

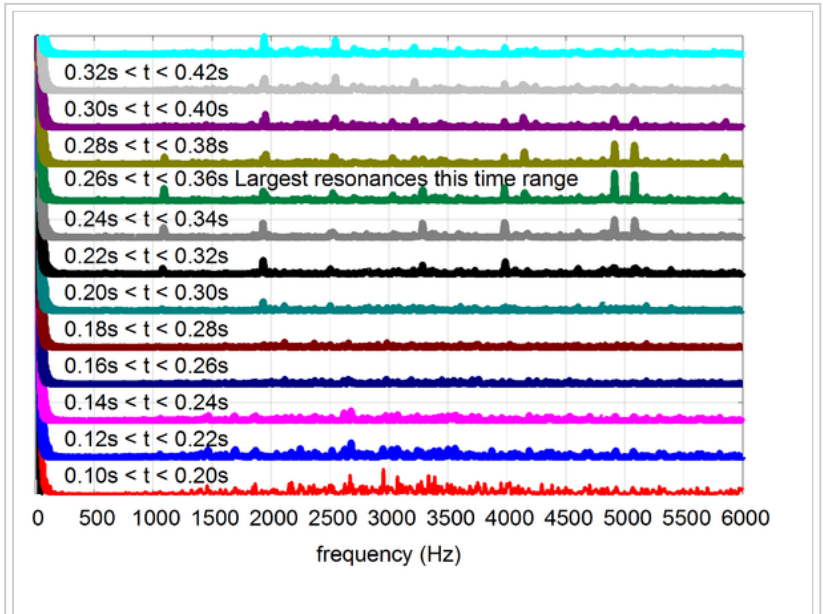
Short-time Fourier transform

From Wikipedia, the free encyclopedia

The **short-time Fourier transform** (**STFT**), or alternatively **short-term Fourier transform**, is a Fourier-related transform used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time.^[1] In practice, the procedure for computing STFTs is to divide a longer time signal into shorter segments of equal length and then compute the Fourier transform separately on each shorter segment. This reveals the Fourier spectrum on each shorter segment. One then usually plots the changing spectra as a function of time.

Contents

- 1 STFT
 - 1.1 Continuous-time STFT
 - 1.2 Discrete-time STFT
 - 1.2.1 Sliding DFT
- 2 Inverse STFT
 - 2.1 Continuous-time STFT
- 3 Resolution issues
 - 3.1 Example
 - 3.2 Explanation
- 4 Rayleigh frequency
- 5 Application
- 6 Implementation
 - 6.1 I. Direct implementation
 - 6.1.1 Constraints
 - 6.2 II. FFT-based method
 - 6.2.1 Constraint
 - 6.3 III. Recursive method
 - 6.3.1 Constraint
 - 6.4 IV. Chirp Z transform
 - 6.4.1 Constraint
 - 6.5 Implementation comparison
- 7 See also
- 8 References
- 9 External links



Example of short time Fourier transforms used to determine time of impact from audio signal.

STFT

Continuous-time STFT

Simply, in the continuous-time case, the function to be transformed is multiplied by a window function which is nonzero for only a short period of time. The Fourier transform (a one-dimensional function) of the resulting signal is taken as the window is slid along the time axis, resulting in a two-dimensional representation of the signal. Mathematically, this is written as:

$$\mathbf{STFT}\{x(t)\}(\tau, \omega) \equiv X(\tau, \omega) = \int_{-\infty}^{\infty} x(t)w(t - \tau)e^{-j\omega t} dt$$

where $w(t)$ is the window function, commonly a Hann window or Gaussian window centered around zero, and $x(t)$ is the signal to be transformed. (Note the difference between w and ω .) $X(\tau, \omega)$ is essentially the Fourier Transform of $x(t)w(t-\tau)$, a complex function representing the phase and magnitude of the signal over time and frequency. Often phase unwrapping is employed along either or both the time axis, τ , and frequency axis, ω , to suppress any jump discontinuity of the phase result of the STFT. The time index τ is normally considered to be "slow" time and usually not expressed in as high resolution as time t .

Discrete-time STFT

In the discrete time case, the data to be transformed could be broken up into chunks or frames (which usually overlap each other, to reduce artifacts at the boundary). Each chunk is Fourier transformed, and the complex result is added to a matrix, which records magnitude and phase for each point in time and frequency. This can be expressed as:

$$\mathbf{STFT}\{x[n]\}(m, \omega) \equiv X(m, \omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n}$$

likewise, with signal $x[n]$ and window $w[n]$. In this case, m is discrete and ω is continuous, but in most typical applications the STFT is performed on a computer using the Fast Fourier Transform, so both variables are discrete and quantized.

The magnitude squared of the STFT yields the spectrogram of the function:

$$\mathbf{spectrogram}\{x(t)\}(\tau, \omega) \equiv |X(\tau, \omega)|^2$$

See also the modified discrete cosine transform (MDCT), which is also a Fourier-related transform that uses overlapping windows.

Sliding DFT

If only a small number of ω are desired, or if the STFT is desired to be evaluated for every shift m of the window, then the STFT may be more efficiently evaluated using a sliding DFT algorithm.^[2]

Inverse STFT

The STFT is invertible, that is, the original signal can be recovered from the transform by the Inverse STFT. The most widely accepted way of inverting the STFT is by using the overlap-add (OLA) method, which also allows for modifications to the STFT complex spectrum. This makes for a versatile signal processing method,^[3] referred to as the *overlap and add with modifications* method.

Continuous-time STFT

Given the width and definition of the window function $w(t)$, we initially require the area of the window function to be scaled so that

$$\int_{-\infty}^{\infty} w(\tau) d\tau = 1.$$

It easily follows that

$$\int_{-\infty}^{\infty} w(t - \tau) d\tau = 1 \quad \forall t$$

and

$$x(t) = x(t) \int_{-\infty}^{\infty} w(t - \tau) d\tau = \int_{-\infty}^{\infty} x(t)w(t - \tau) d\tau.$$

The continuous Fourier Transform is

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt.$$

Substituting $x(t)$ from above:

$$\begin{aligned} X(\omega) &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} x(t)w(t - \tau) d\tau \right] e^{-j\omega t} dt \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x(t)w(t - \tau) e^{-j\omega t} d\tau dt. \end{aligned}$$

Swapping order of integration:

$$\begin{aligned} X(\omega) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x(t)w(t - \tau) e^{-j\omega t} dt d\tau \\ &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} x(t)w(t - \tau) e^{-j\omega t} dt \right] d\tau \\ &= \int_{-\infty}^{\infty} X(\tau, \omega) d\tau. \end{aligned}$$

So the Fourier Transform can be seen as a sort of phase coherent sum of all of the STFTs of $x(t)$. Since the inverse Fourier transform is

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)e^{+j\omega t} d\omega,$$

then $x(t)$ can be recovered from $X(\tau, \omega)$ as

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} X(\tau, \omega)e^{+j\omega t} d\tau d\omega.$$

or

$$x(t) = \int_{-\infty}^{\infty} \left[\frac{1}{2\pi} \int_{-\infty}^{\infty} X(\tau, \omega)e^{+j\omega t} d\omega \right] d\tau.$$

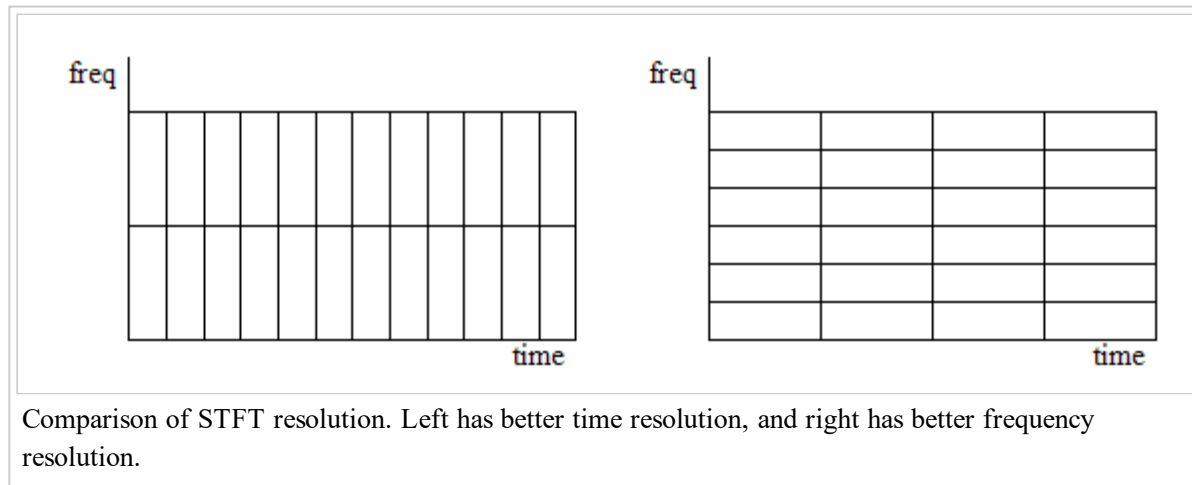
It can be seen, comparing to above that windowed "grain" or "wavelet" of $x(t)$ is

$$x(t)w(t - \tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\tau, \omega)e^{+j\omega t} d\omega.$$

the inverse Fourier transform of $X(\tau, \omega)$ for τ fixed.

Resolution issues

One of the pitfalls of the STFT is that it has a fixed resolution. The width of the windowing function relates to how the signal is represented—it determines whether there is good frequency resolution (frequency components close together can be separated) or good time resolution (the time at which frequencies change). A wide window gives better frequency resolution but poor time resolution. A narrower window gives good time resolution but poor frequency resolution. These are called narrowband and wideband transforms, respectively.



This is one of the reasons for the creation of the wavelet transform and multiresolution analysis, which can give good time resolution for high-frequency events and good frequency resolution for low-frequency events, the combination best suited for many real signals.

This property is related to the Heisenberg uncertainty principle, but not directly – see Gabor limit for discussion. The product of the standard deviation in time and frequency is limited. The boundary of the uncertainty principle (best simultaneous resolution of both) is reached with a Gaussian window function, as the Gaussian minimizes the Fourier uncertainty principle. This is called the Gabor transform (and with modifications for multiresolution becomes the Morlet wavelet transform).

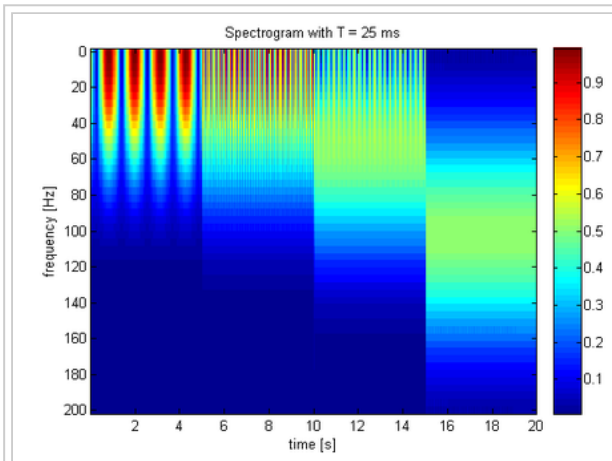
One can consider the STFT for varying window size as a two-dimensional domain (time and frequency), as illustrated in the example below, which can be calculated by varying the window size. However, this is no longer a strictly time–frequency representation – the kernel is not constant over the entire signal.

Example

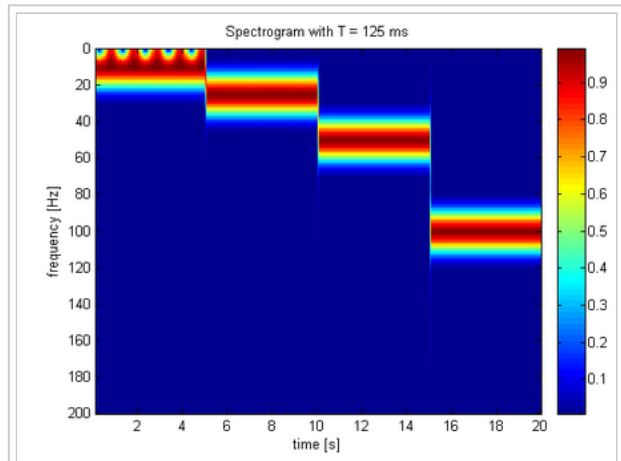
Using the following sample signal $x(t)$ that is composed of a set of four sinusoidal waveforms joined together in sequence. Each waveform is only composed of one of four frequencies (10, 25, 50, 100 Hz). The definition of $x(t)$ is:

$$x(t) = \begin{cases} \cos(2\pi 10t) & 0 \text{ s} \leq t < 5 \text{ s} \\ \cos(2\pi 25t) & 5 \text{ s} \leq t < 10 \text{ s} \\ \cos(2\pi 50t) & 10 \text{ s} \leq t < 15 \text{ s} \\ \cos(2\pi 100t) & 15 \text{ s} \leq t < 20 \text{ s} \end{cases}$$

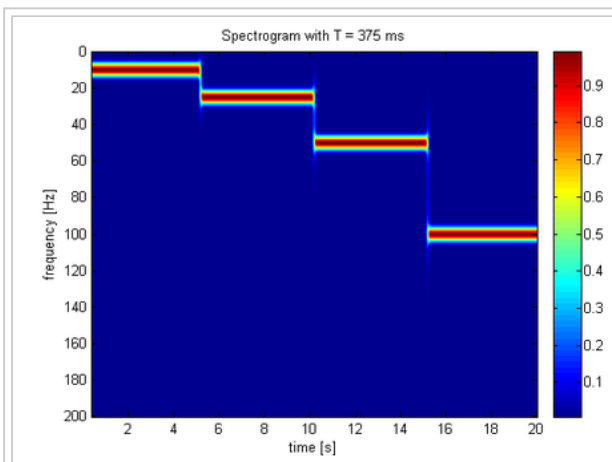
Then it is sampled at 400 Hz. The following spectrograms were produced:



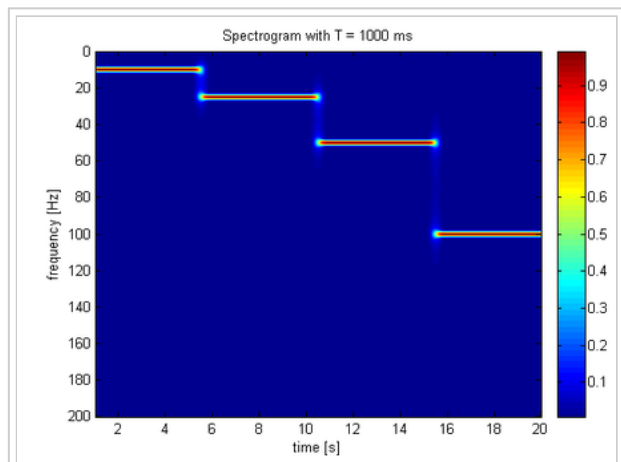
25 ms window



125 ms window



375 ms window



1000 ms window

The 25 ms window allows us to identify a precise time at which the signals change but the precise frequencies are difficult to identify. At the other end of the scale, the 1000 ms window allows the frequencies to be precisely seen but the time between frequency changes is blurred.

Explanation

It can also be explained with reference to the sampling and Nyquist frequency.

Take a window of N samples from an arbitrary real-valued signal at sampling rate f_s . Taking the Fourier transform produces N complex coefficients. Of these coefficients only half are useful (the last $N/2$ being the complex conjugate of the first $N/2$ in reverse order, as this is a real valued signal).

These $N/2$ coefficients represent the frequencies 0 to $f_s/2$ (Nyquist) and two consecutive coefficients are spaced apart by f_s/N Hz.

To increase the frequency resolution of the window the frequency spacing of the coefficients needs to be reduced. There are only two variables, but decreasing f_s (and keeping N constant) will cause the window size to increase — since there are now fewer samples per unit time. The other alternative is to increase N , but this again causes the window size to increase. So any attempt to increase the frequency resolution causes a larger window size and therefore a reduction in time resolution—and vice versa.

Rayleigh frequency

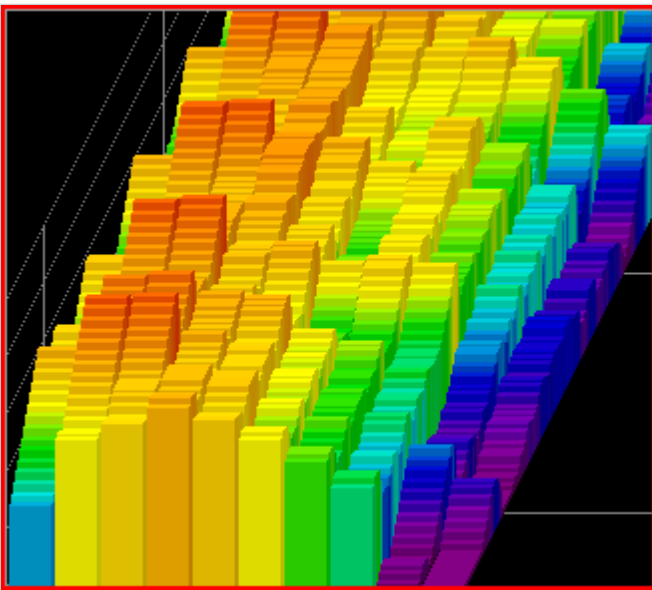
As the Nyquist frequency is a limitation in the maximum frequency that can be meaningfully analysed, so is the Rayleigh frequency a limitation on the minimum frequency.

Rayleigh frequency is the minimum frequency that can be resolved by a finite duration time window.^{[4][5]}

Given a time window that is T seconds long, the minimum frequency that can be resolved is $1/T$ Hz.

Rayleigh frequency is important to consider in applications of the short-time Fourier transform (STFT), such as in analysing neural signals^{[6][7]}

Application



An STFT being used to analyze an audio signal across time

STFTs as well as standard Fourier transforms and other tools are frequently used to analyze music. The spectrogram can, for example, show frequency on the horizontal axis, with the lowest frequencies at left, and the highest at the right. The height of each bar (augmented by color) represents the amplitude of the frequencies within that band. The depth dimension represents time, where each new bar was a separate distinct transform. Audio engineers use this kind of visual to gain information about an audio sample, for example, to locate the frequencies of specific noises (especially when used with greater frequency resolution) or to find frequencies which may be more or less resonant in the space where the signal was recorded. This information can be used for equalization or tuning other audio effects.

Implementation

Original function:
$$X(t, f) = \int_{-\infty}^{\infty} w(t - \tau)x(\tau)e^{-j2\pi f\tau} d\tau$$

Converting into the discrete form:

$$t = n\Delta_t, f = m\Delta_f, \tau = p\Delta_t$$

$$X(n\Delta_t, m\Delta_f) = \sum_{-\infty}^{\infty} w((n - p)\Delta_t)x(p\Delta_t)e^{-j2\pi pm\Delta_t\Delta_f} \Delta_t$$

Suppose that $w(t) \cong 0$ for $|t| > B$, $\frac{B}{\Delta_t} = Q$

Then we can write the original function into

$$X(n\Delta_t, m\Delta_f) = \sum_{p=n-Q}^{n+Q} w((n-p)\Delta_t)x(p\Delta_t)e^{-j2\pi pm\Delta_t\Delta_f\Delta_t}$$

I. Direct implementation

Constraints

a. Nyquist criterion(Avoiding the aliasing effect):

$$\Delta_t < \frac{1}{2\Omega}, \Omega \text{ is the bandwidth of } x(\tau)w(t-\tau)$$

II. FFT-based method

Constraint

a. $\Delta_t\Delta_f = \frac{1}{N}$, N is an integer

b. $N \geq 2Q + 1$

c. Nyquist criterion(Avoiding the aliasing effect):

$$\Delta_t < \frac{1}{2\Omega}, \Omega \text{ is the bandwidth of } x(\tau)w(t-\tau)$$

$$X(n\Delta_t, m\Delta_f) = \sum_{p=n-Q}^{n+Q} w((n-p)\Delta_t)x(p\Delta_t)e^{\frac{-j2\pi pm}{N}\Delta_t}$$

$$\text{for } q = p - (n - Q) \rightarrow p = (n - Q) + q$$

$$X(n\Delta_t, m\Delta_f) = \Delta_t e^{\frac{j2\pi(Q-n)m}{N}} \sum_{q=0}^{N-1} x_1(q) e^{\frac{-j2\pi qm}{N}}$$

$$\text{where } \begin{cases} x_1(q) = w((Q-q)\Delta_t)x((n-Q+q)\Delta_t) & \text{for } 0 \leq q \leq 2Q \\ x_1(q) = 0 & \text{for } 2Q < q < N \end{cases}$$

III. Recursive method

Constraint

a. $\Delta_t \Delta_f = \frac{1}{N}$, N is an integer

b. $N \geq 2Q + 1$

c. Nyquist criterion(Avoiding the aliasing effect):

$$\Delta_t < \frac{1}{2\Omega}, \Omega \text{ is the bandwidth of } x(\tau)w(t - \tau)$$

d. Only for implementing the rectangular-STFT

when $w((n - p)\Delta_t) = 1$

$$X(n\Delta_t, m\Delta_f) = \sum_{p=n-Q}^{n+Q} x(p\Delta_t) e^{\frac{-j2\pi pm}{N} \Delta_t}$$

$$X((n - 1)\Delta_t, m\Delta_f) = \sum_{p=n-1-Q}^{n-1+Q} x(p\Delta_t) e^{\frac{-j2\pi pm}{N} \Delta_t}$$

Calculate $X(\min(n) \Delta_t, m\Delta_f)$ by the N-point FFT:

$$X(n_0\Delta_t, m\Delta_f) = \Delta_t e^{\frac{j2\pi(Q-n_0)m}{N}} \sum_{q=0}^{N-1} x_1(q) e^{\frac{-j2\pi qm}{N}}, \quad n_0 = \min(n)$$

$$\begin{cases} x_1(q) = x((n - Q + q)\Delta_t) & \text{for } q \leq 2Q \\ x_1(q) = 0 & \text{for } q > 2Q \end{cases}$$

Applying the recursive formula to calculate $X(n\Delta_t, m\Delta_f)$

$$X(n\Delta_t, m\Delta_f) = X((n - 1)\Delta_t, m\Delta_f) - x((n - Q - 1)\Delta_t) e^{\frac{-j2\pi(n-Q-1)m}{N} \Delta_t} + x((n + Q)\Delta_t) e^{\frac{-j2\pi(n+Q)m}{N} \Delta_t}$$

IV. Chirp Z transform

Constraint

a. $\exp(-j2\pi pm\Delta_t\Delta_f) = \exp(-j\pi p^2\Delta_t\Delta_f) \exp(j\pi(p - m)^2\Delta_t\Delta_f) \exp(-j\pi m^2\Delta_t\Delta_f)$

$$X(n\Delta_t, m\Delta_f) = \Delta_t \sum_{p=n-Q}^{n+Q} w((n - p)\Delta_t) x(p\Delta_t) e^{-j2\pi pm\Delta_t\Delta_f}$$

$$X(n\Delta_t, m\Delta_f) = \Delta_t e^{-j2\pi m^2\Delta_t\Delta_f} \sum_{p=n-Q}^{n+Q} w((n - p)\Delta_t) x(p\Delta_t) e^{-j\pi p^2\Delta_t\Delta_f} e^{j\pi(p-m)^2\Delta_t\Delta_f}$$

Implementation comparison

Method	Complexity
Direct implementation	$O(TFQ)$
FFT-based	$O(TN \log_2 N)$
Recursive	$O(TF)$
Chirp Z transform	$O(TN \log_2 N)$


See also

- Spectral density estimation
- Time-frequency representation
- Reassignment method

Other time-frequency transforms:

- Cone-shape distribution function
- Constant-Q transform
- Fractional Fourier transform
- Gabor transform
- Newland transform
- S transform
- Wavelet transform
- Chirplet transform

References

1. Sejdić E.; Djurović I.; Jiang J. (2009). "Time-frequency feature representation using energy concentration: An overview of recent advances". *Digital Signal Processing*. **19** (1): 153–183. doi:10.1016/j.dsp.2007.12.004.
2. E. Jacobsen and R. Lyons, The sliding DFT (http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1184347), *Signal Processing Magazine* vol. 20, issue 2, pp. 74–80 (March 2003).
3. Jont B. Allen (June 1977). "Short Time Spectral Analysis, Synthesis, and Modification by Discrete Fourier Transform". *IEEE Transactions on Acoustics, Speech, and Signal Processing*. ASSP-25 (3): 235–238.
4. <https://physics.ucsd.edu/neurophysics/publications/Cold%20Spring%20Harb%20Protoc-2014-Kleinfeld-pdb.top081075.pdf>
5. http://fieldtrip.fcdonders.nl/faq/what_does_padding_not_sufficient_for_requested_frequency_resolution_mean
6. Zeitler M, Fries P, Gielen S (2008). "Biased competition through variations in amplitude of gamma-oscillations". *J Comput Neurosci*. **25**: 89–107. doi:10.1007/s10827-007-0066-2. PMC 2441488  PMID 18293071.
7. <http://www.jneurosci.org/content/30/20/7078.full>

External links

- DiscreteTFDs – software for computing the short-time Fourier transform and other time-frequency distributions (<http://tfd.sourceforge.net/>)
- Singular Spectral Analysis - MultiTaper Method Toolkit (<http://www.atmos.ucla.edu/tcd/ssa/>) - a free software program to analyze short, noisy time series.
- kSpectra Toolkit for Mac OS X from SpectraWorks (<http://www.spectraworks.com>)
- Time stretched short time Fourier transform for time frequency analysis of ultra wideband signals (http://www.researchgate.net/publication/3091384_Time-stretched_short-time_Fourier_transform)
- A BSD-licensed Matlab class to perform STFT and inverse STFT (<http://www.mathworks.fr/matlabcentral/fileexchange/33451-stft-mdct-and-inverses-onset-and-pitch-detection>)
- LTFAT - A free (GPL) Matlab / Octave toolbox to work with short-time Fourier transforms and time-frequency analysis (<http://lftfat.sourceforge.net/>)

Retrieved from "https://en.wikipedia.org/w/index.php?title=Short-time_Fourier_transform&oldid=744863164"

Categories: [Fourier analysis](#) | [Time–frequency analysis](#) | [Transforms](#) | [Signal processing](#)

- This page was last modified on 17 October 2016, at 22:01.
- Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.