This script requires the installation of tftb, which can be downloaded from https://tftb.nongnu.org/

Download and unzip the files to your computer, then add the path with the following command

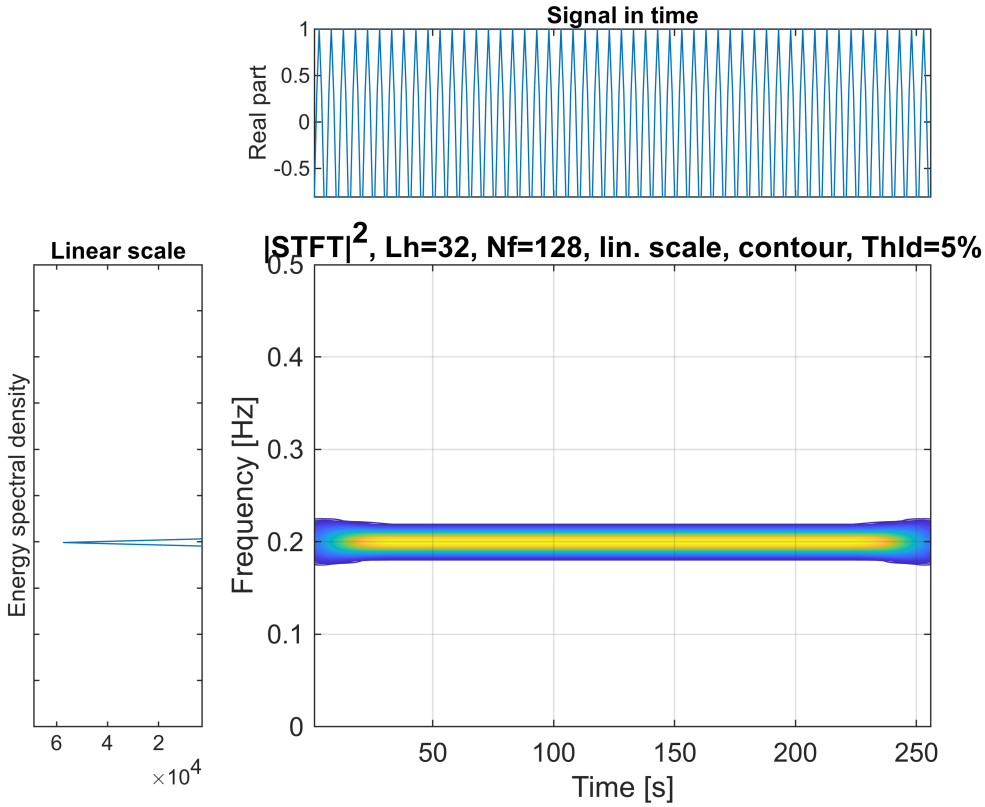
```
addpath("C:\Users\zouyi\OneDrive\文档\MATLAB\tftb-0.2\mfiles")
% addpath("[your_path]\tftb-0.2\mfiles")
```

Then. let's start.

Here, x1 is a pure sine wave of a constant frequency 0.2. Let's visualize the STFT

```
x1 = fmconst(256, 0.2);
tfrstft(x1); grid on
```

```
Options saved
```

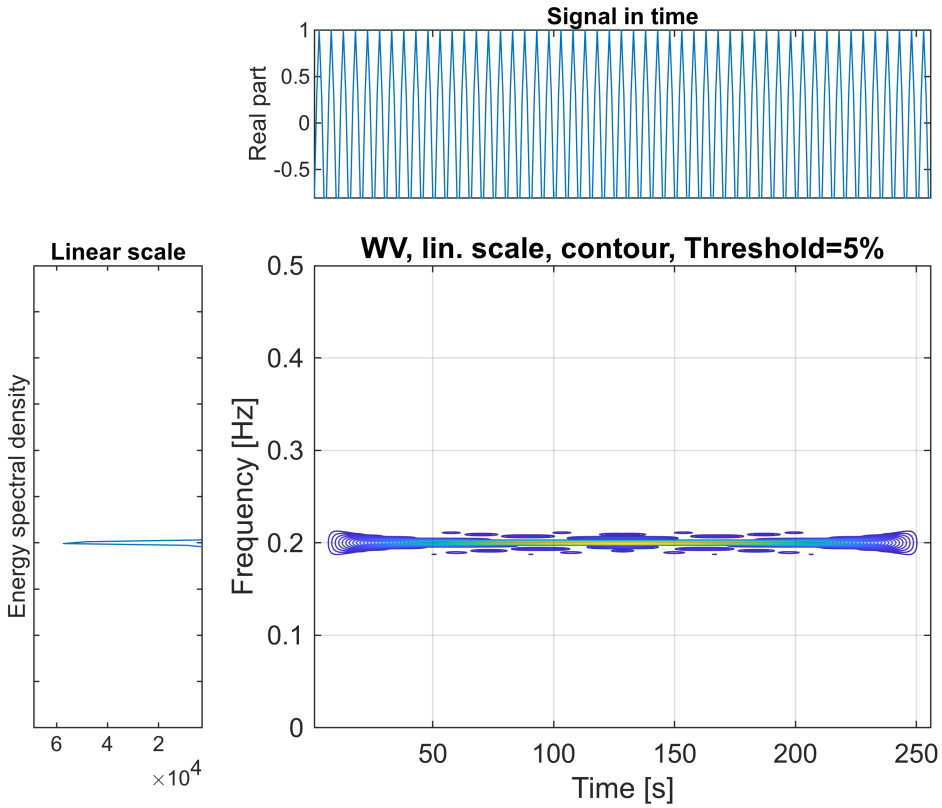



We get a fairly good frequency resolution around 0.2.

*If this is your first time using tftb, you get a default display window that only contains the WVD figure. To show the signal and the spectrum, go to 'change display layout' and turn on the corresponding plots. Then save the settings.

As a comparison, let's perform WVD on the same sine wave.

```
tfrwv(x1); grid on
```



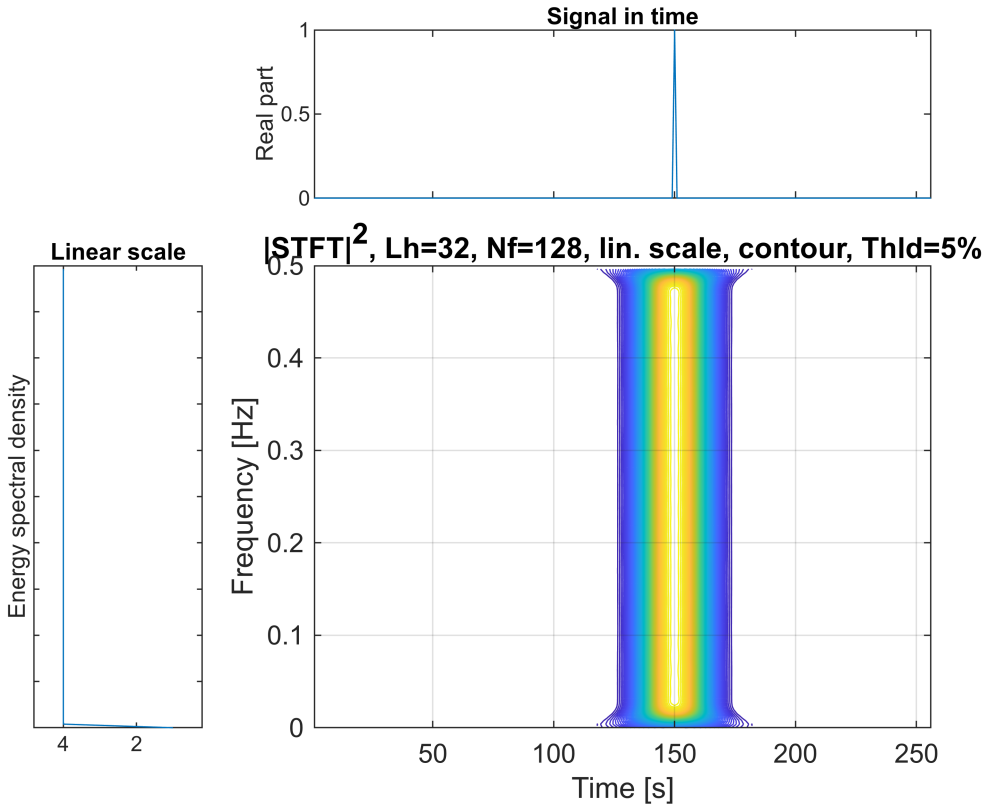

We can see that the WVD gives a much more precise result. **WVD has a much higher resolution in frequency**!

Let's test the **time resolution**.

x2 is a zero signal with a sudden pulse at t=150. Here, I use the analytical representation of the signal i.e. hilbert(x2)

First, we perform the STFT

```
x2 = zeros(256,1); x2(150) = 1;
tfrstft(hilbert(x2)); grid on
```

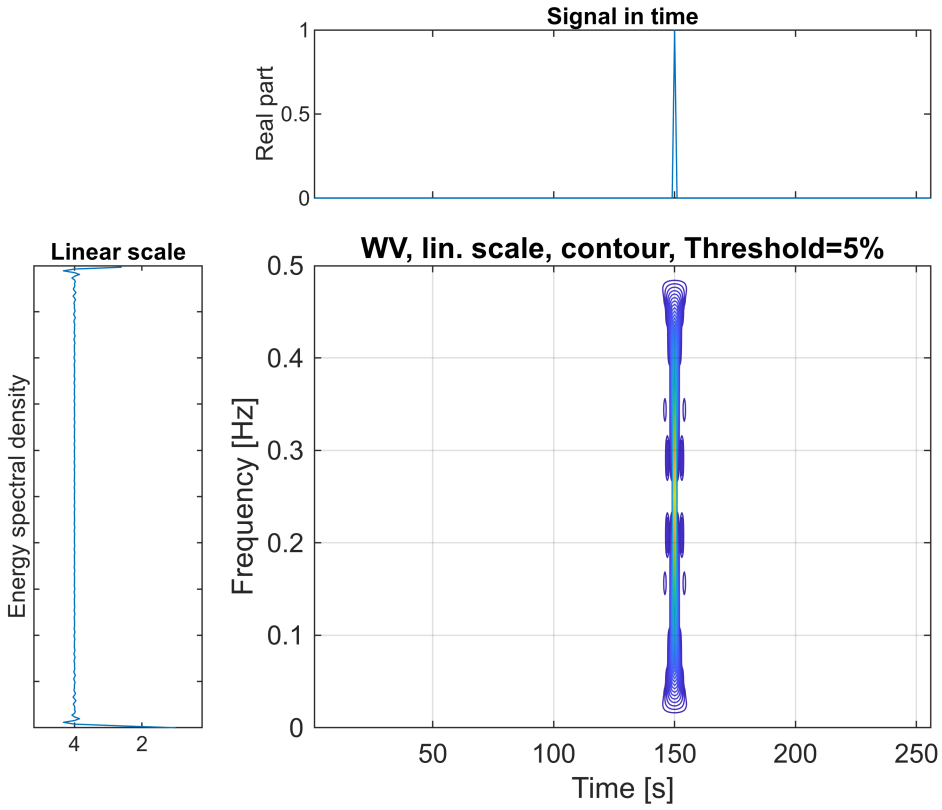



We get a wide block around t=150. The time resolution is not good.

Also, notice that the energy spectrum has a high value for all frequencies. This is related to the uncertainty principle. In short, because we are observing a very short duration (only one time instant), is it impossible to known the exact frequency. This pulse can belong of the signal of any frequency, so we have a high uncertainty in frequency.

Then, let's see the WVD of this pulse

```
tfrwv(hilbert(x2)); grid on
```

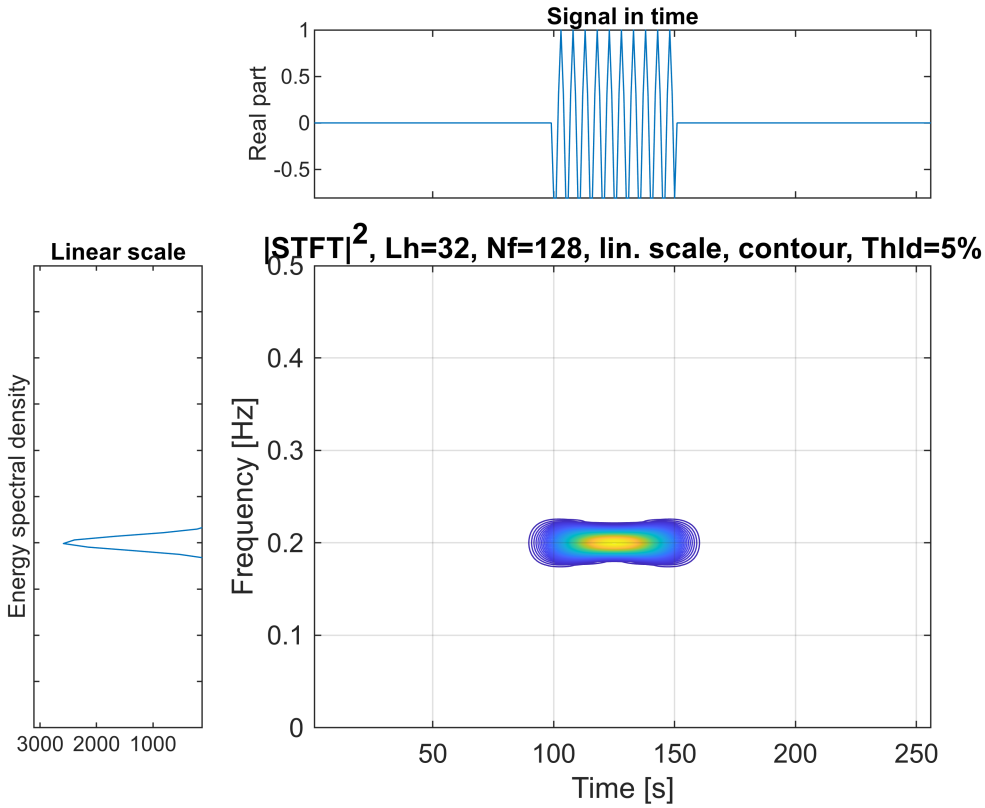



See how narrow the WVD output is!

Let's try a signal with a contant frequency but limited to a time interval

x11 is a the sine wave of frequence 0.2 over t=[100,150], and 0 otherwise. Here is the STFT of x11:

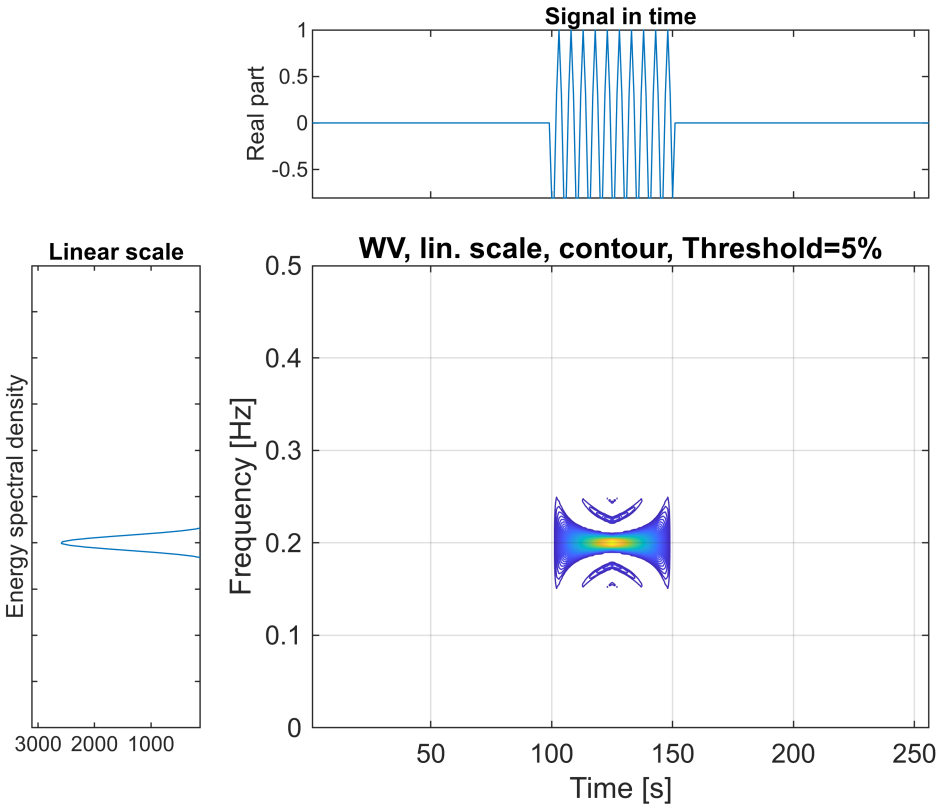
```
x11 = zeros(256,1); x11(100:150) = x1(100:150);
tfrstft(x11); grid on
```




The STFT output has leakage outside the time interval [100,150]. So we cannot determine the time limitation based on the STFT output. Also, the frequency resolution is poor.

Here is the WVD of x11

```
tfrwv(x11); grid on
```




WVD perfectly preserves the time-limitation in the signal! The WVD is zero outsie the interval [100, 150]. It also achieves a higher resolution in frequency compared to STFT.

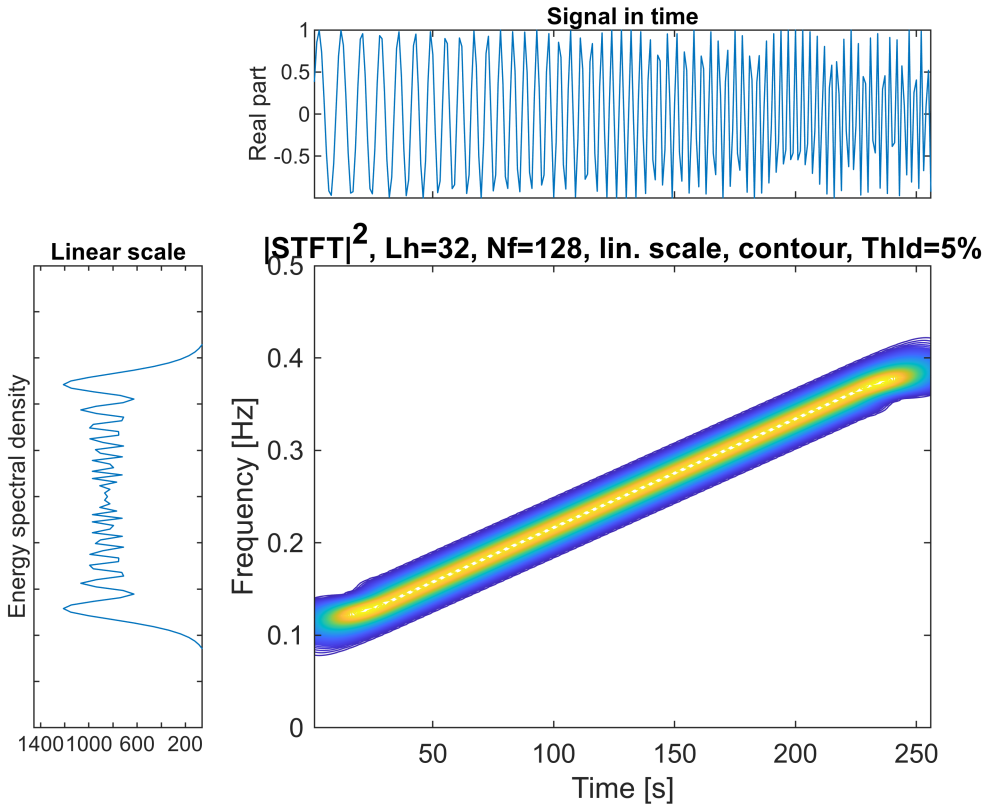
The problem with WVD is the unwanted noise around t=100 and t=150, arised from the Fourier transform of the discountinuity at the time limitations.

What about **non-stationary signals**? For instant, a chrip. Recall that a chirp is a signal in which the frequency increases or decreases with time

Here, x3 is a chirp whose frequency linearly increases from 0.1 to 0.4 with time.

First we perform the STFT of x3:

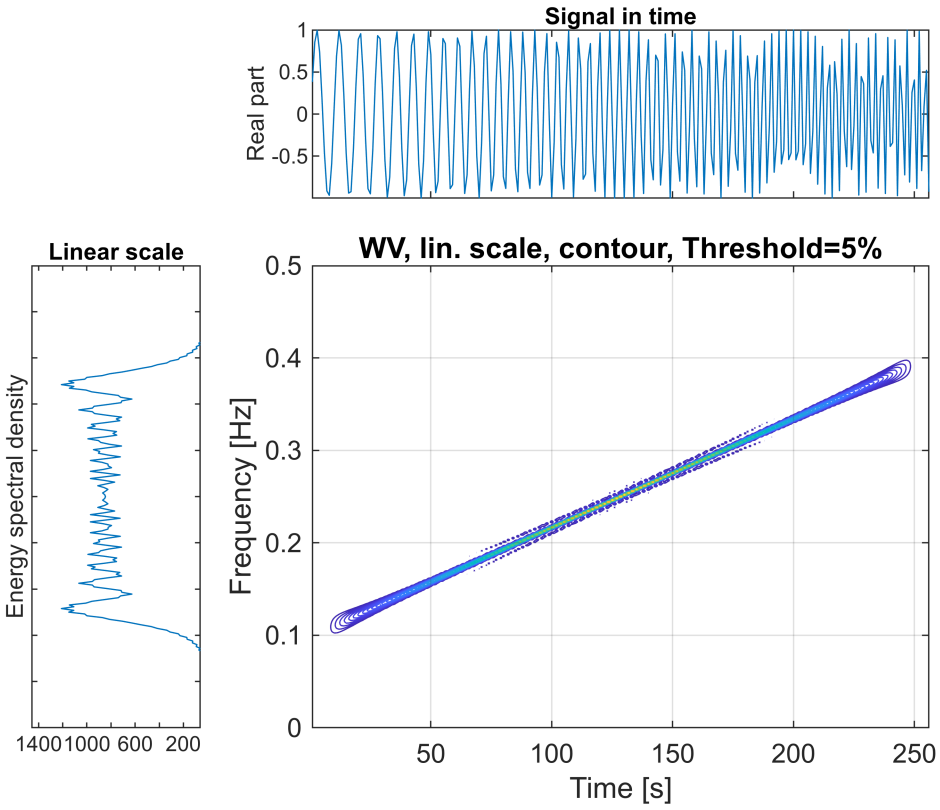
```
x3 = fmlin(256, 0.1, 0.4);
tfrstft(x3);
```




STFT reveals the linearly increasing frequency in the chirp, but the resolution is quite low.

Now, we perform WVD on the chirp:

```
tfrwv(x3); grid on
```



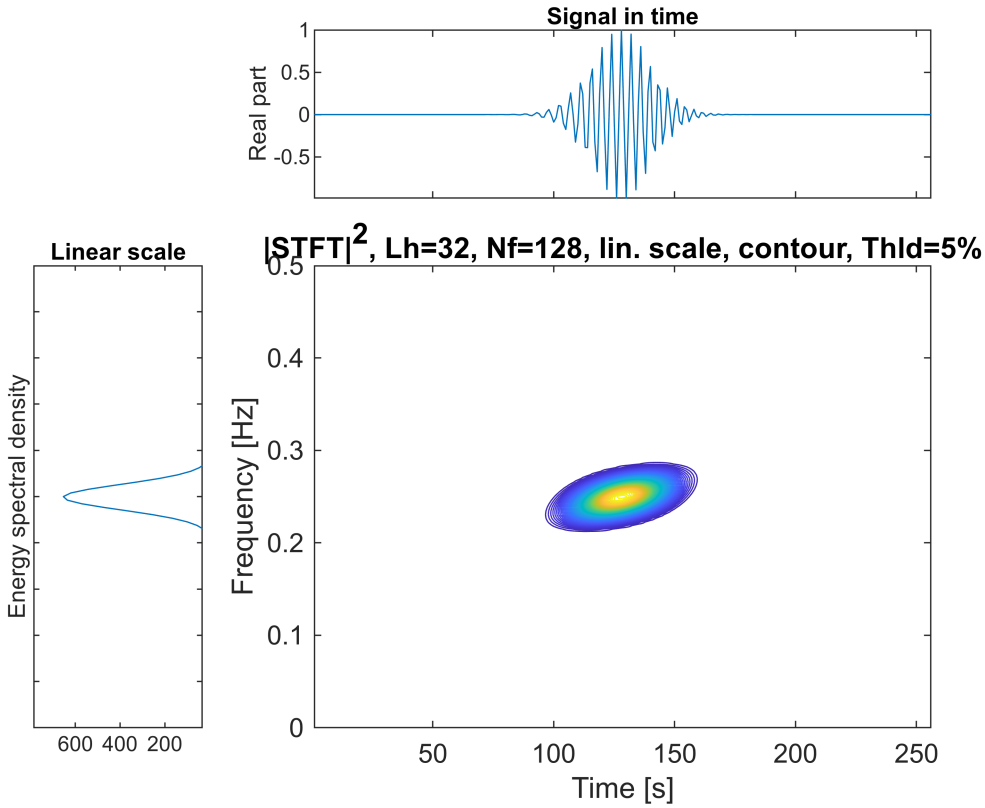

Comparing the WVD of the chirp to the STFT results, we can observe a significant improvement in time-frequency resolution.

Hwo about a **modulated chirp**?

x31 is modulated version of x3, by using a Gaussian envelop.

Here is the STFT of x31

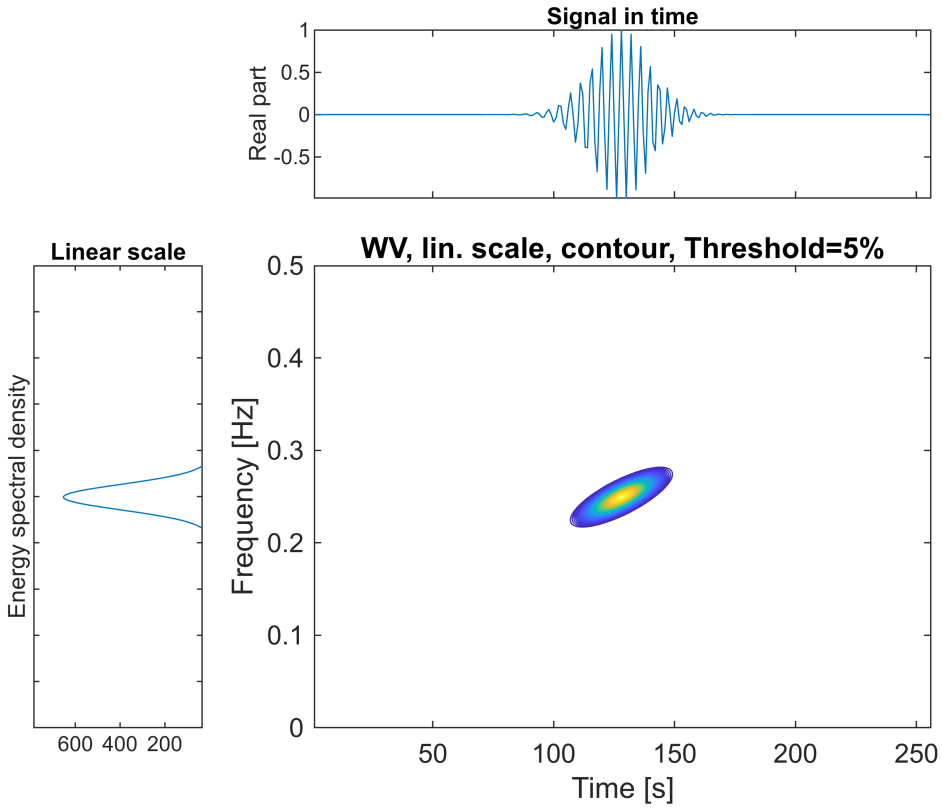
```
x31 = amgauss(256).*x3;
tfrstft(x31);
```




We have a disc that shows the linearly increasing frequency in the chirp. However, the disc has a low resolution in both time and frequency

Here is the WVD of the Gaussian chirp

```
tfrwv(x31);
```



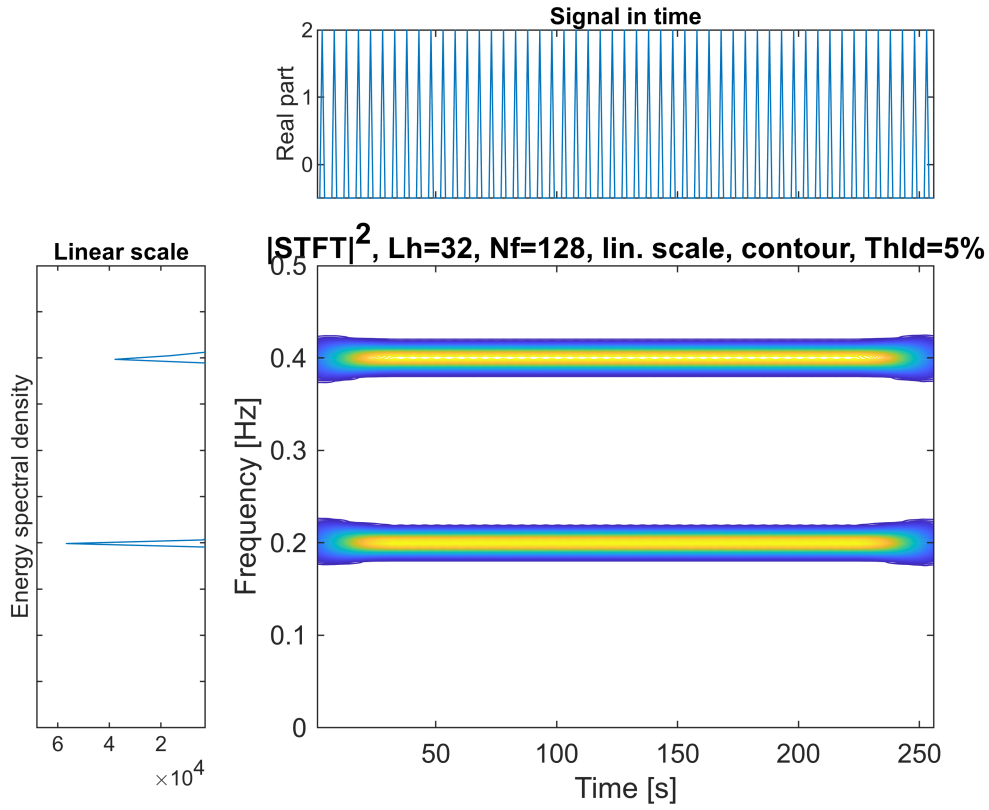

The disc of WVD shows a more precise linear relationship between time and frequency, which is what we want.

So far, we have looked at some simple waves, where WVD beats STFT on all of them. We now turn to some more complicated waves, such as **composite waves**.

x4 is a composite signal that contains two sine waves of different frequencies 0.2 and 0.4.

As usual, we perform the STFT on x4 first:

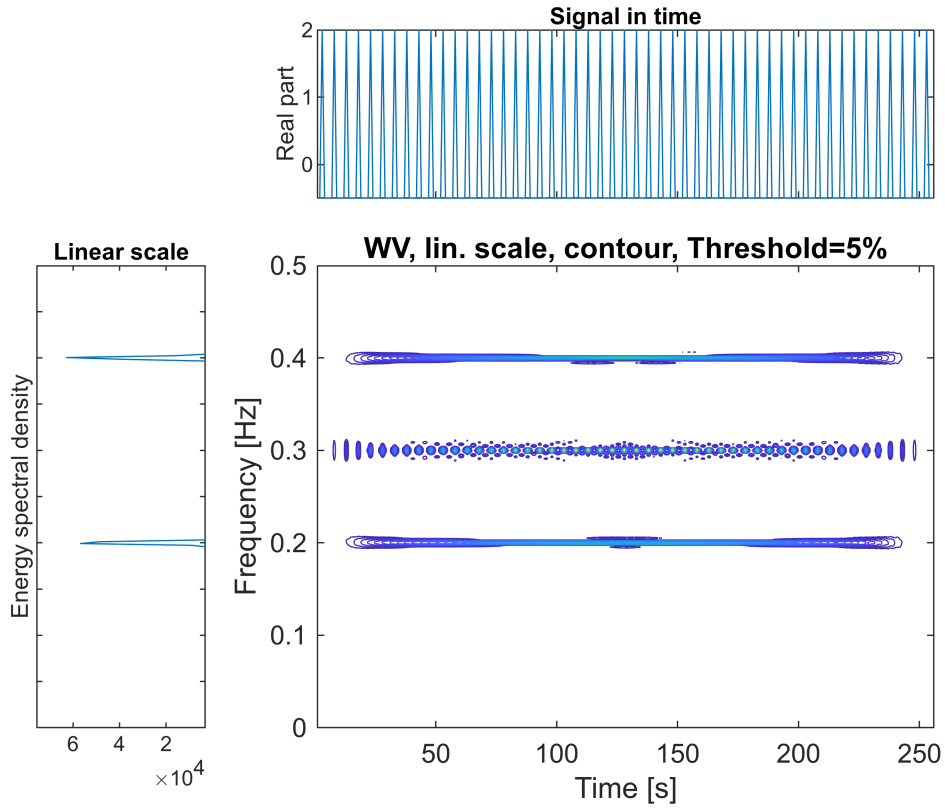
```
x4_1 = fmconst(256, 0.2);
x4_2 = fmconst(256, 0.4);
x4 = x4_1 + x4_2;
tfrstft(x4);
```




In the plot, we can clearly get two distinct lines around frequencies 0.2 and 0.4. The frequency resolution is not that good though.

Here is the WVD of the composite signal:

```
tfrwv(x4);
```




We see some interesting outputs of WVD. As in STFT, there are two lines around frequencies 0.2 and 0.4. The lines are very concentrated, indicating a higher frequency resolution than STFT.

However, there are a lot of extra terms:

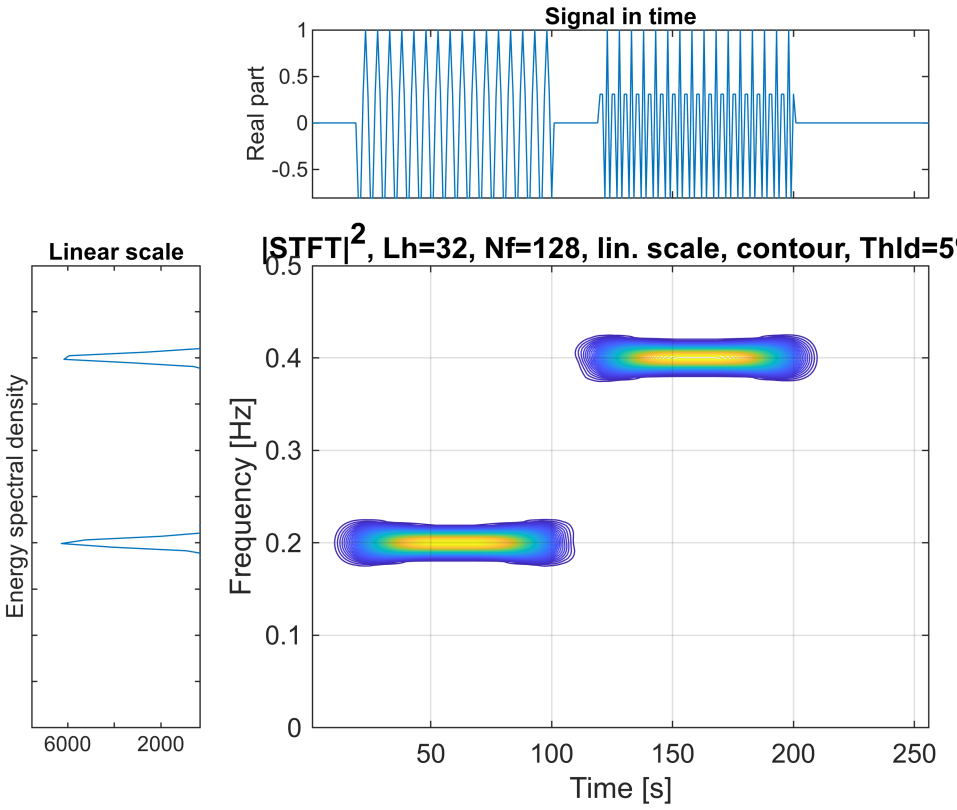
1. The cross terms are visible around frequency 0.3. The magnitue of the cross terms are quite large.

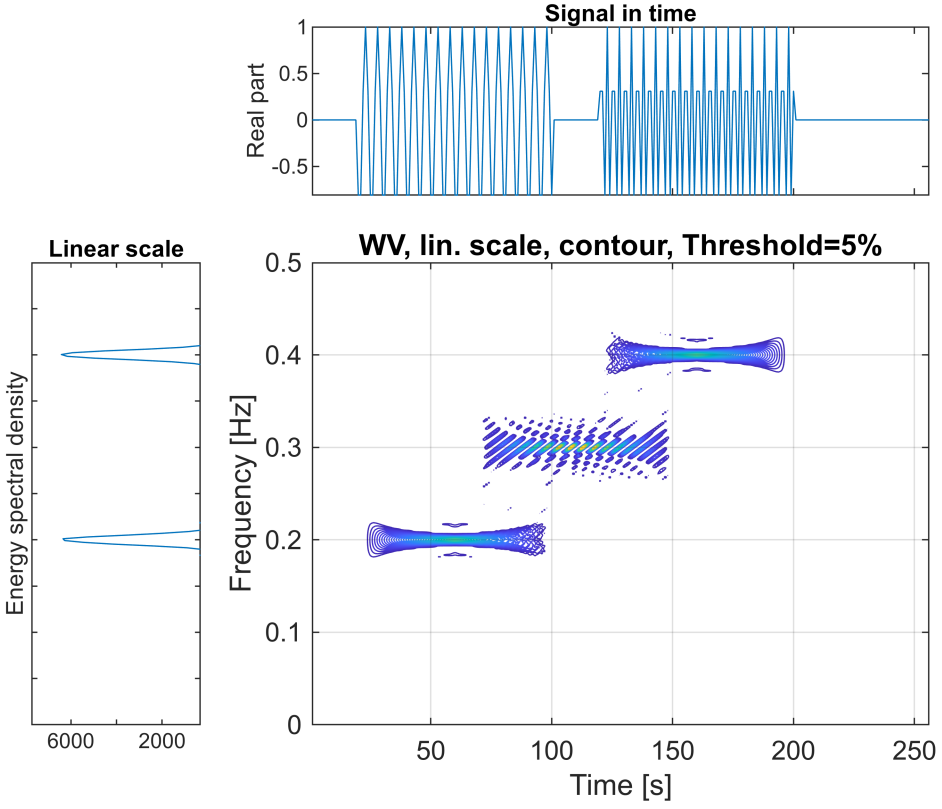
The cross terms look like strong interference, but they are not a complete disaster. Notice that the cross terms appears exactly at the average value of two composite frequencies. So we know exactly where the cross terms will show up. Even if you are only looking at the WVD figure, you can tell that the composite frequencies are around 0.2 and 0.4.

To visualize the cross-terms on some more complicate signals:

x5 is the composite of two pure sine waves at different frequencies and different time intervals

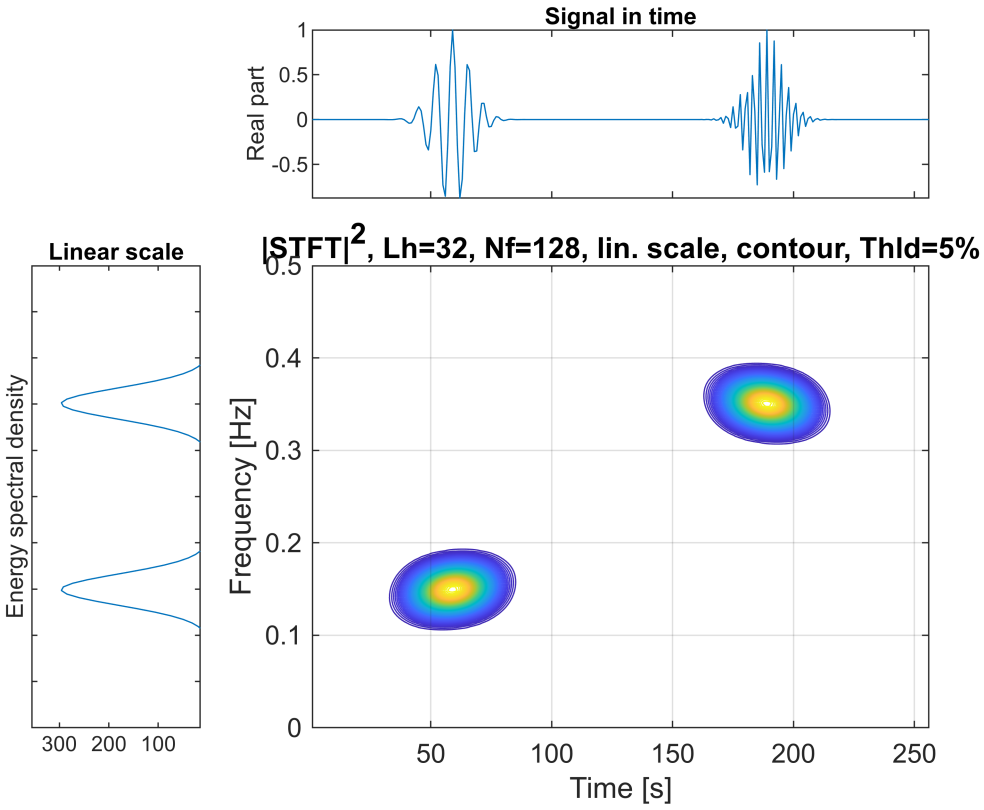
```
x5 = zeros(256,1); x5(20:100) = x4_1(20:100); x5(120:200) = x4_2(120:200);
tfrstft(x5); grid on
tfrwv(x5); grid on
```

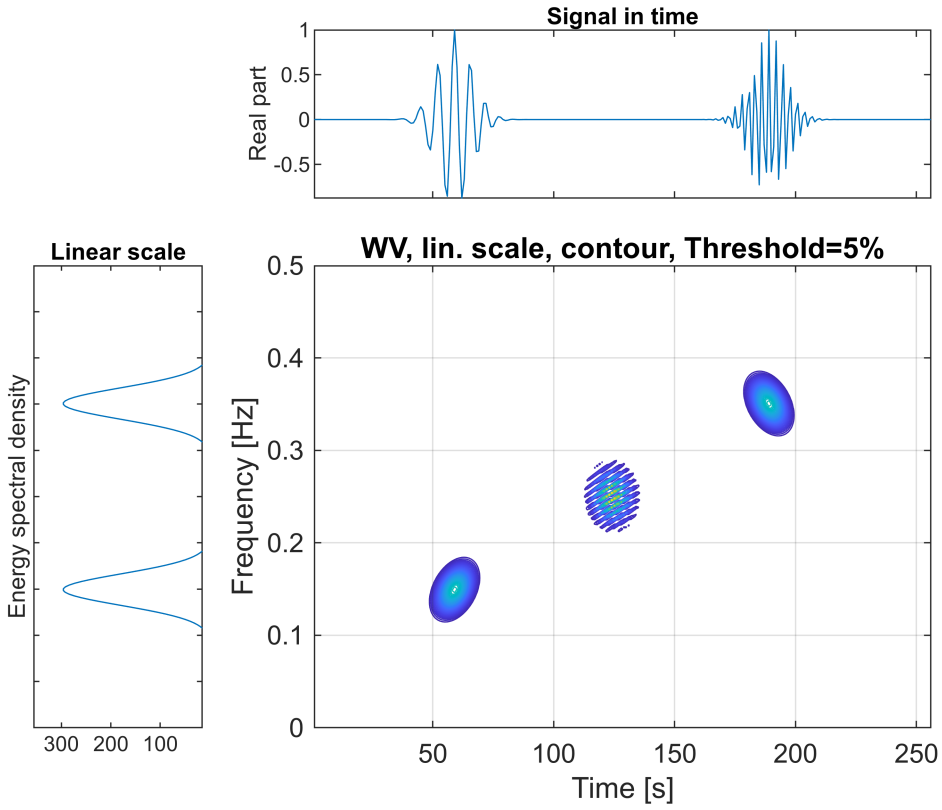

Signal in time
Real part
1
0.5
0
-0.5
Linear scale
|STFT|2, Lh=32, Nf=128, lin. scale, contour, ThId=5%
Energy spectral density
6000
2000
Frequency [Hz]
0.5
0.4
0.3
0.2
0.1
0
50
100
150
200
250
Time [s]


Signal in time
Real part
1
0.5
0
-0.5
Linear scale
WV, lin. scale, contour, Threshold=5%
Energy spectral density
6000
2000
Frequency [Hz]
0.5
0.4
0.3
0.2
0.1
0
50
100
150
200
250
Time [s]

x6 is a composite signal that consists of two Gaussian chirplets. One chirplet has increasing frequency over time, the other chirplet has decreasing frequency over time.

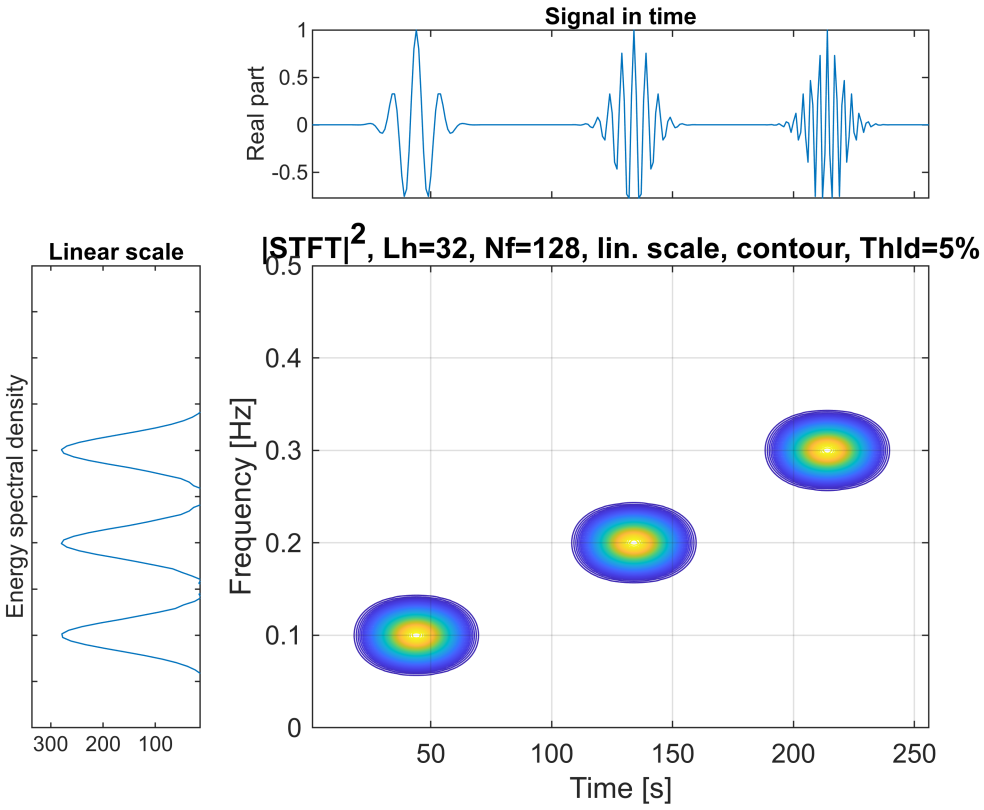
```
x6_1 = fmlin(80, 0.1, 0.2);
x6_2 = fmlin(80, 0.4, 0.3);
x6 = zeros(256,1); x6(20:99) = amgauss(80).*x6_1; x6(150:229) = amgauss(80).*x6_2;
tfrstft(x6); grid on
tfrwv(x6); grid on
```

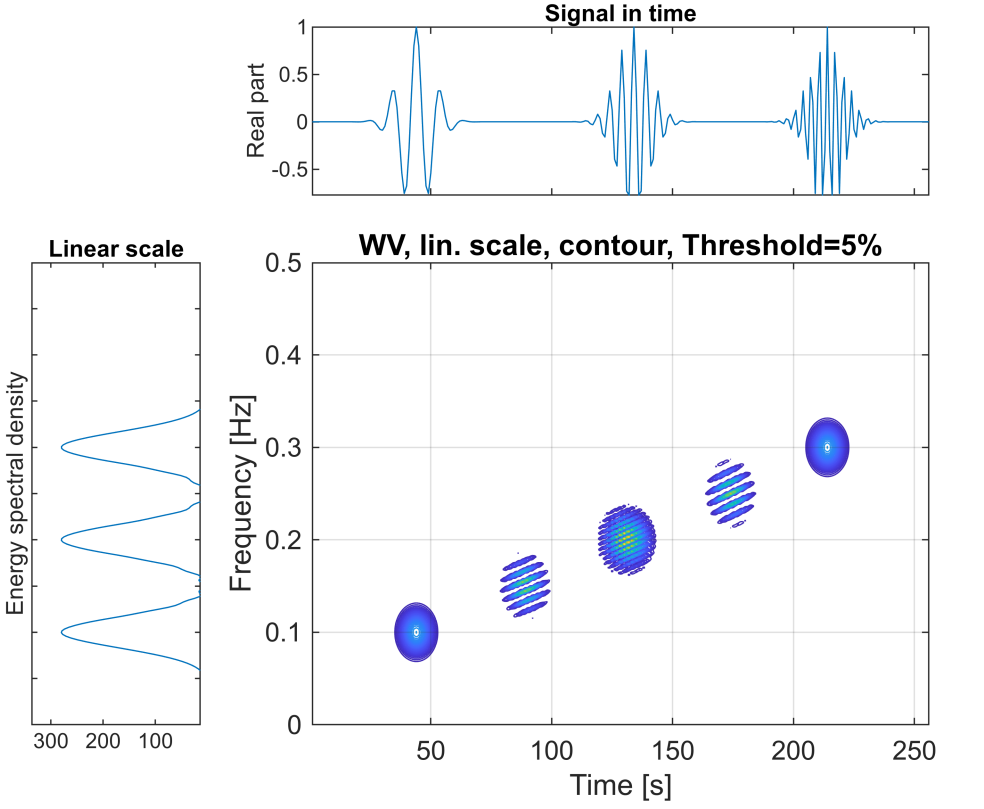





You can barely see the linearly increasing/decreasing TF relationship from STFT

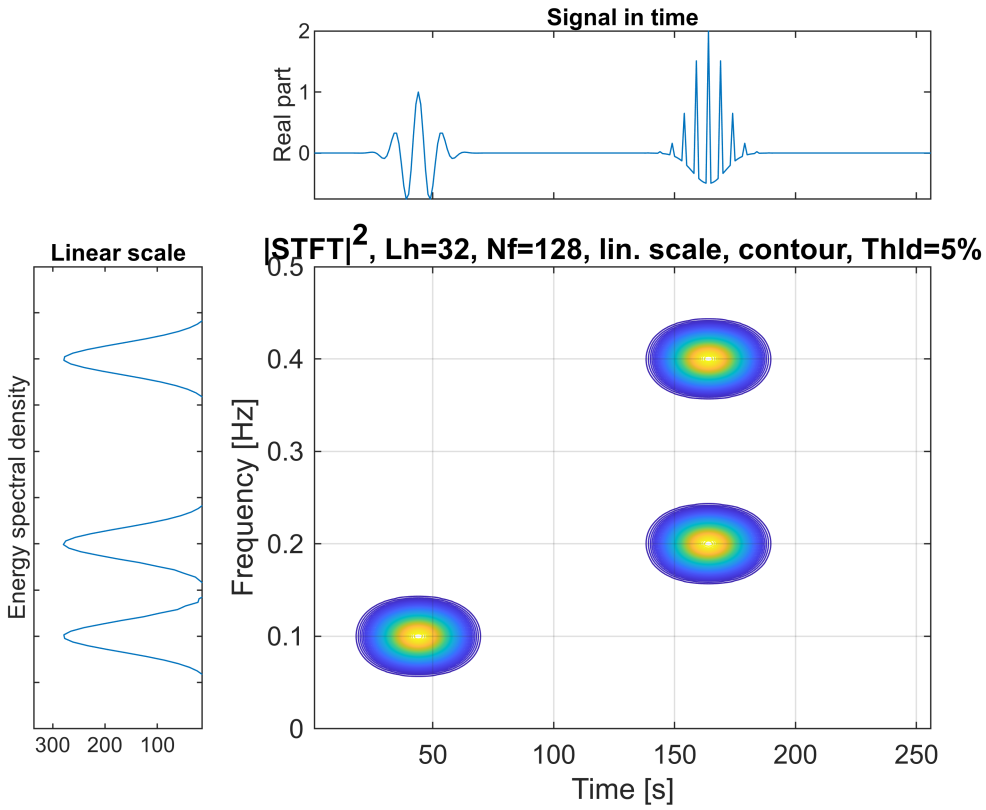
```
x7_1 = amgauss(70).*fmconst(70, 0.1);
x7_2 = amgauss(70).*fmconst(70, 0.2);
x7_3 = amgauss(70).*fmconst(70, 0.3);
x7 = zeros(256,1); x7(10:79) = x7_1; x7(100:169) = x7_2; x7(180:249) = x7_3;
tfrstft(x7); grid on
tfrwv(x7); grid on
```

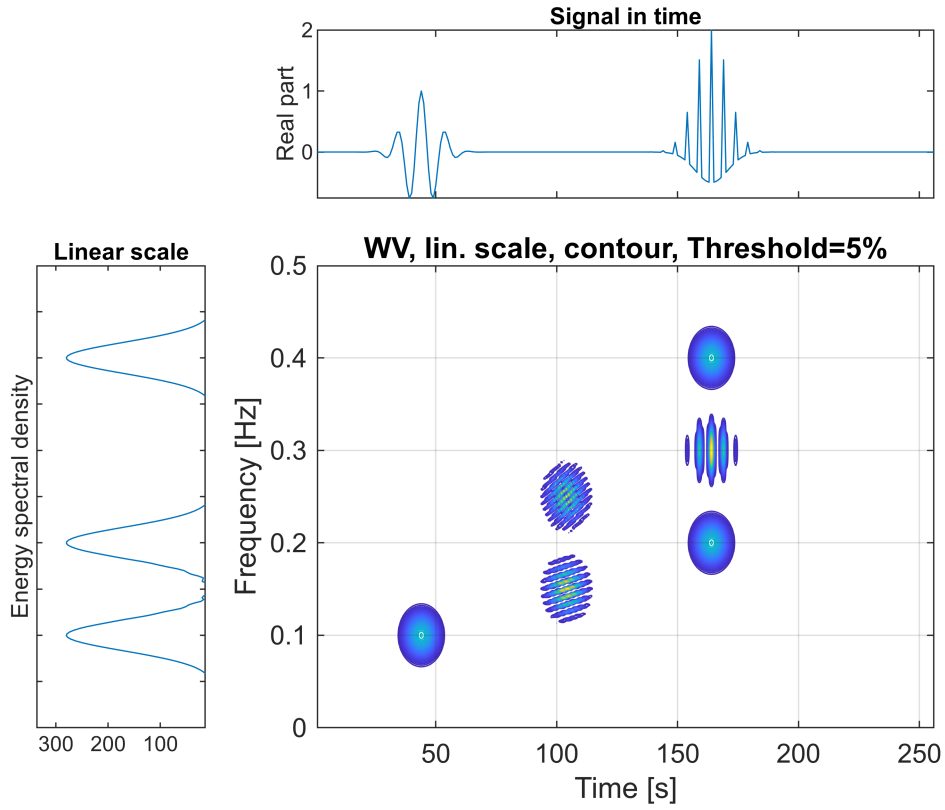

Signal in time
Real part
1
0.5
0
-0.5
Linear scale
|STFT|2, Lh=32, Nf=128, lin. scale, contour, Thld=5%
Energy spectral density
300
200
100
Frequency [Hz]
0.5
0.4
0.3
0.2
0.1
0
50
100
150
200
250
Time [s]


Signal in time
Real part
1
0.5
0
-0.5
Linear scale
WV, lin. scale, contour, Threshold=5%
Energy spectral density
300
200
100
Frequency [Hz]
0.5
0.4
0.3
0.2
0.1
0
50
100
150
200
250
Time [s]

This is an interesting one. We know that if the signal is a composition of three sub-signals, we will get three cross terms. But in the figure, we can only see two. This is because one of the cross terms coincides with the auto-term at frequency 0.2

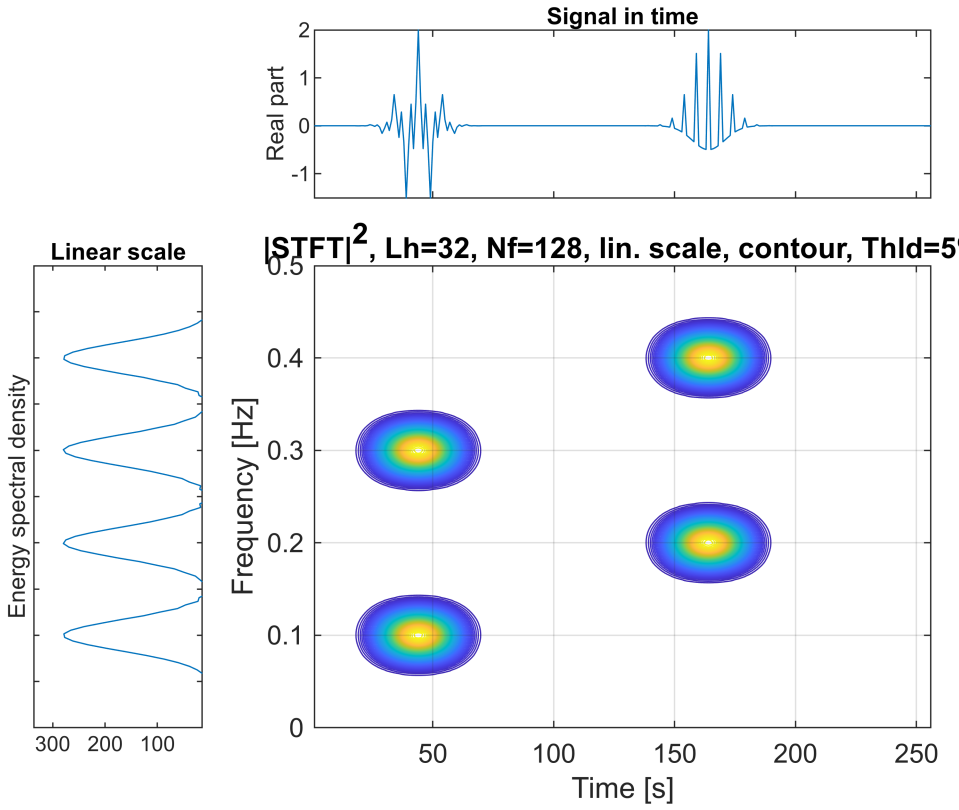
```
x8_1 = amgauss(70).*fmconst(70, 0.1);
x8_2 = amgauss(70).*fmconst(70, 0.2);
x8_3 = amgauss(70).*fmconst(70, 0.4);
x8 = zeros(256,1); x8(10:79) = x8_1; x8(130:199) = x8_2 + x8_3;
tfrstft(x8); grid on
tfrwv(x8); grid on
```

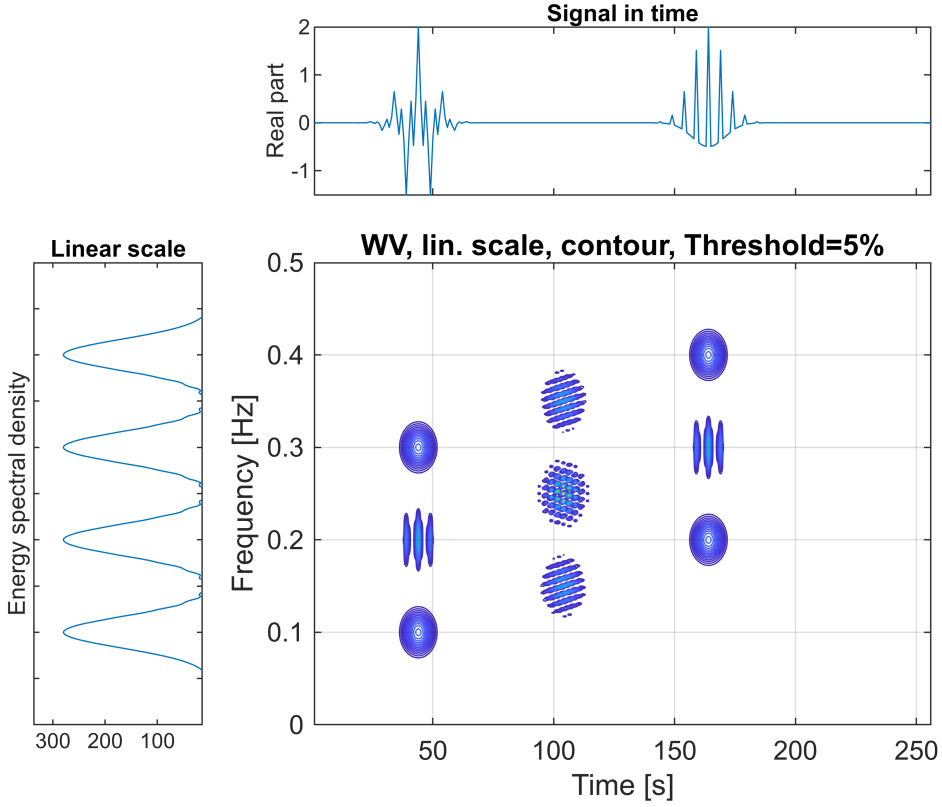





```
x9_1 = amgauss(70).*fmconst(70, 0.1);
x9_2 = amgauss(70).*fmconst(70, 0.2);
x9_3 = amgauss(70).*fmconst(70, 0.4);
x9_4 = amgauss(70).*fmconst(70, 0.3);

x9 = zeros(256,1); x9(10:79) = x9_1+x9_4; x9(130:199) = x9_2 + x9_3;
tfrstft(x9); grid on
tfrwv(x9); grid on
```



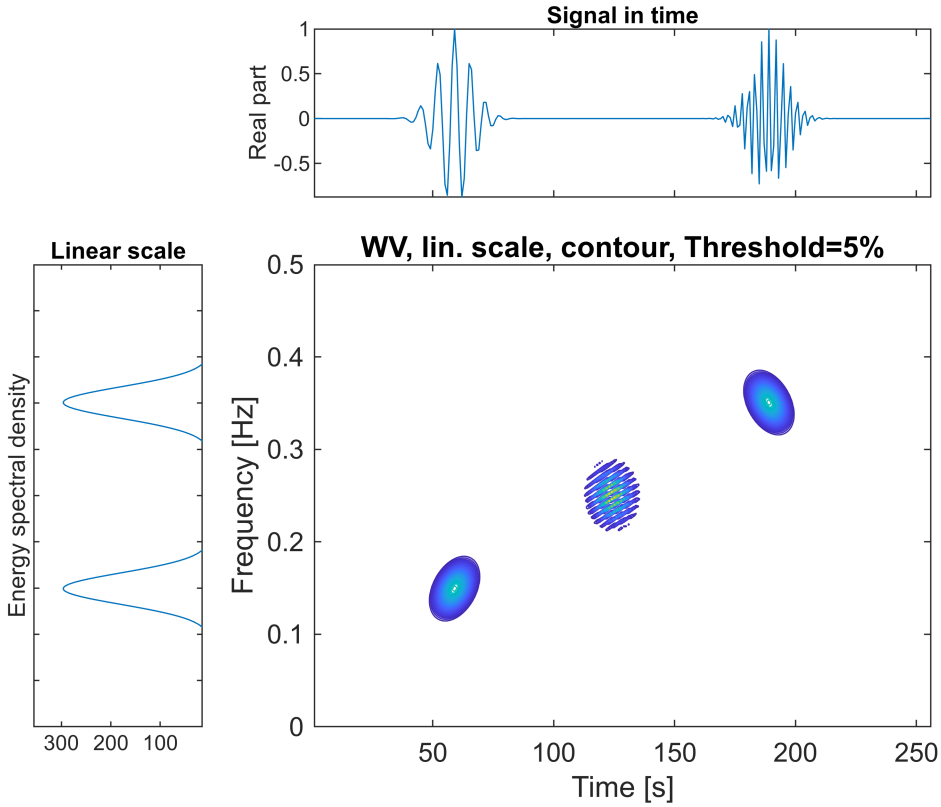




In the WVD plot, how many cross-terms are there? 5?

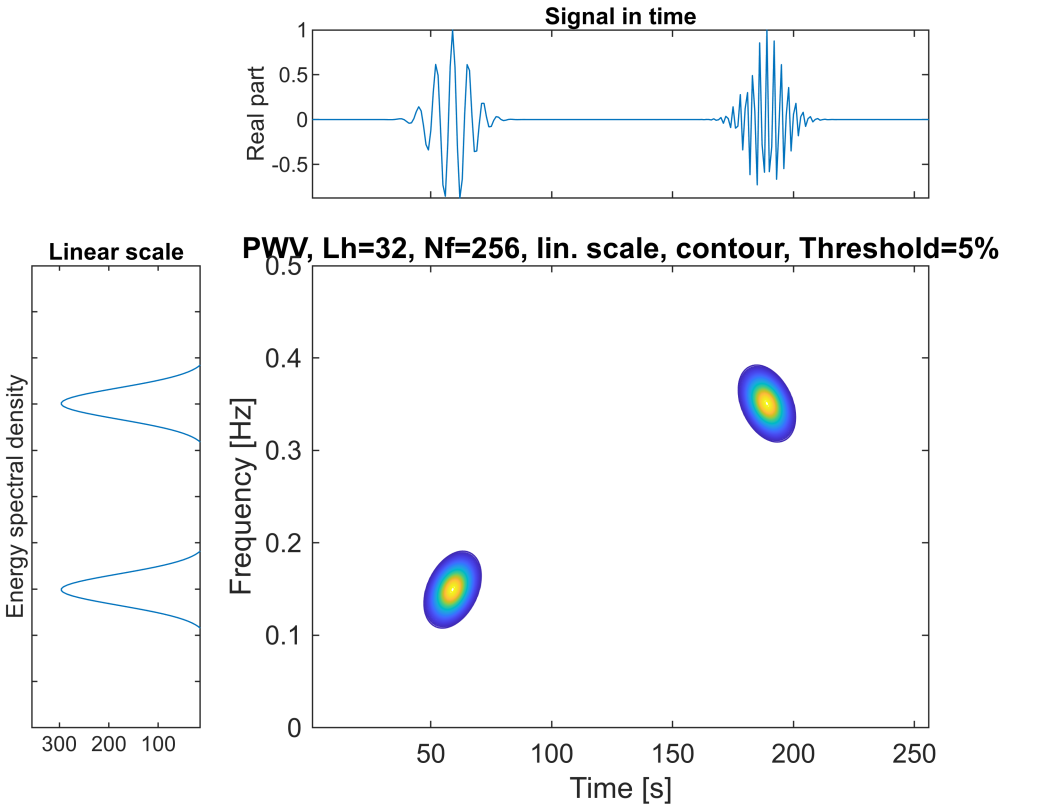
The cross terms can be suppressed by filtering and smoothing, leading to pseudo WVD (PWVD) and smoothed PWVD (SPWVD):

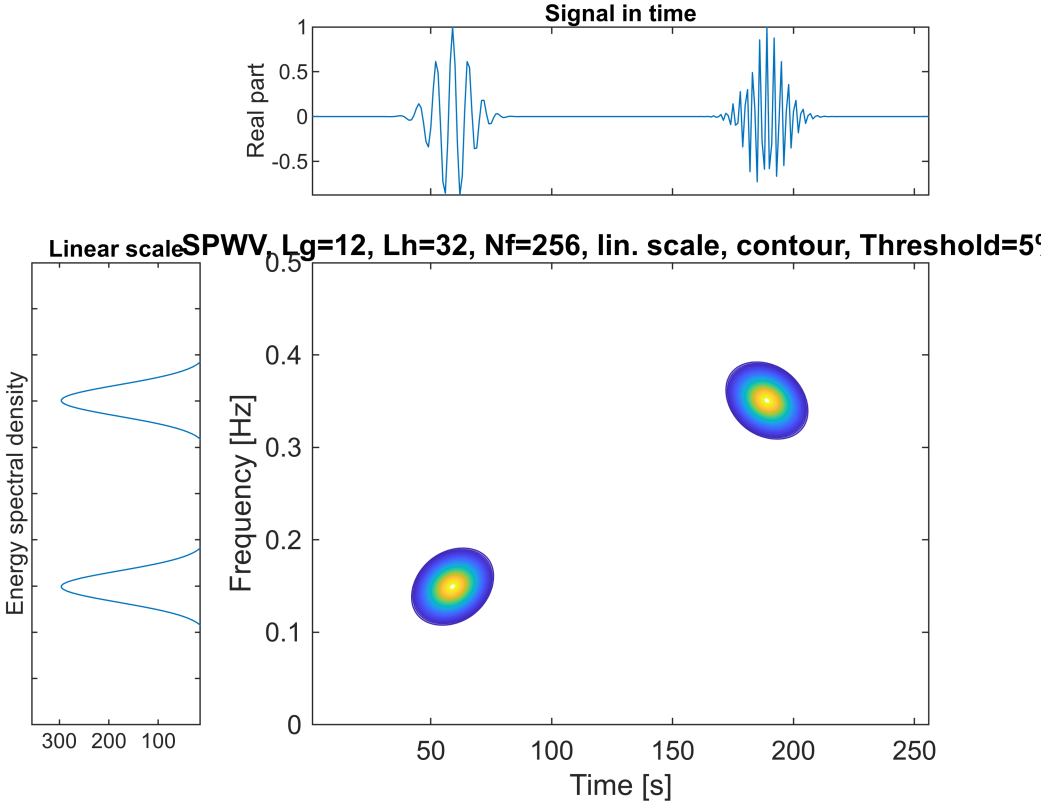
Suppose we perform WVD, PWVD, and SPWVD on x6

```
tfrwv(x6);
tfrpwv(x6);
```




```
tfrspwv(x6);
```

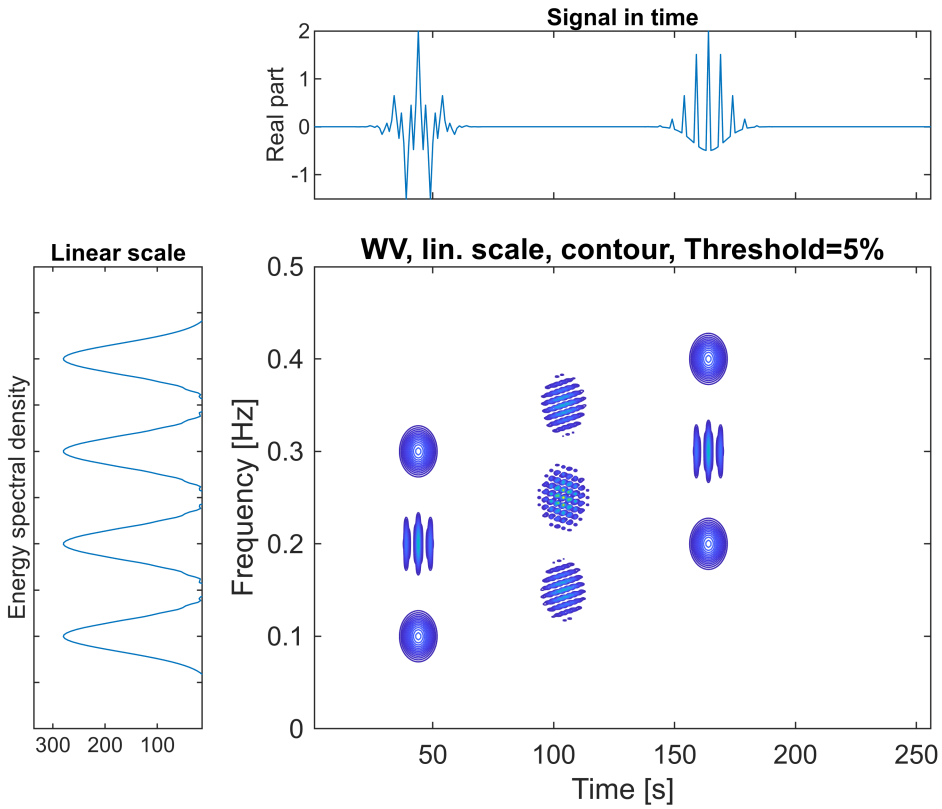


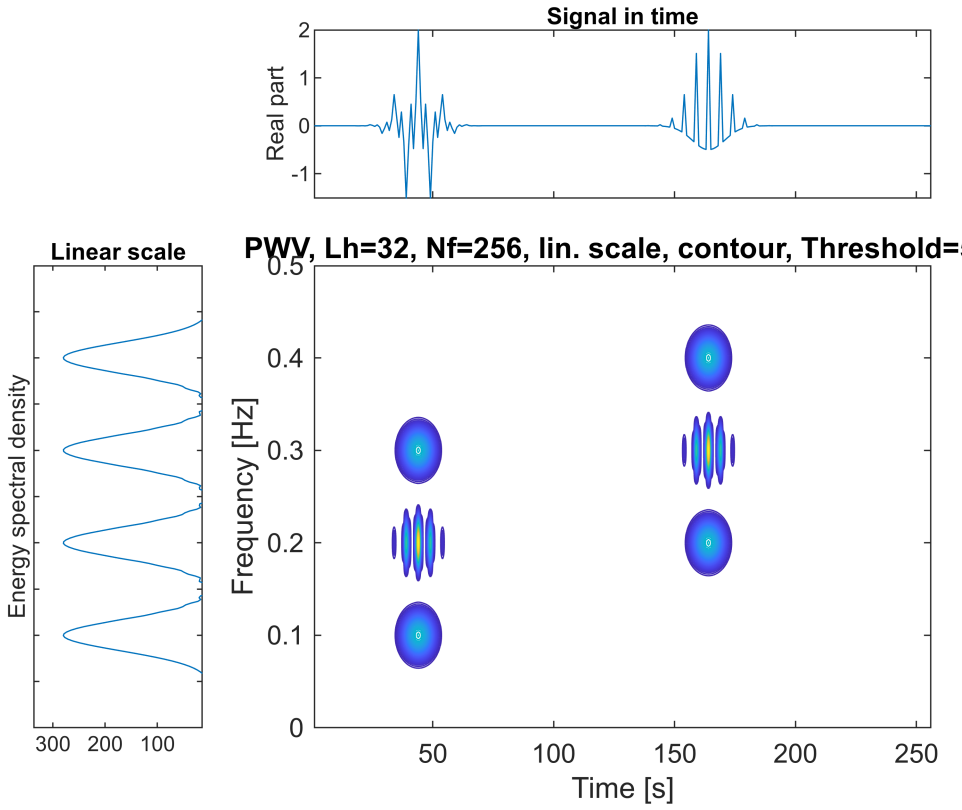

The cross term disappered! But the tf resolution decreased by a little bit

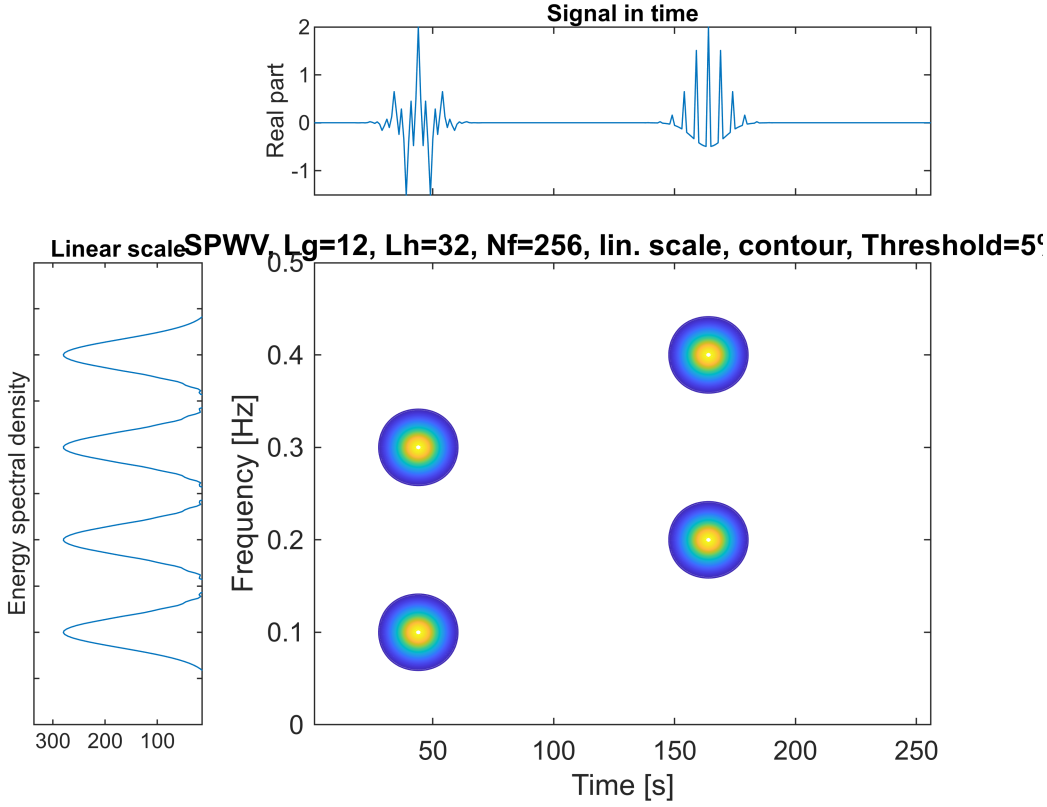
Let's try a more complicated signal

```
tfrwv(x9);
tfrpwv(x9);
```




```
tfrspwv(x9);
```




PWVD convolves the WVD with a window function. So it eliminates the cross terms along the time axis.

SPWVD perform frequency filtering on top of PWVD, hence it reduces the cross-terms along both the time and frequency axes. But the resolution is reduced significantly.

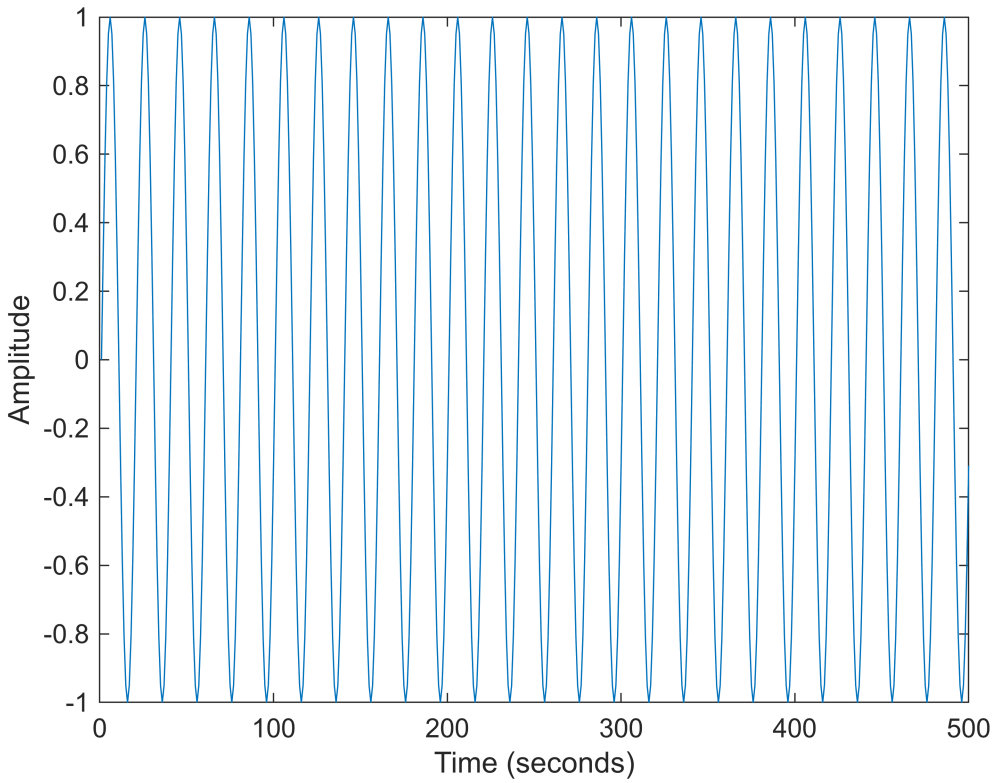
If you are interested to see more WVD results. Create your own signals and plot the outputs. You will see many interesting results!

The code below is to illustrate Uncertainty Principle

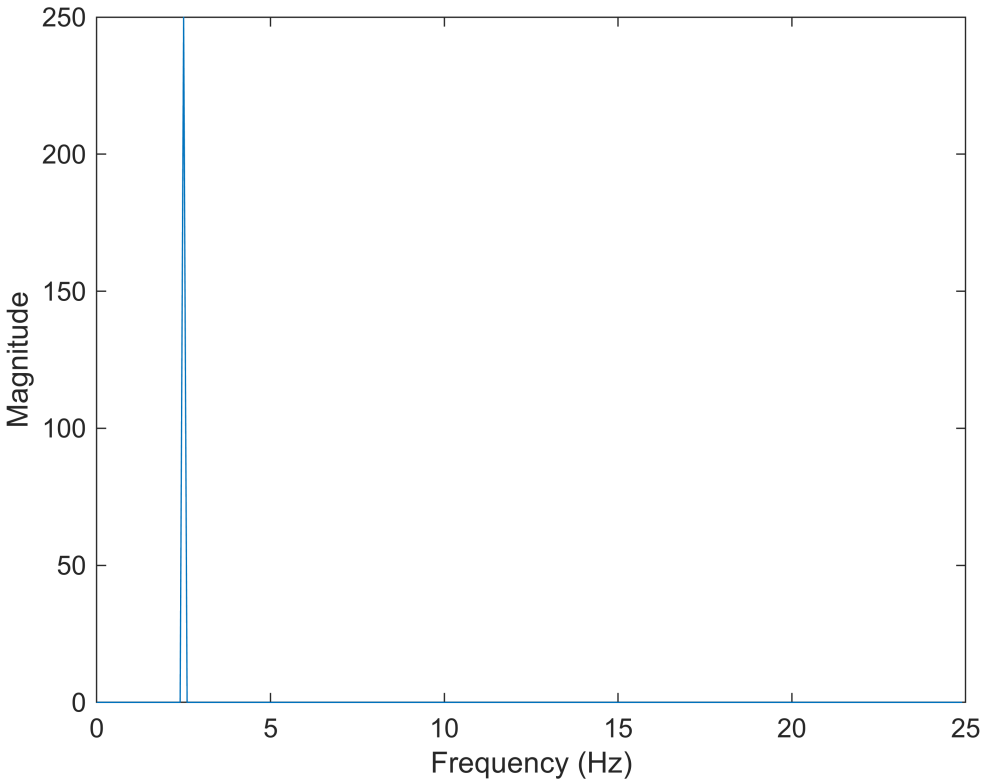
```
Ts = 1/50;
t = 0:Ts:10-Ts;
x = sin(5*pi*t);


y = fft(x);
fs = 1/Ts;
f = (0:length(y)-1)*fs/length(y);

figure
plot(x); xlabel('Time (seconds)'); ylabel('Amplitude')
```



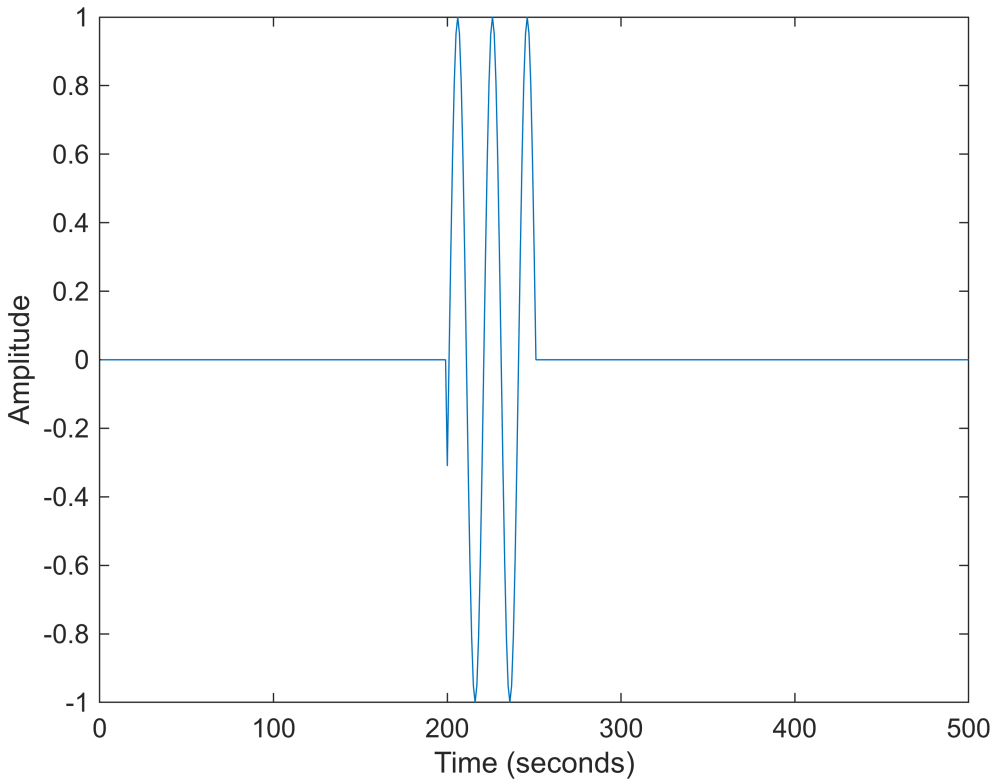

```
figure
plot(f(1:length(f)/2),abs(y(1:length(y)/2))); ylabel('Magnitude'); xlabel('Frequency (Hz)')
```




```
x1 = zeros(length(t),1);
x1(200:250) = x(200:250);

y1 = fft(x1);
fs = 1/Ts;
f = (0:length(y)-1)*fs/length(y);

figure
plot(x1); xlabel('Time (seconds)'); ylabel('Amplitude')
```




```
figure
plot(f(1:length(f)/2),abs(y1(1:length(y1)/2))); ylabel('Magnitude'); xlabel('Frequency (Hz)')
```