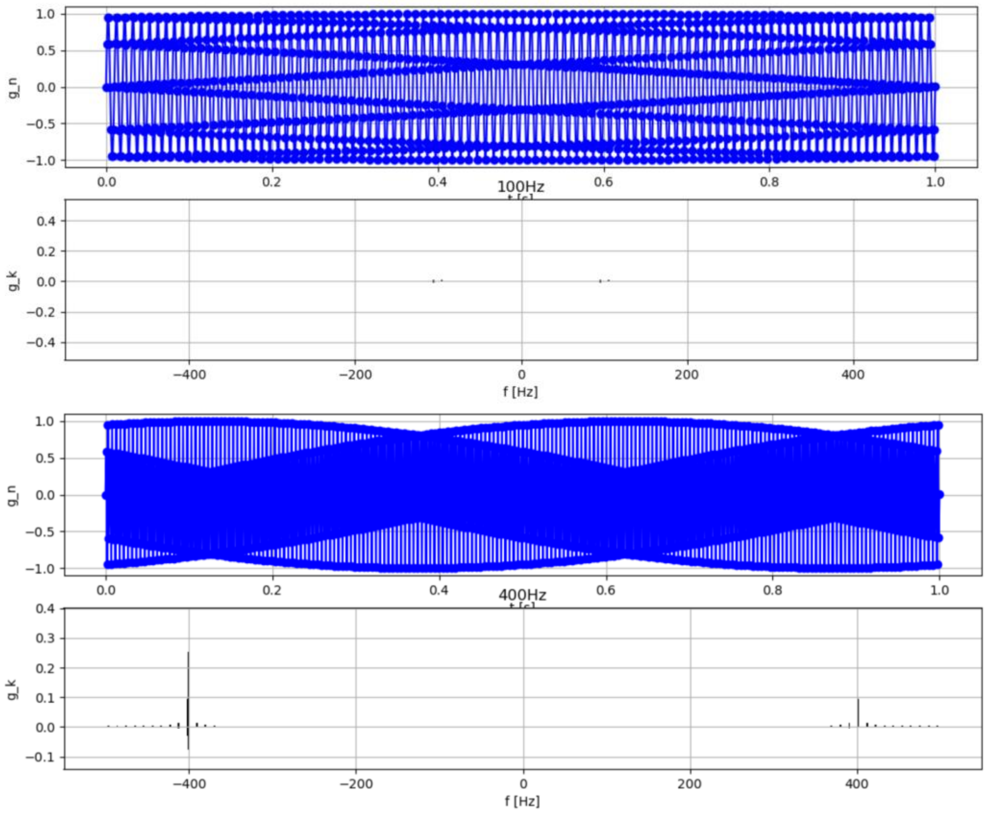
FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

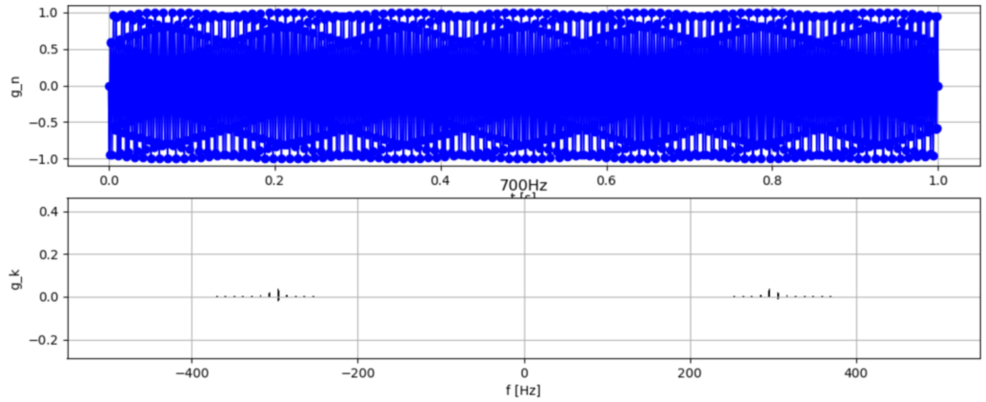
Oppgave 1

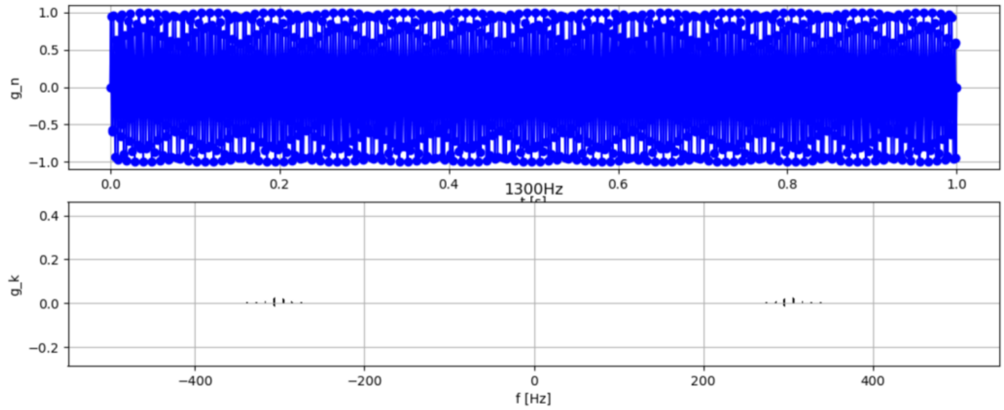
a) Er amplituden på den Fouriertransformerte som forventet? Ja, på grafen så ser vi at ved -100hz og 100hz får vi avlesninger.

b) Vi ser at 700Hz og 1300Hz ikke gir det vi forventer og dette er på grunn av nyquist frekvensen som er den maksimale frekvensen der aliasing vil forekomme.



FYS2130 regneoppgaver 6
Vibishan Raveendrarajah





Oppgave 2 utregning:

FYS2130 oblig uke 6

2) $X_k = \frac{1}{N} \sum_{n=0}^{N} x_n e^{-i \frac{2\pi}{N} kn}$

når $k=0$ så blir $e^{-i \frac{2\pi}{n} kn} = e^0 = 1$

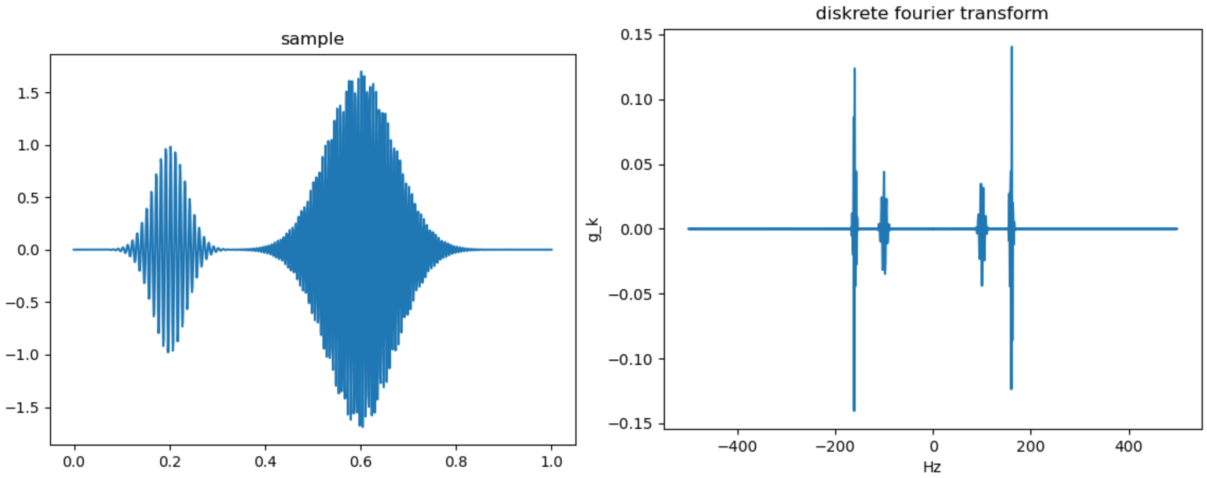
da er $X_0 = \frac{1}{N} \sum_{n=0}^{N-1} X_n$ og da er summen av

$\frac{\sum X_n}{N}$ = gjennomsnitts verdien til start signalet

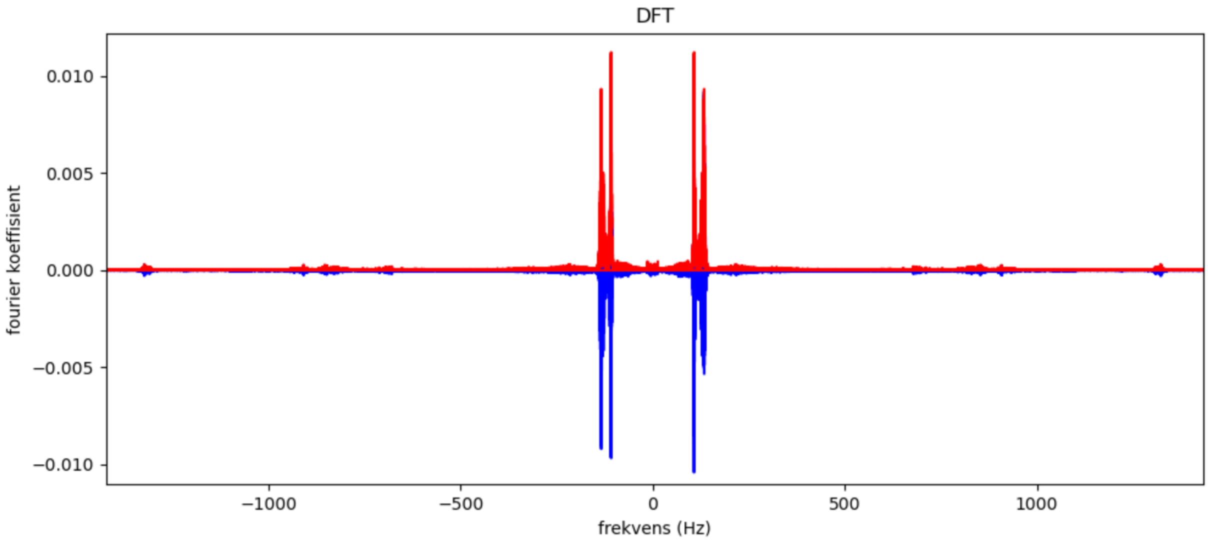
Kode vedlagt nede*

FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

Oppgave 3a



Oppgave 4a



I terminalen: sample rate: 44100

Oppgave 4b)
I følge samplings teoremet må samplingsfrekvensen være minst dobbelt av den maksimale frekvensen man vil gjenopprette, siden mennesker klarer å høre i 20Hz til 20,000Hz området må da samplingsfrekvensen være dobbelt av det maksimale ønsket frekvensen. Derfor er samplings frekvensen rundt 40,000Hz.

Kode

Kode oppgave 1:

```
import numpy as np
import matplotlib.pyplot as plt
from numpy import pi, sin
"""
#oppgave 1
A = 1
T = 1 #total samplings tid
f_samp = 1000 # samplingsfrekvens  1KHz = 1000Hz
N = T*f_samp; # total samplingspunkter
f = 100 #100Hz
dt = 1/f_samp
t = np.linspace(0, T, N) #tids array

for i in range (4):
    ti = ['100Hz', '400Hz', '700Hz', '1300Hz']
    f = [100, 400, 700, 1300]

    g = A*sin(2*pi*f[i]*t)

    # regner DFT via FFT
    g_k = (1/N)*np.fft.fft(g)
    freq = np.fft.fftfreq(N, dt)

    fig, ax = plt.subplots(2,1, figsize = (13, 5))
    ax[0].grid(1)
    ax[0].plot(t, g, color='blue', linestyle='solid', marker='o')
    ax[0].set_xlabel('t [s]')
    ax[0].set_ylabel('g_n')

    ax[1].grid(1)
    ax[1].bar(freq, np.imag(g_k), color='black', width=0.2)
    ax[1].bar(freq, np.abs(g_k), color='black', width=0.2) #
    ax[1].set_xlabel('f [Hz]')
    ax[1].set_ylabel('g_k')
    plt.title(ti[i])

    plt.show()
```

FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

Kode oppgave 2:

```
import matplotlib.pyplot as plt
import numpy as np
from numpy import sin, pi

N = 1000
X = np.linspace(-2*pi, 2*pi, N)
g = sin(X)**2
mean = sum(g/N)
g_k = (1/N)*np.fft.fft(g)

print('gjennomsnitt', mean)
print('første komponent', g_k[0])

plt.plot(X, g)
plt.axhline(y = mean, color = 'r', linestyle = '-')
plt.show()
```

Terminal: gjennomsnitt 0.49950000000000017 første komponent (0.4995+0j)

Kode oppgave 3a:

```
import matplotlib.pyplot as plt
import numpy as np
from numpy import sin, exp, pi, imag
A1, A2 = 1, 1.7      #amplitude
f1, f2 = 100, 160    #freq in Hz
t1, t2 = 0.2, 0.6    #seconds
std1, std2 = 0.05, 0.1 #standard deviation
T = 1
N = 1000
t = np.linspace(0,1,N)
dt = T/N
f = A1*sin(2*pi*f1*t)*exp(-((t-t1)/std1)**2) + A2*sin(2*pi*f2*t)*exp(-((t-
t2)/std2)**2)
#samplede tidsserien
plt.plot(t, f)
plt.title('sample')
plt.show()

#DFT via fft
g_k = (1/N)*np.fft.fft(f)
freq = np.fft.fftfreq(N, dt)

plt.plot(freq, imag(g_k))
plt.title('diskrete fourier transform')
plt.xlabel('Hz')
plt.ylabel('g_k')
plt.show()
```

FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

kode oppgave 3b:

```
import matplotlib.pyplot as plt

import numpy as np
from numpy import sin, exp, pi, imag

A1, A2 = 1, 1.7      #amplitude
f1, f2 = 100, 160    #freq in Hz
t1, t2 = 0.2, 0.6    #seconds
std1, std2 = 0.05, 0.1 #standard deviation
T = 1
N = 1000
t = np.linspace(0,1,N)
dt = T/N
f = A1*sin(2*pi*f1*t)*exp(-((t-t1)/std1)**2) + A2*sin(2*pi*f2*t)*exp(-((t-
t2)/std2)**2)

#DFT via fft
g_k = (1/N)*np.fft.fft(f)
freq = np.fft.fftfreq(N, dt)

def wavelet(K, w, tk, tn): #wavelet formula (14.8)
    cplx = np.complex(0, 1)
    fs = A1*sin(2*pi*f1*t)*exp(-((t-t1)/std1)**2) + A2*sin(2*pi*f2*t)*exp(-
((t-t2)/std2)**2)
    C = 0.798*w/(fs*K1)  #formula (14.7)
    return C*(exp(-cplx*w*(tn-tk)) - exp(-K**2)) * exp(-w*2*(tn-
tk)**2 / (2*K)**2)

def wavelet_transform(K, w, tk, tn, x, N): #wavelet transform (14.9)
    gamma = np.zeros(N, dtype=np.complex_)
    for i in range(N):
        print(w)
        gamma[i] = x* np.conjugate(wavelet(K, w, tk, tn))
    return gamma

def wavelet_diagram(K, w, tk, tn, x, N): #der w er en liste
    wav_list = np.zeros(N)
    for i in range(N):
        wav_list[i] = wavelet_transform(K, w[i], tk, tn, x[i], N)

N = 100 #analyse
w = np.logspace(80,200, num = N)
K1 = 6
k2 = 60
fs = A1*sin(2*pi*f1*t)*exp(-((t-t1)/std1)**2) + A2*sin(2*pi*f2*t)*exp(-((t-
t2)/std2)**2)
x = np.linspace(-2*pi, 2*pi, N)
wavelet_diagram(K1, w, t1, t2, x, N)
```

FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

```
Oppgave 3C funker ikke, men skrev mest av det som trengs, problemet var bare med wavelet transform. Oppgave 3c er med oppgave 3b koden.
```

Opggave 4a

```
#oppgave 4a
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile

f_s , data = wavfile.read('cuckoo.wav')
print('sample rate:',f_s)
N = data.shape[0]
T = N/f_s
x = data[:,0]
f_c = (1/N) *np.fft.fft(x)
freq = np.fft.fftfreq(N, T/f_s)

plt.plot(freq, f_c)
plt.title('DFT')
plt.xlabel('frekvens')
plt.ylabel('fourier koeffisient')
plt.show()
s
```

Oppgave 4c) funker ikke

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile

f_s , data = wavfile.read('cuckoo.wav')
print('sample rate:',f_s)
N = data.shape[0]
T = N/f_s
x = data[:,0]

samp_n = 20
t1, t2 = 0.4, 1.1
x_n = data[np.int_(t1*f_s):np.int_(t2*f_s):samp_n, 0]

t = (1/f_s)*np.linspace(t1, t2, len(x_n))
# calc DFT via FFT
x_g = (1/(len(x_n)))*np.fft.fft(x_n)
freq = np.fft.fftfreq(len(x_n), 1/f_s)


"""
plt.plot(freq, np.imag(f_c), color ='blue')
```

FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

```
plt.plot(freq, np.abs(f_c), color ='red')
plt.title('DFT')
plt.xlabel('frekvens (Hz)')
plt.ylabel('fourier koeffisient')
plt.show()
"""
plt.plot(t, x_n, color='blue', linestyle='solid', linewidth=0.2)
plt.plot(freq, np.abs(x_g[:len(x_n)]), color='black', linewidth=0.5)
plt.show()
```