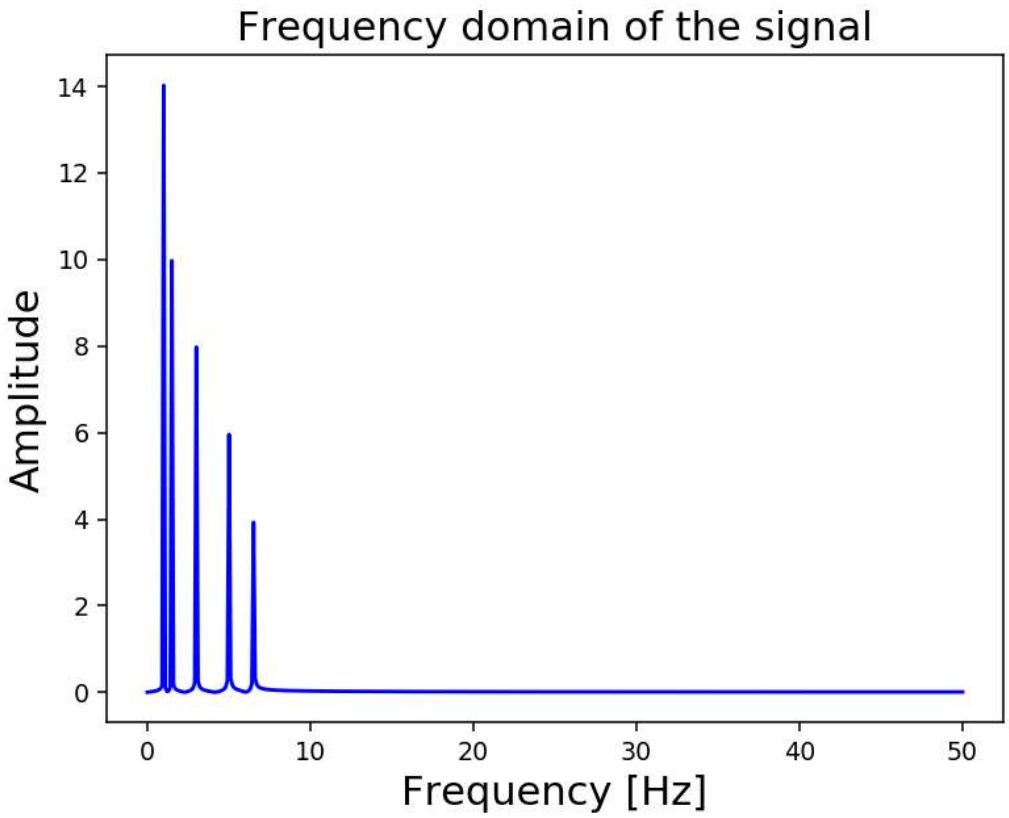
Test SN

FFT in Python

In [25]:

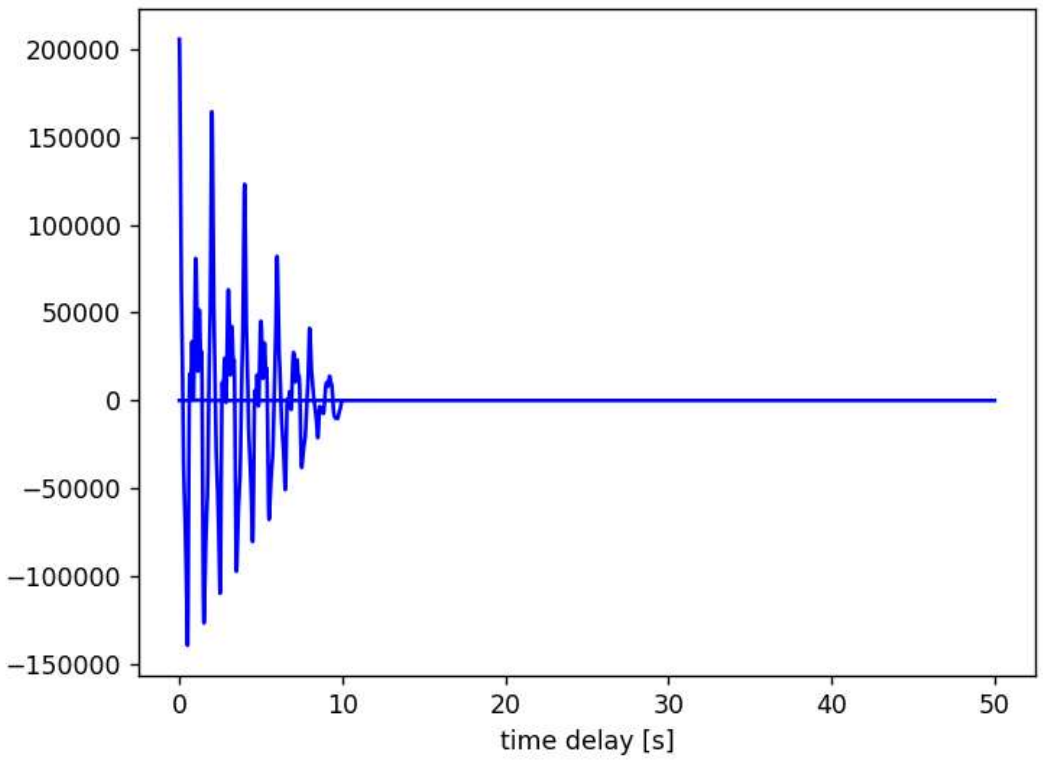
```
from scipy.fftpack import fft
import matplotlib.pyplot as plt
%matplotlib notebook

def get_fft_values(y_values, T, N, f_s):
    f_values = np.linspace(0.0, 1.0/(2.0*T), N//2)
    fft_values_ = fft(y_values)
    fft_values = 2.0/N * np.abs(fft_values_[0:N//2])
    return f_values, fft_values

t_n = 10
N = 1000
T = t_n / N
f_s = 1/T

f_values, fft_values = get_fft_values(composite_y_value, T, N, f_s)

plt.plot(f_values, fft_values, linestyle='-', color='blue')
plt.xlabel('Frequency [Hz]', fontsize=16)
plt.ylabel('Amplitude', fontsize=16)
plt.title("Frequency domain of the signal", fontsize=16)
plt.show()
```



PSD in Python

In [21]:

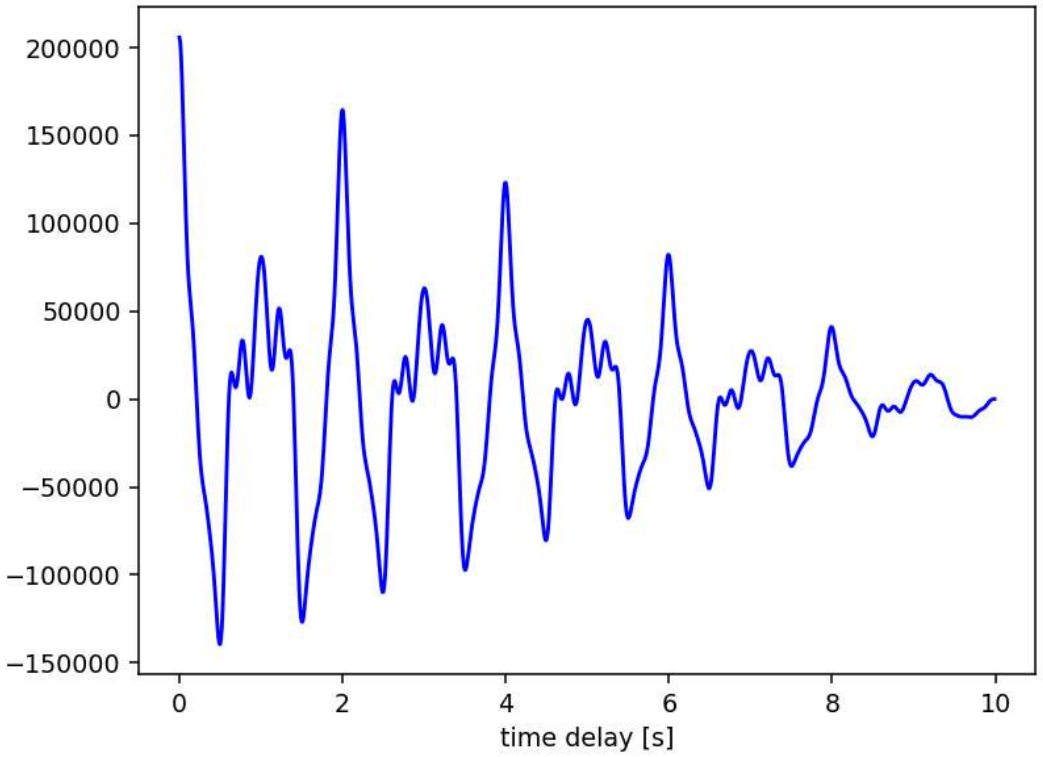
```
from scipy.signal import welch
import matplotlib.pyplot as plt
%matplotlib notebook

def get_psd_values(y_values, T, N, f_s):
    f_values, psd_values = welch(y_values, fs=f_s)
    return f_values, psd_values


t_n = 10
N = 1000
T = t_n / N
f_s = 1/T

f_values, psd_values = get_psd_values(composite_y_value, T, N, f_s)

plt.plot(f_values, psd_values, linestyle='-', color='blue')
plt.xlabel('Frequency [Hz]')
plt.ylabel('PSD [V**2 / Hz]')
plt.show()
```



Calculating the auto-correlation in Python

In [24]:

```
import matplotlib.pyplot as plt
%matplotlib notebook

def autocorr(x):
    result = np.correlate(x, x, mode='full')
    return result[len(result)//2:]

def get_autocorr_values(y_values, T, N, f_s):
    autocorr_values = autocorr(y_values)
    x_values = np.array([T * jj for jj in range(0, N)])
    return x_values, autocorr_values

t_n = 10
N = 1000
T = t_n / N
f_s = 1/T

t_values, autocorr_values = get_autocorr_values(composite_y_value, T, N, f_s)

plt.plot(t_values, autocorr_values, linestyle='-', color='blue')
plt.xlabel('time delay [s]')
plt.ylabel('Autocorrelation amplitude')
plt.show()
```



In [ ]:

```
```