SATFD lab 03

Dawid Karpiński, 12.04.2024 r.

1 The main goals

- Use a short-time Fourier transform to analyze signals.
- Perform a Wigner-Ville transform and compare with STFT.

2 Chirp signal

Chirp is a signal of a time-dependent frequency.

The signal (fig. 1) I have generated for further analysis has its frequency changing linearly [1] as

$$f(t) = f_0 + (f_1 - f_0) * \frac{t}{t_1}.$$

The parameters I have used throughout the analysis:

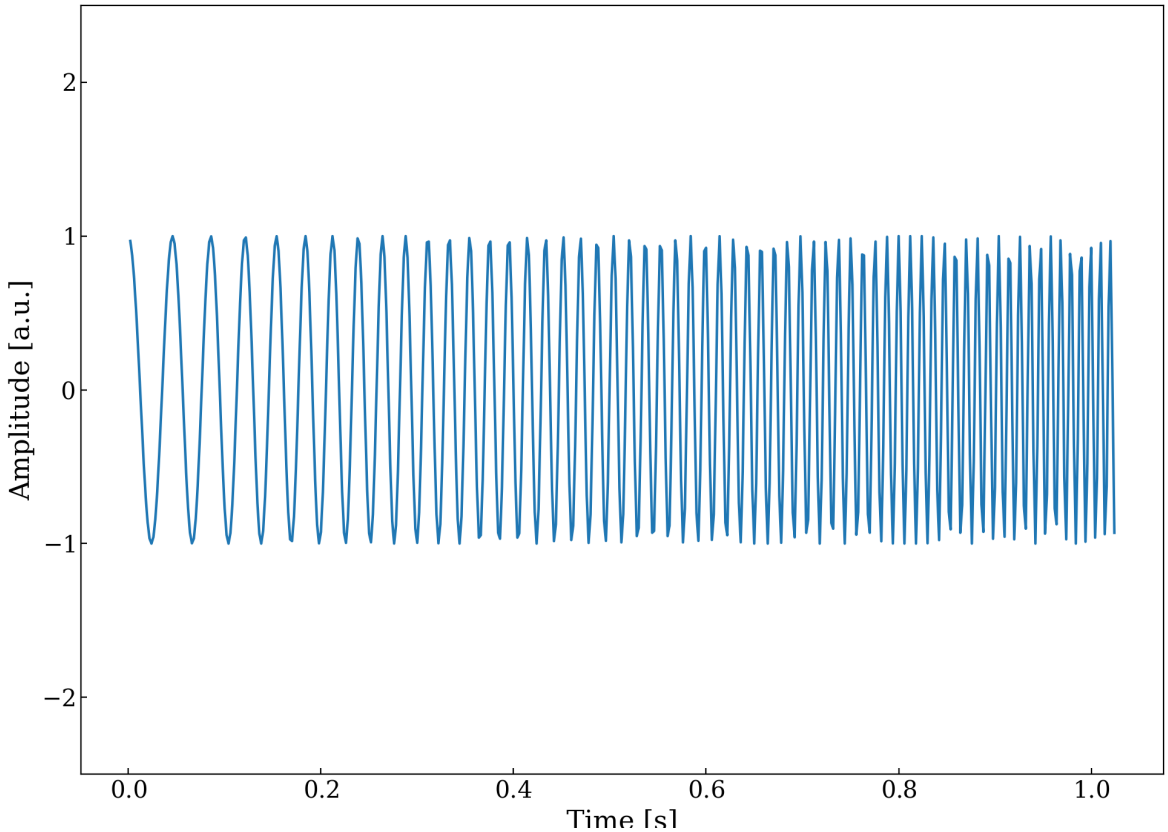
- sampling rate: $fs = 500$ [Hz],
- initial frequency: $f_0 = 20$ [Hz],
- final frequency: $f_1 = 100$ [Hz],
- time at which f_1 is specified: $t = N/500$ [s],
- number of samples N.

Listing 1: **Snippet with a custom function to generate the chirp signal with given parameters.**

```
def get_chirp(*, N: int, f0=20, f1=100):
    time = np.arange(1, N + 1, step=1) / fs
    wave = signal.chirp(time, f0=f0, t1=N / fs, f1=f1)
    return time, wave

fs = 500
time, wave = get_chirp(N=512)
```

Figure 1: **Chirp signal.**



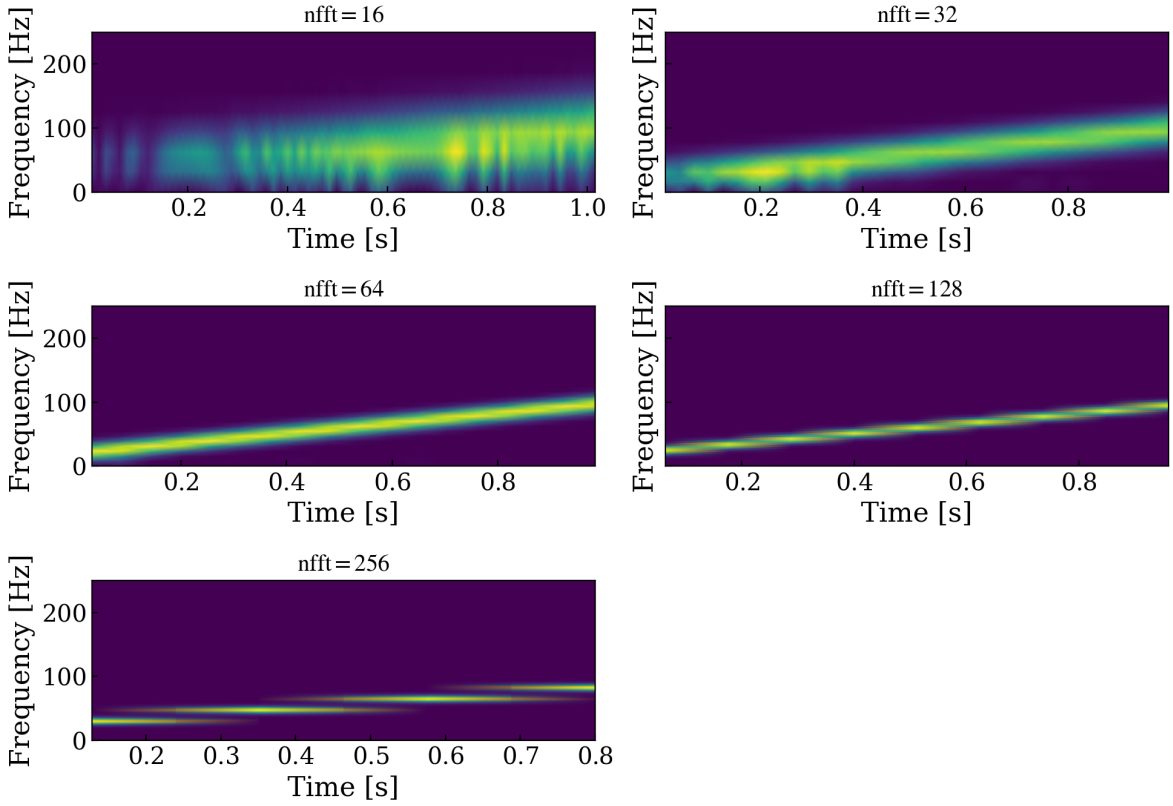
3 Spectrogram

The following spectrograms of the prepared chirp signal showcase the manifestation of Heisenberg's uncertainty principle in signal analysis.

Listing 2: **Snippet for generating spectrogram with different nfft values.**

```
for nfft in [16, 32, 64, 128, 256]:
    _, wave = get_chirp(N=512)
    freqs, times, spect = signal.spectrogram(
        wave,
        fs=fs,
        window=("hamming"),
        nperseg=nfft // 2,
        nfft=nfft,
    )
```

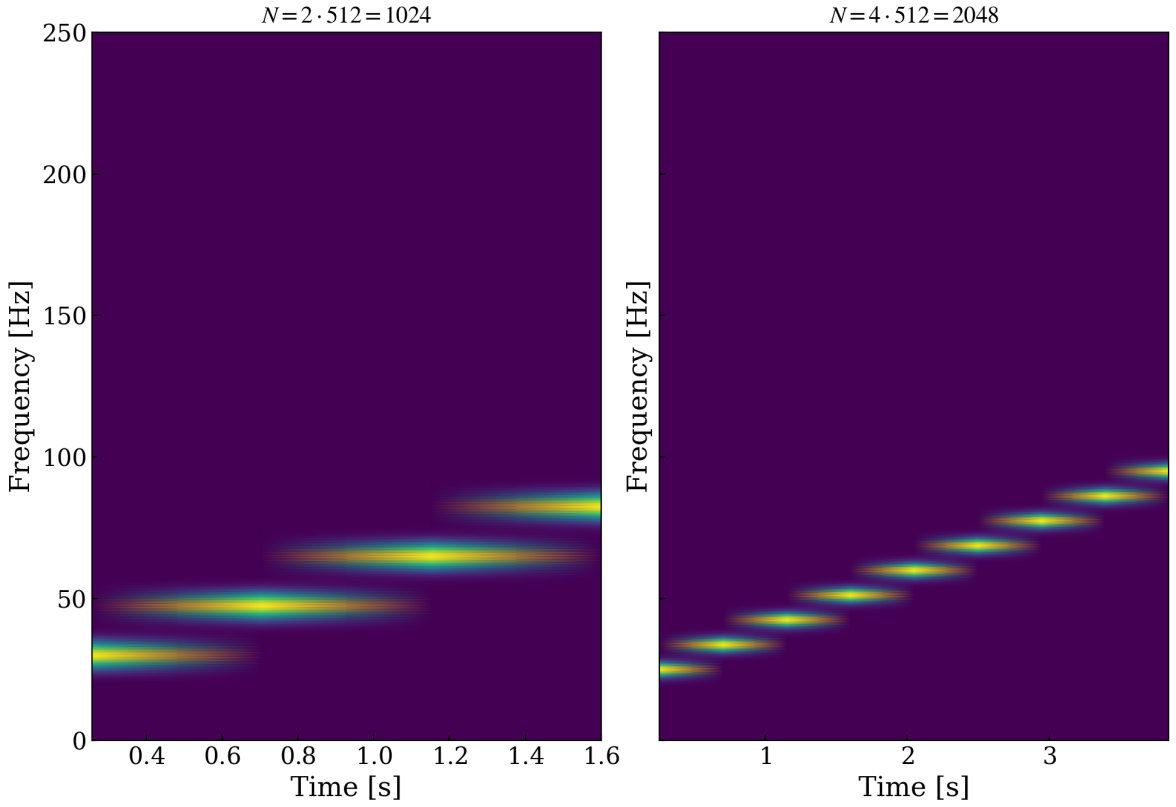
Figure 2: **Comparison of spectrograms of the chirp signal, with different nfft values.**



Listing 3: **Snippet for generating spectrogram with different number of samples.**

```
for N in [1024, 2048]:
    _, wave = get_chirp(N=N)
    freqs, times, spect = signal.spectrogram(
        wave,
        fs=fs,
        window=("hamming"),
        nperseg=nfft // 2,
        nfft=nfft,
    )
```

Figure 3: **Comparison of spectrograms of the chirp signal, with different number of samples values.**



For the lowest analyzed value, nfft = 16, we can observe a better resolution in the time domain. However, as one increases the value of nfft, the resolution in the time domain decreases in favor of the frequency domain resolution.

The above dichotomy proves the presence of the Heisenberg's uncertainty principle between time and frequency of a signal.

4 Wigner-Ville transform

The Wigner-Ville (W-V) transform is another time-frequency analysis tool used in signal processing to study the spectral content of a signal as it varies over time.

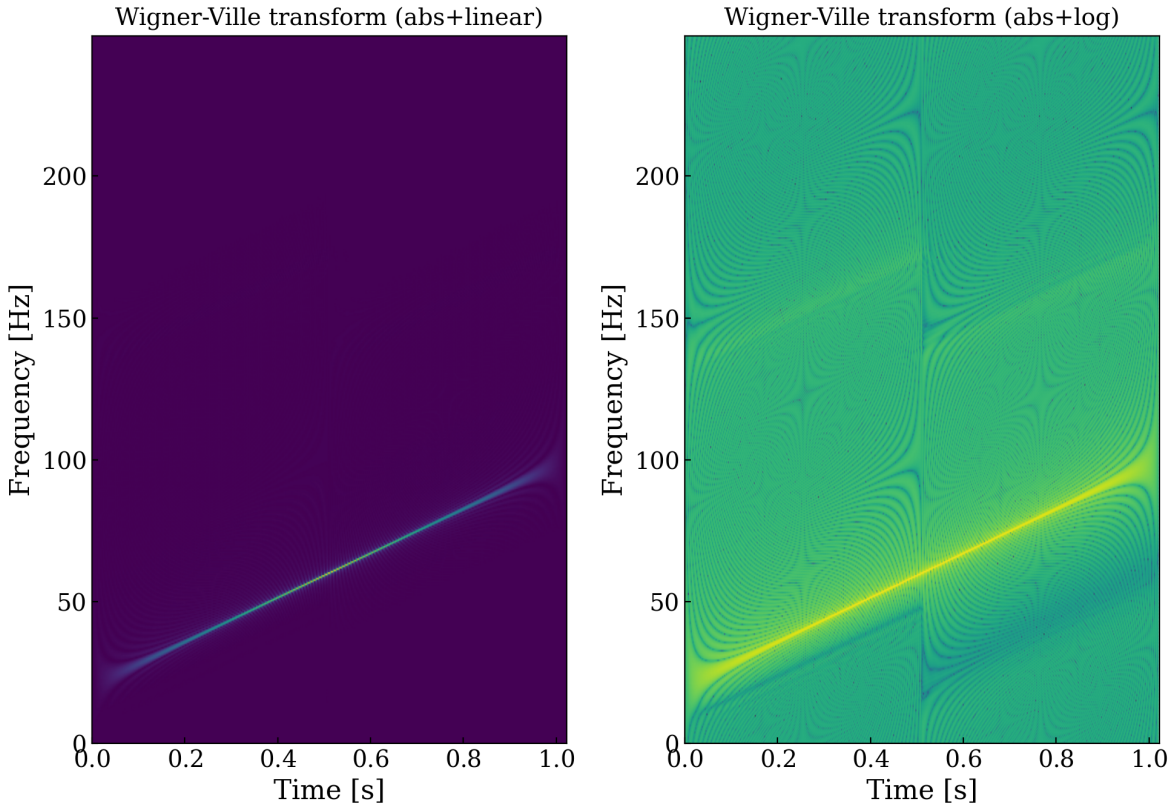
As seen in the figure 4, W-V can offers a detailed and accurate representation of the signal's time-varying spectrum. Despite this, it greatly struggles to provide accurate results at the beginning and end of the time window.

To perform the W-V transform, the function `cohen(...)`, provided with the instruction, has been used. Additionaly, to obtain an analytical function, I have used the function `hilbert` from the `scipy` module.

Listing 4: **Snippet for generating the W-V transform**.

```
spectrogram, frequencies, times = cohen(signal.hilbert(wave), fs)
```

Figure 4: **W-V of the chirp signal**.



5 W-V and STFT

For the analysis in this section, nfft = 128 was used.

The first plot, fig. 5, shows the W-V transform and the STFT of the chirp signal for frequencies from 200 to 500 [Hz]. Above the Nyquist frequency (f_{nyq} = 500/2 = 250 [Hz]), both of the transforms exhibit aliasing, where the successive frequency components start to overlap. Such a fact makes identifying the underlying frequencies, difficult or even impossible, particularly in the W-V.

Listing 5: **Snippet for generating the STFT and W-V transform for different frequencies within specified range**.

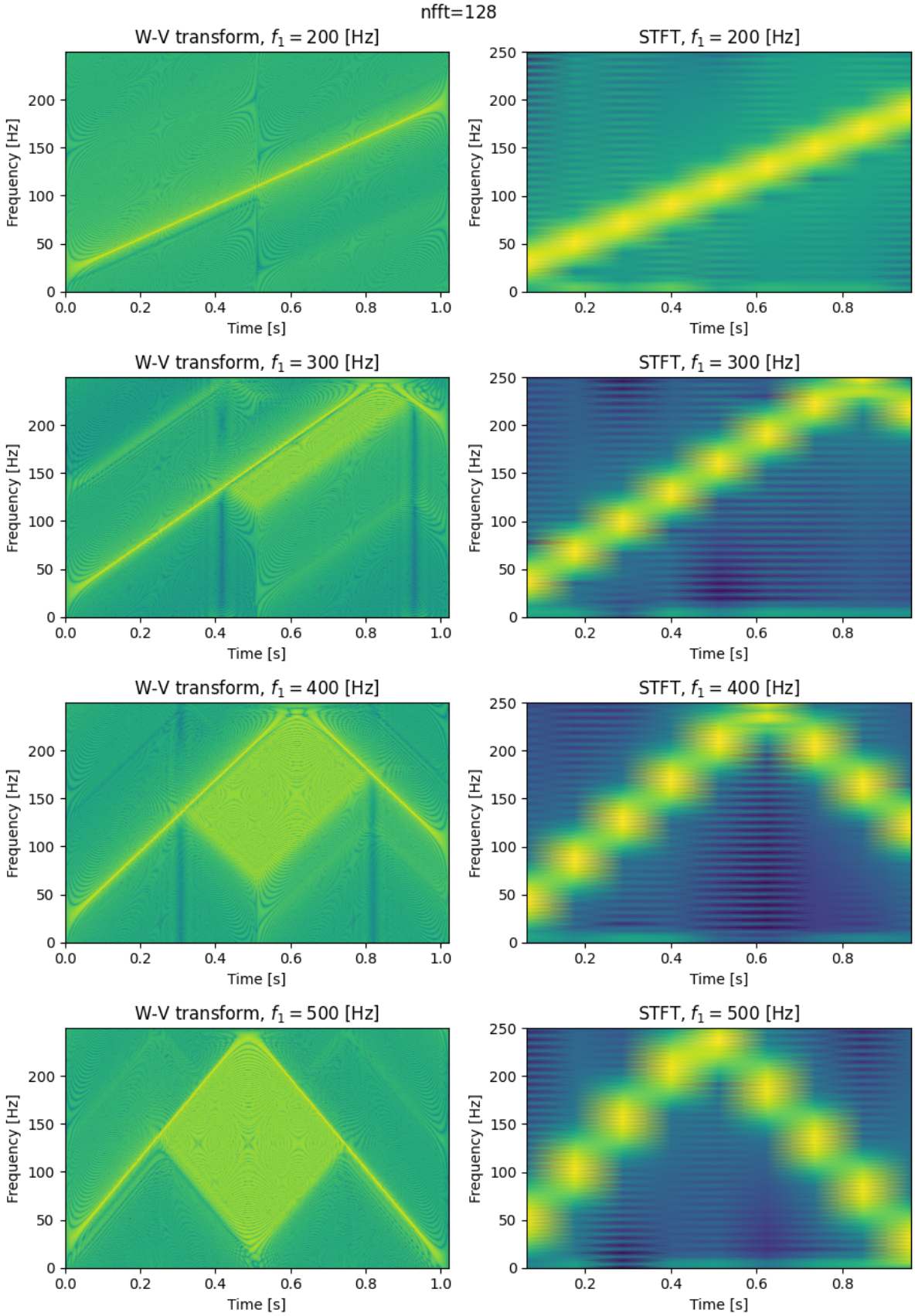
```
for f1 in [200, 300, 400, 500]:
    time, wave = get_chirp(N=512, f1=f1)

    // W-V transform
    sxx, f, t = cohen(signal.hilbert(wave), fs)

    ...

    // STFT
    f, t, sxx = signal.spectrogram(
        wave,
        fs=fs,
        window=("hamming"),
        nperseg=nfft // 2,
        nfft=nfft,
    )
```

Figure 5: **STFT and W-V transform for different frequencies within the range 200 to 500 [Hz].**



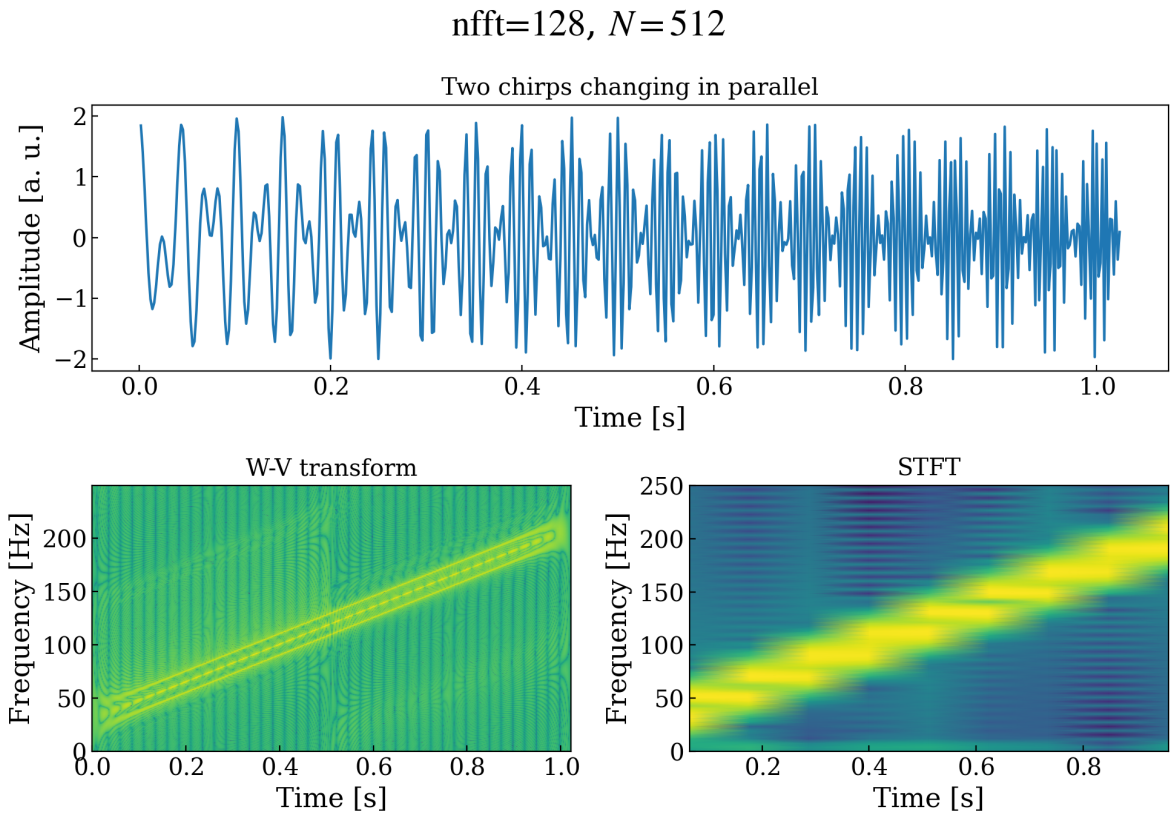
In fig. 6, both chirps' frequency changes align in parallel. Moreover, a third frequency component is observed, which is an average of the two.

Listing 6: **Snippet for calculating the spectrum of the sum of two chirps changing in parallel.**

```
N = 512
nfft = 128
f0 = 20
f1 = 200

time, chirp1 = get_chirp(N=N, f0=f0, f1=f1)
_, chirp2 = get_chirp(N=N, f0=2 * f0, f1=f0 + f1)
wave = chirp1 + chirp2
```

Figure 6: **W-V transform and STFT of the sum of two chirps changing in parallel.**



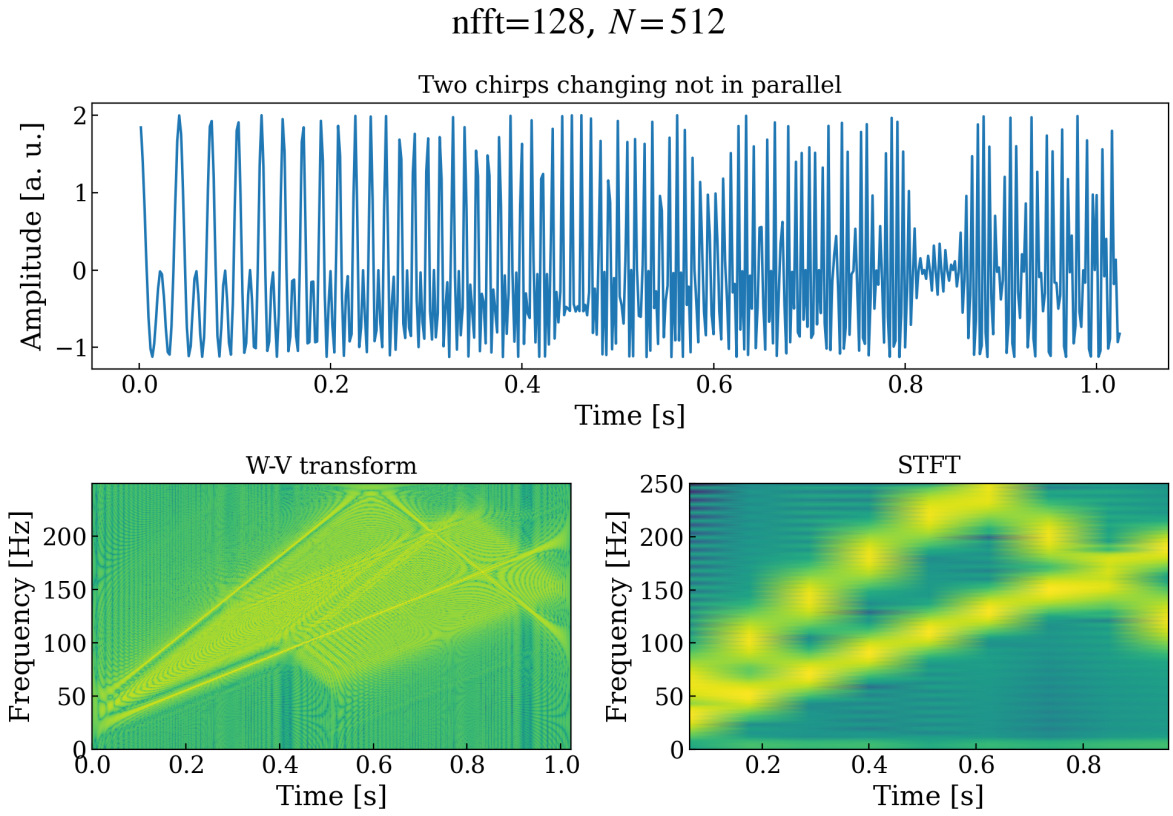
As expected, in fig. 7, chirps' frequencies do not change in parallel, which manifests itself as multiple intersecting lines.

Listing 7: **Snippet for calculating the spectrum of the sum of two chirps changing in parallel.**

```
N = 512
nfft = 128
f0 = 20
f1 = 200

time, chirp1 = get_chirp(N=N, f0=f0, f1=f1)
_, chirp2 = get_chirp(N=N, f0=2 * f0, f1=2 * f1)
wave = chirp1 + chirp2
```

Figure 7: **W-V transform and STFT of the sum of two chirps changing not in parallel.**



6 Conclusion

In conclusion, the report explores time-frequency analysis techniques, focusing on Short-time Fourier Transform (STFT) and Wigner-Ville transform. Analysis of chirp signals illustrates the manifestation of Heisenberg's uncertainty principle. Despite limitations, both transforms provide valuable insights into signal processing, aiding in understanding time-varying spectral content.

The entire code for generating the data and plots can be found at:

https://github.com/davkk/signal-analysis/tree/main/sat/lab03

References

[1] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.chirp.html