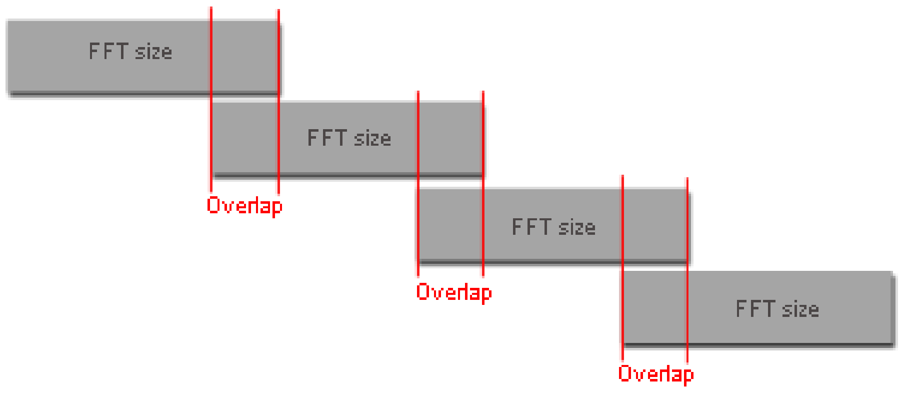
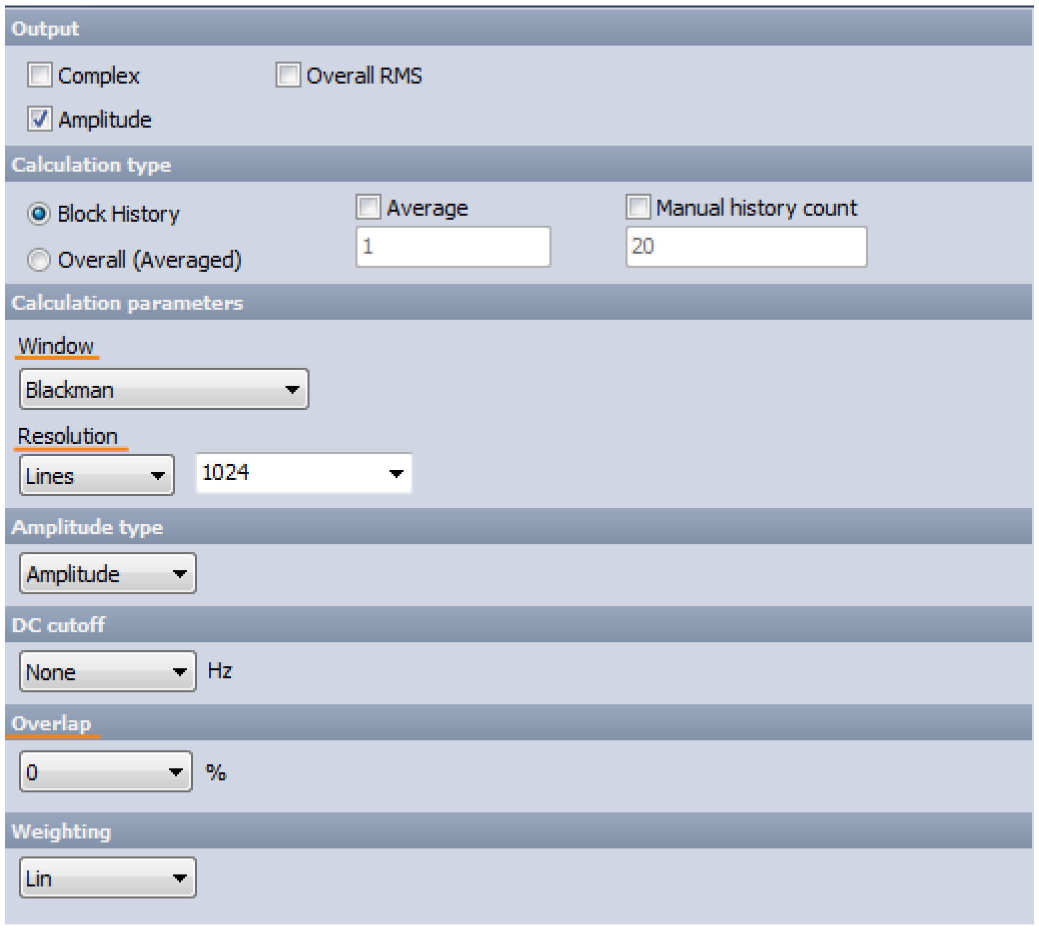
Short Time Fourier Transform

STFT is a well known technique in signal processing to analyze non-stationary signals. STFT is segmenting the signal into narrow time intervals and takes the Fourier transform of each segment. In Dewesoft's FFT setup you can set FFT's resolution, Window and Overlap and for better understanding what that means, lets look at the picture below. Window size depends on FFT's resolution, we can just say FFT size (representing segment of a signal).





Output: [ ] Complex, [ ] Overall RMS, [✓] Amplitude

Calculation type: (●) Block History, ( ) Overall (Averaged); [ ] Average: 1; [ ] Manual history count: 20

Window: Blackman

Resolution: Lines, 1024

Amplitude type: Amplitude

DC cutoff: None Hz

Overlap: 0 %

Weighting: Lin

Why use STFT? Relationship between acquisition time T and frequency resolution df is

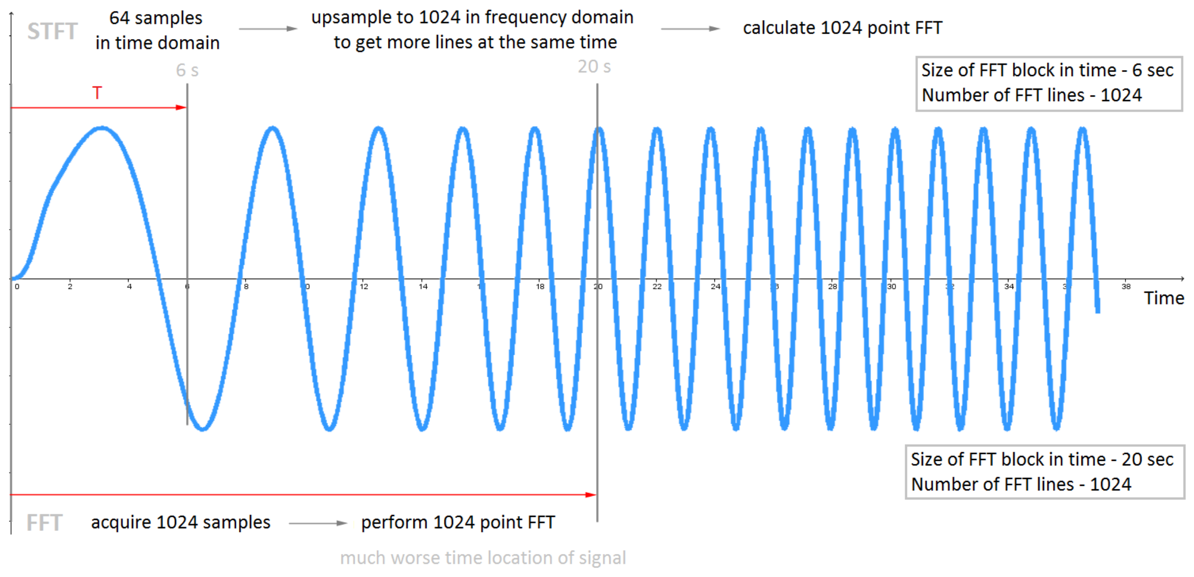
$$\frac{1}{T} = df,$$

where $df = \frac{sample\ rate}{2 \cdot FFTsize}$. Smaller time frame, T, will result in poorer frequency resolution (bigger df). When signal changes fast, you need small T to calculate frequency spectrum faster. But you still want better frequency resolution and here STFT comes in place.

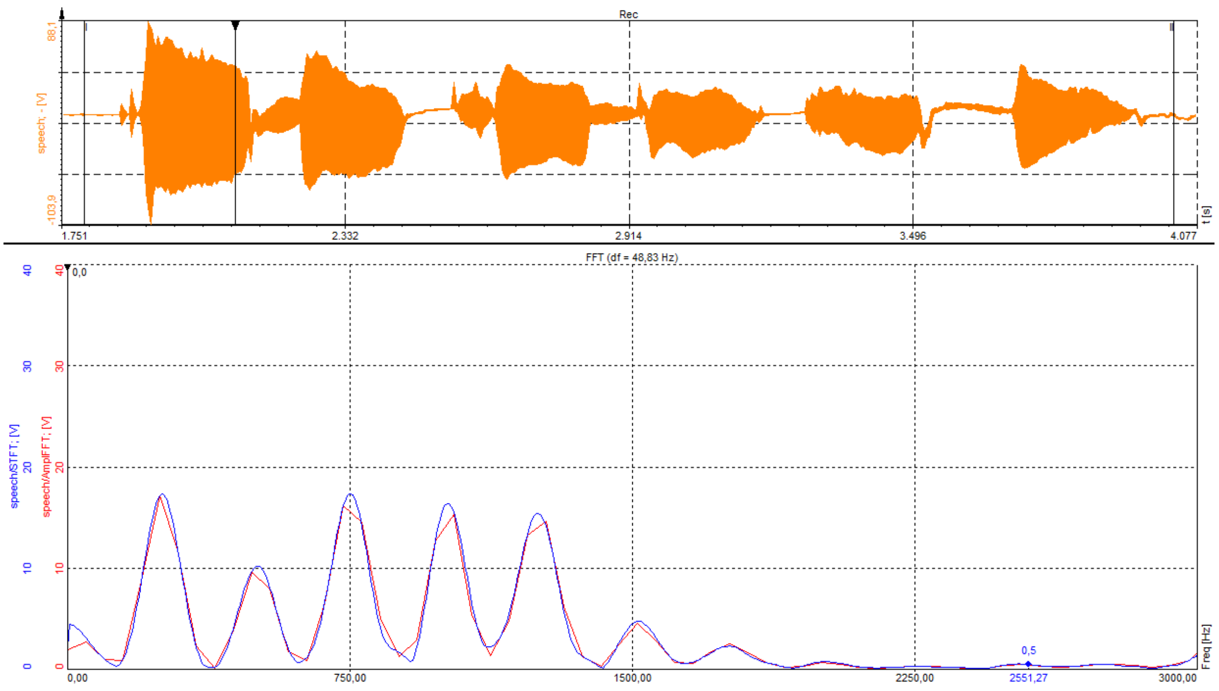
STFT takes $Block\ size$ long segment of signal in time domain, let's say $Block\ size = 64$ and $sample\ rate = 2000$. Currently frequency resolution would be $df = \frac{2000}{2 \cdot 64} = 15{,}625$. Now we will use desired $FFT\ size$, that you set in STFT setup, to improve df, take $FFT\ size = 1024$. We employ frequency interpolation and obtain $FFT\ size$ long block. We can say, frequency interpolation in the time domain results in an increased sampling rate in the frequency domain. Here frequency interpolation increased our frequency-domain sampling (resolution) by a factor of 16 $(1024/64)$, i.e.

$$df = \frac{2000}{2 \cdot 64 \cdot 16} = \frac{2000}{2 \cdot 1024} = 0{,}977.$$

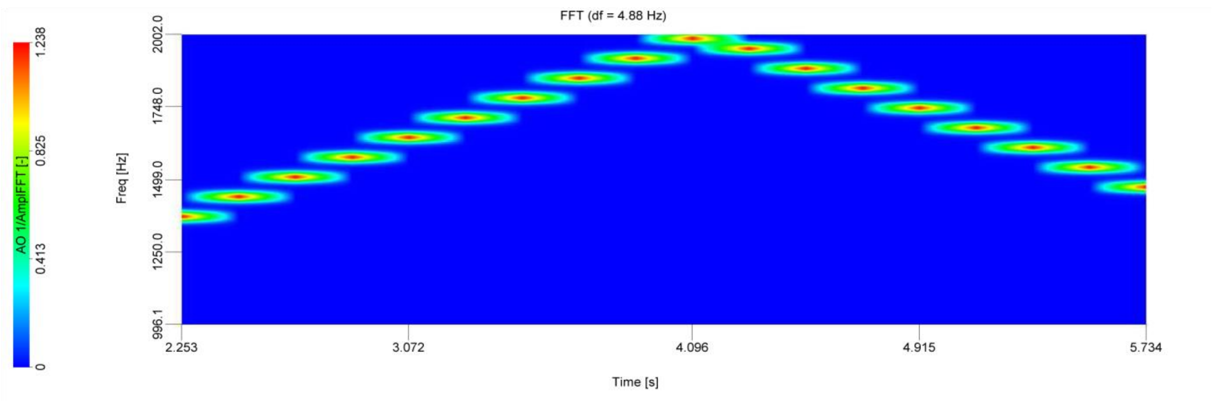
We get 16 times better frequency resolution for the same time frame.

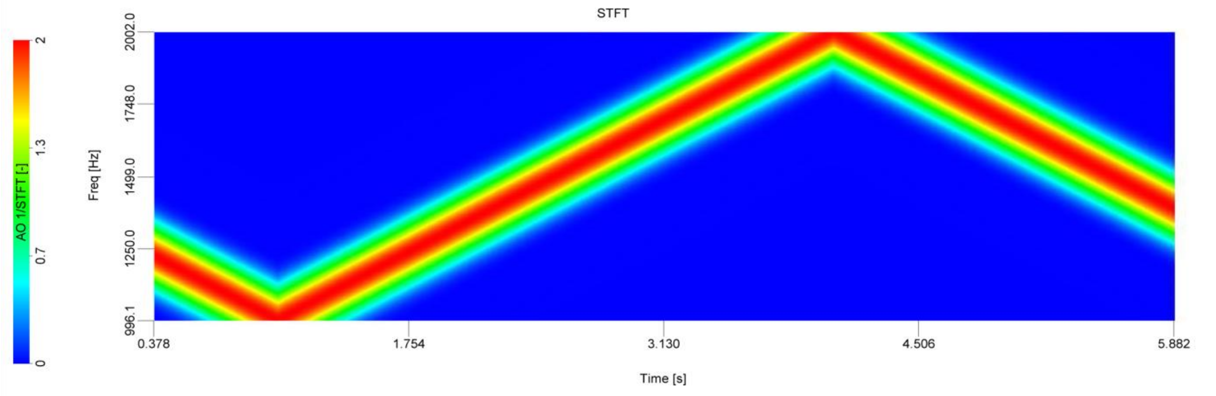

STFT
64 samples in time domain
upsample to 1024 in frequency domain to get more lines at the same time
calculate 1024 point FFT
6 s
20 s
T
Size of FFT block in time - 6 sec
Number of FFT lines - 1024
Time
Size of FFT block in time - 20 sec
Number of FFT lines - 1024
FFT
acquire 1024 samples
perform 1024 point FFT
much worse time location of signal

For example, FFT (red) and STFT (blue) of speech waves are shown below. FFT has resolution of 2048 lines, Blackman window and 50% overlap and STFT also has Block size 2048, FFT size 16K, Blackman window used and 50% overlap. As we can see, STFT performs better with the same block size (but more calculated lines). We improved frequency resolution for the same amount of scooped data. In most cases here, FFT does not strike center frequencies (peaks), which are usually wanted result.



Next example compares, how STFT and FFT performs with signal, where frequency changes fast over time. Here, frequency changes in a loop from 1000 Hz to 2000 Hz in 3 seconds. On first 3D graph is used FFT (1024 lines) and we can see, that we have gaps between frequency spectrums, but in measured data frequency changes continuous over time. Second 3D graph shows STFT ($\text{Block size} = 64$, $\text{FFT size} = 1024$ and 0% overlap), where there are no gaps and frequency traces are connected. If you look closely, there is a difference in the time frame on 3D graphs between STFT and FFT. STFT has smaller time frames, consequently, frequency spectrum moves smoother over time, therefore it is more accurate.


FFT (df = 4.88 Hz)
AO 1/AmplFFT [-]
Freq [Hz]
Time [s]


STFT
AO 1/STFT [-]
Freq [Hz]
Time [s]