$ fs (black solid line), and the `optFrog`-trace ($\sigma^\star = 39.1$ fs; red solid line). (b) Comparison of the squared magnitude analytic signal spectrum to the obtained frequency marginal.

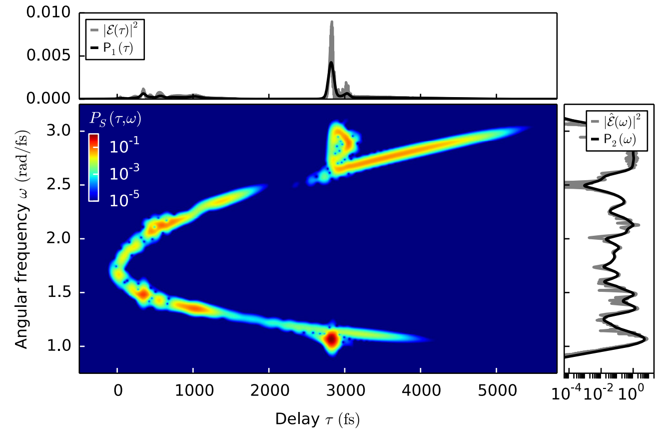

Fig. 4. Analytic signal spectrogram obtained using the balanced `optFrog`-trace for $\sigma^\star = 39.1$ fs.

I.e., if the user opts for a window function that is either too wide or too narrow in comparison to the signal features in the time domain, only one marginal will approximate its underlying original intensity well and, as a result, the spectrogram will appear distorted. This is shown in Fig. 2(a-1), where `vanillaFrog` trace using $\sigma = 10$ fs yields a good time resolution and a bad frequency resolution. Conversely, as evident from Fig. 2(b-1), a `vanillaFrog` trace using $\sigma = 140$ fs exhibits a good frequency resolution but a bad time resolution. To highlight the difficulties for the post-processing we present close-up views on two selected parts of the full spectrogram. In case of, say, Figs. 2(a-3,b-3), typical challenges for pulse characterization are given. It is clear that by using a too narrow window function (e.g. $\sigma = 10$ fs), the good time-resolution yielding 55 fs comes at the expense of a bad frequency-resolution. The converse holds while opting for a too wide window function. This problem gets exceedingly difficult when we are faced with the characterization of pulse interaction processes as shown in Figs. 2(a-2,b-2). In this regard we emphasized the interaction between a soliton and a dispersive wave, demonstrating a temporal reflection in the vicinity of an optical event horizon [25], indicated by the dashed white boxes in Fig. 2. By visual inspection of the close-up views the conflicting appearance is immediate, demonstrating that interpretation cannot be obtained by any empirical approach.

In contrast, if the IAEs of both marginals are minimized simultaneously by aid of a numerical algorithm, both marginals of the optimized spectrogram are found to approximate the original intensities per unit time and frequency similarly well. Consequently, the resulting spectrogram provides a most reasonable time–frequency representation of the underlying signal. To demonstrate this, the balanced ($\alpha = 0$) `optFrog` trace for the optimized window function, obtained for $\sigma^\star = 39.1$ fs with $Q = 0.39$, is shown in Fig. 2(c-1). As evident from the close-up views (subfigures (a-2), (b-2), and (c-2)), a reliable time–frequency resolution is a prerequisite for recognizing the small scale features, e.g. given by nodal points in the spectrogram due to interference of incoming and reflected dispersive wave components, caused by the interaction mechanism.

A direct comparison of the time- and frequency-marginals, where the former is restricted to the delay range in which the abovementioned interaction between the soliton and dispersive wave prevails, is provided by Fig. 3. So as to qualify the approximation quality of either spectrogram, considering an exemplary pulse characterization: the time-domain pulse FWHM varies from 281 fs to 55 fs for σ decreasing from 140 fs to 10 fs, respectively. Further, the FWHM estimate of the spectral range relating to the soliton part of the signal results in the corresponding estimates 0.048 rad/fs and 0.18 rad/fs. Comparing these values to the actual pulse width and spectral range (50 fs, 0.046 rad/fs) indicates that either the time-marginal or the frequency-marginal provides a bad match. In contrast, the presented optimization scheme yields 85 fs and 0.066 rad/fs, adequately resolving both values. As mentioned above, the width of the Gaussian window function results in $\sigma^\star = 39.1$ fs. Considering a function of sech-squared type, resembling the intensity profile of a soliton, the optimization procedure yields $\sigma^\star = 51$ fs in well agreement with the width of the propagating soliton.

Finally, a summarizing figure of the balanced `optFrog` trace obtained using the code listing discussed in subsection is shown in Fig. 4.

4. Impact

Computing reliable spectrograms represents an integral part in the analysis of the characteristics of ultrashort optical pulses. The open-source Python package `optfrog` performs the nontrivial task of computing such spectrograms with optimized time–frequency resolution. It is based on a computational approach to parameter optimization in opposition to common trial-and-error approaches, helping to save time and effort, and yielding reproducible results. The package is aimed at researchers in the field of ultrashort pulse propagation and related disciplines where signal analysis in terms of short-time Fourier transforms is of relevance. As independent software postprocessing tool it is ideally suited for the analysis of output data obtained by existing pulse propagation codes, as, e.g., the open source `LaserFOAM` (Python) [26] and `gnlse` (Matlab) [27] solver for the generalized nonlinear Schrödinger equation.

5. Conclusions

The `optfrog` Python package provides easy-to-use tools that yield a time–frequency representation of a real valued input signal and allows one to quantify how well the resulting spectrogram approximates the signal under scrutiny for a user supplied window function.

We have shown how optfrog can be used to calculate analytic signal based spectrograms that are optimal in the sense that their visual inspection exhibits a minimal amount of distortion, giving a reliable interpretation of the time–frequency composition of the input signal.

The optfrog software tool, including scripts that implement the exemplary use-cases illustrated in Section 3, is available for download and installation under Ref. [17].

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.softx.2019.100275.

Acknowledgments

We acknowledge support from the Deutsche Forschungsgemeinschaft (DFG), Germany under Germany's Excellence Strategy within the Cluster of Excellence PhoenixD (Photonics, Optics, and Engineering — Innovation Across Disciplines) (EXC 2122, projectID 390833453). The publication of this article was funded by the Open Access Fund of the Leibniz Universität Hannover.

References

[1] Cohen L. Time-frequency distributions – A review. Proc IEEE 1989;77:941–81.
[2] Kane DJ, Trebino R. Characterization of arbitrary fremtosecond pulses using frequency-resolved optical gating. IEEE J Quantum Electron 1993;29:571.
[3] Linden S, Kuhl J, Giessen H. XFROG—Cross-correlation frequency-resolved optical gating. In: Frequency-resolved optical gating: the measurement of ultrashort laser pulses. 2000, p. 313–22, in Trebino [4].
[4] Trebino R, editor. Frequency-resolved optical gating: The measurement of ultrashort laser pulses. Boston, MA: Springer US; 2000.
[5] Efimov A, Yulin AV, Skryabin DV, Knight JC, Joly N, Omenetto FG, et al. Interaction of an optical soliton with a dispersive wave. Phys Rev Lett 2005;95. 213902.
[6] Dudley JM, Gu X, Xu L, Kimmel M, Zeek E, O'Shea P, et al. Cross-correlation frequency resolved optical gating analysis of broadband continuum generation in photonic crystal fiber: simulations and experiments. Opt Express 2002;10(21):1215–21.
[7] Skryabin DV, Yulin AV. Theory of generation of new frequencies by mixing of solitons and dispersive waves in optical fibers. Phys Rev E 2005;72. 016619.
[8] Guo H, Wang S, Zeng X, Bache M. Understanding soliton spectral tunneling as a spectral coupling effect. IEEE Photonics Technol Lett 2013;25(19):1928.
[9] Cohen L, Loughlin PJ. The marginals and time-frequency distributions. In Proc. SPIE 5205, Advanced signal processing algorithms, architectures, and implementations XIII; 2003. p. 59.
[10] Dudley JM, Genty G, Coen S. Supercontinuum generation in photonic crystal fiber. Rev Modern Phys 2006;78:1135–84.
[11] Gabor D. Acoustical quanta and the theory of hearing. Nature 1947;159:591–4.
[12] Marple SL. Computing the discrete-time analytic signal via FFT. IEEE Trans Signal Process 1999;47:2600–3.
[13] Alternatively, the optical pulse might be expressed in terms of its Wigner distribution [1], i.e. a quasi-probability distribution having the correct time and frequency marginals already. However, due to its similarity to the XFROG-traces retrieved from experiments in order to analyse optical pulses, our discussion here is focused on the spectrogram.
[14] van Rossum G, Warsaw B, Coghlan N. PEP8 – Style guide for python code; 2001, https://www.python.org/dev/peps/pep-0008, [Online; Accessed 27 June 2018].
[15] Rossum G. Python reference manual. Technical report, Amsterdam, The Netherlands, The Netherlands; 1995.
[16] Jones E, Oliphant T, Peterson P, et al. Scipy: Open source scientific tools for python; 2001–2018, http://www.scipy.org/, [Online; Accessed 4 June 2018].
[17] Melchert O. Optfrog – Analytic signal spectrograms with optimized time-frequency resolution; 2018, https://github.com/omelchert/optfrog.git, [Online; Accessed 10 April 2019].
[18] Demircan A, Bandelow U. Analysis of the interplay between soliton fission and modulation instability in supercontinuum generation. Appl Phys B 2007;86:31–9.
[19] Birks TA, Knight JC, St. J. Russell P. Endlessly single-mode photonic crystal fiber. Opt Lett 1997;13:961–3.
[20] Stone JM, Knight JC. Visibly 'white' light generation in uniform photonic crystal fiber using a microchip laser. Opt Express 2007;16:2670.
[21] Hollenbeck D, Cantrell CD. Multiple-vibrational-mode model for fiber-optic Raman gain spectrum and response function. J Opt Soc Amer B 2002;19:2886.
[22] Amiranashvili Sh, Demircan A. Hamiltonian structure of propagation equations for ultrashort optical pulses. Phys Rev A 2010;82. 013812.
[23] Sh Amiranashvili, Demircan A. Ultrashort optical pulse propagation in terms of analytic signal. Adv Opt Technol 2011;2011. 989515.
[24] Melchert O, Morgner U, Roth B, Babushkin I, Demircan A. Accurate propagation of ultrashort pulses in nonlinear waveguides using propagation models for the analytic signal. In Proc. SPIE 10694, computational optics II; 2018. p. 106940M.
[25] Demircan A, Amiranashvili Sh, Brée C, Morgner U, Steinmeyer G. Adjustable pulse compression scheme for generation of few-cycle pulses in the midinfrared. Opt Lett 2014;39(9):2735–8.
[26] Amorim AA, Tognetti MV, Oliveira P, Silva JL, Bernardo LM, Kärtner FX, Crespo HM. Sub-two-cycle pulses by soliton self-compression in highly nonlinear photonic crystal fibers. Opt Lett 2009;34(24):3851–3.
[27] Travers JC, Frosz MH, Dudley JM. Nonlinear fibre optics overview. In: Supercontinuum generation in optical fibers. Cambridge University Press; 2010.