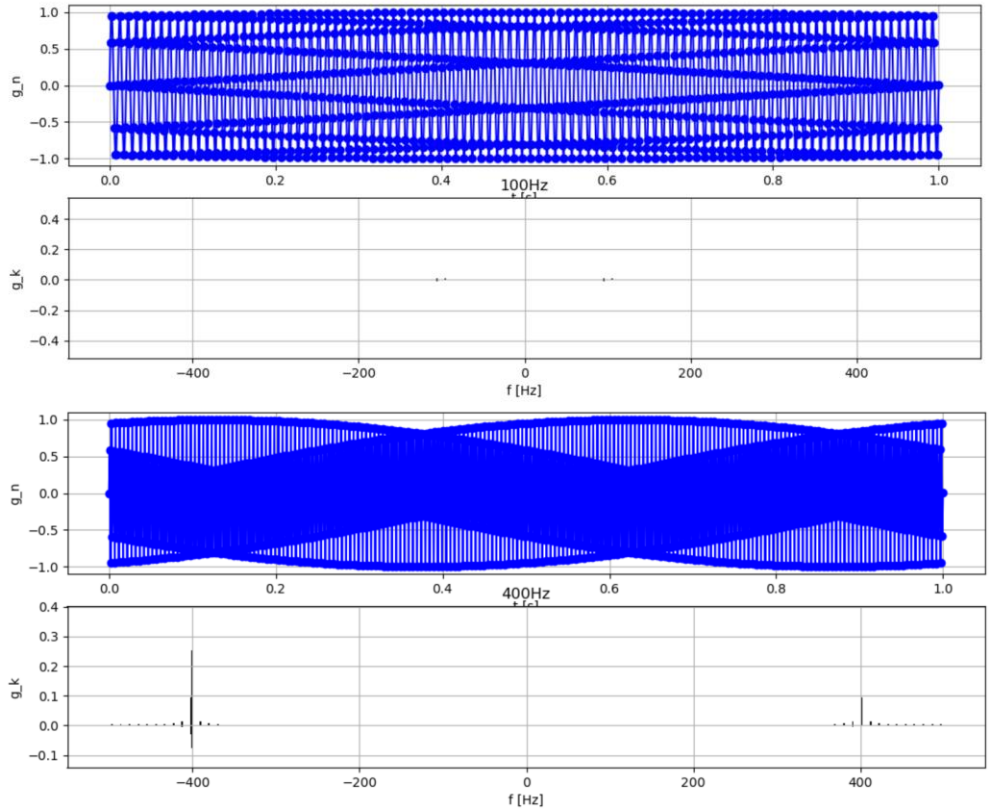
FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

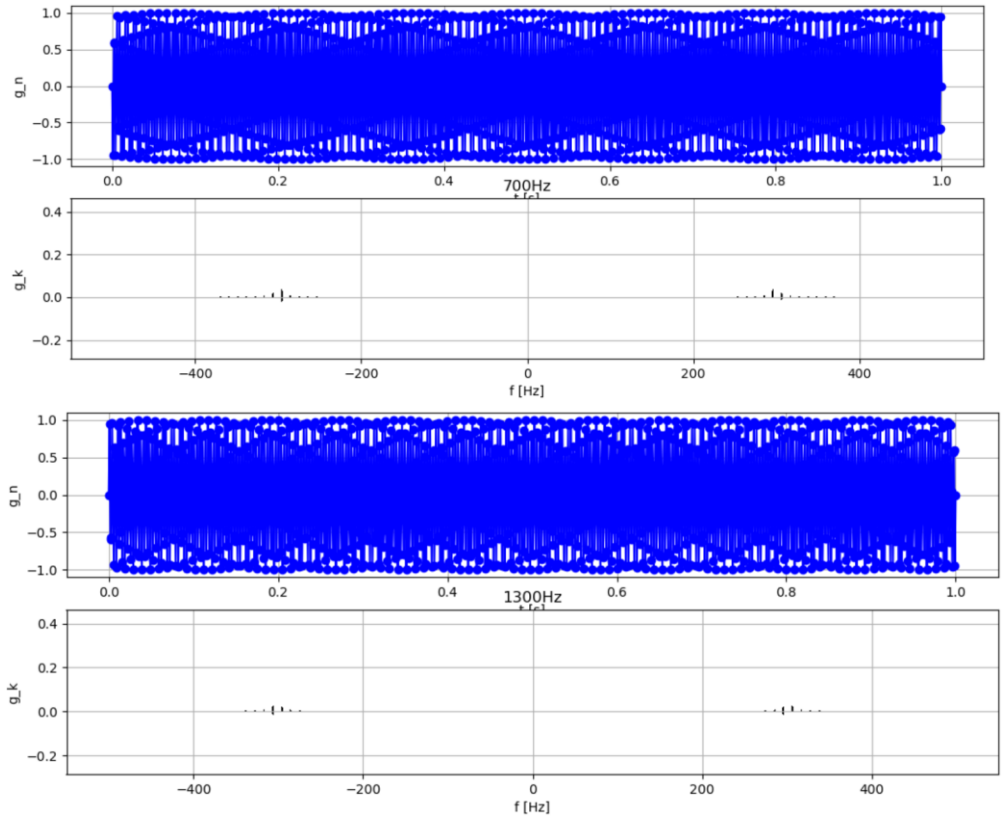
Oppgave 1

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

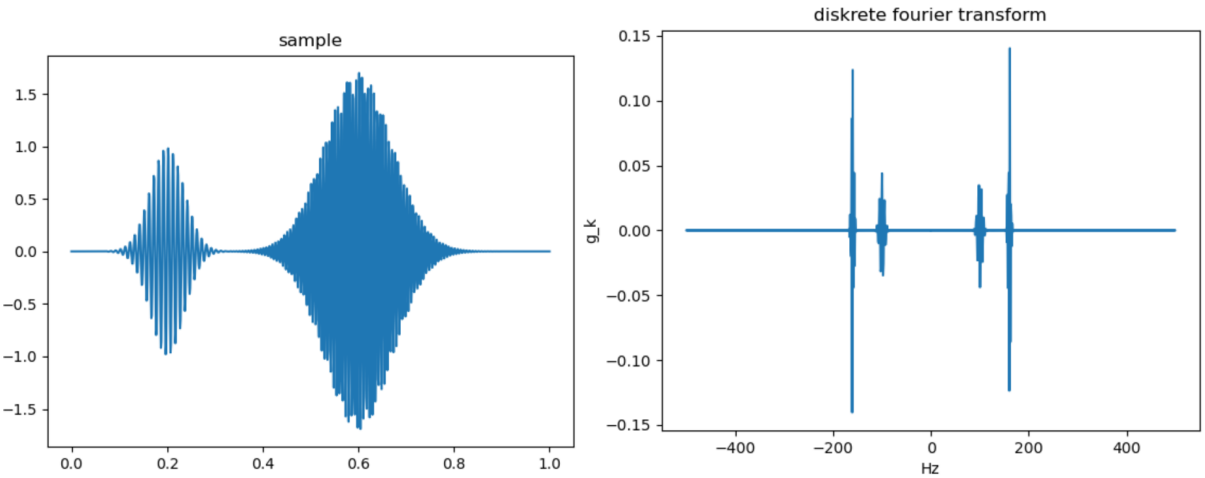
da er $X_0 = \frac{1}{N}\sum_{n=0}^{N-1} x_n$ og da er Summen av

$\frac{\Sigma x_n}{N}$ = gjennomsnitts verdien til start signalet

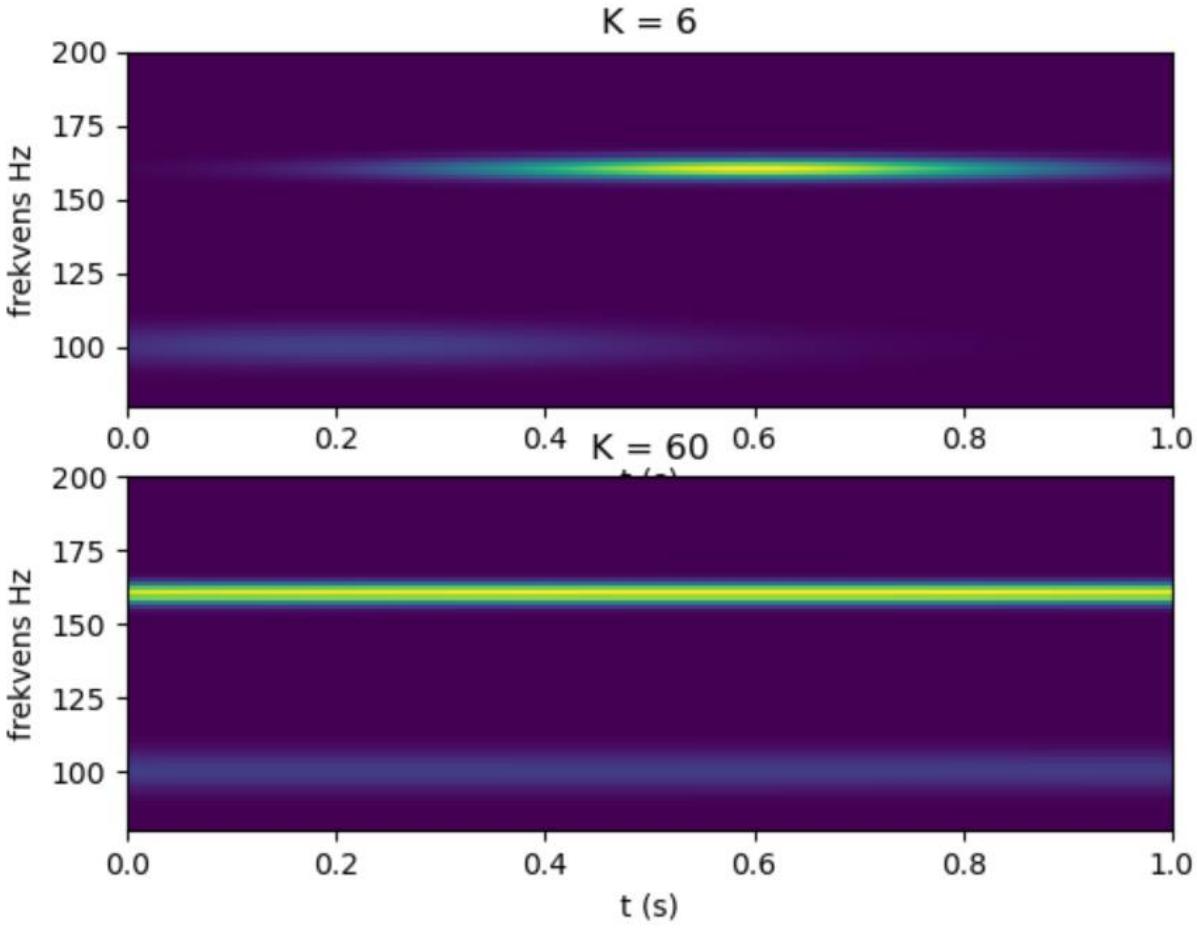
Kode vedlagt nede*

FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

Oppgave 3a

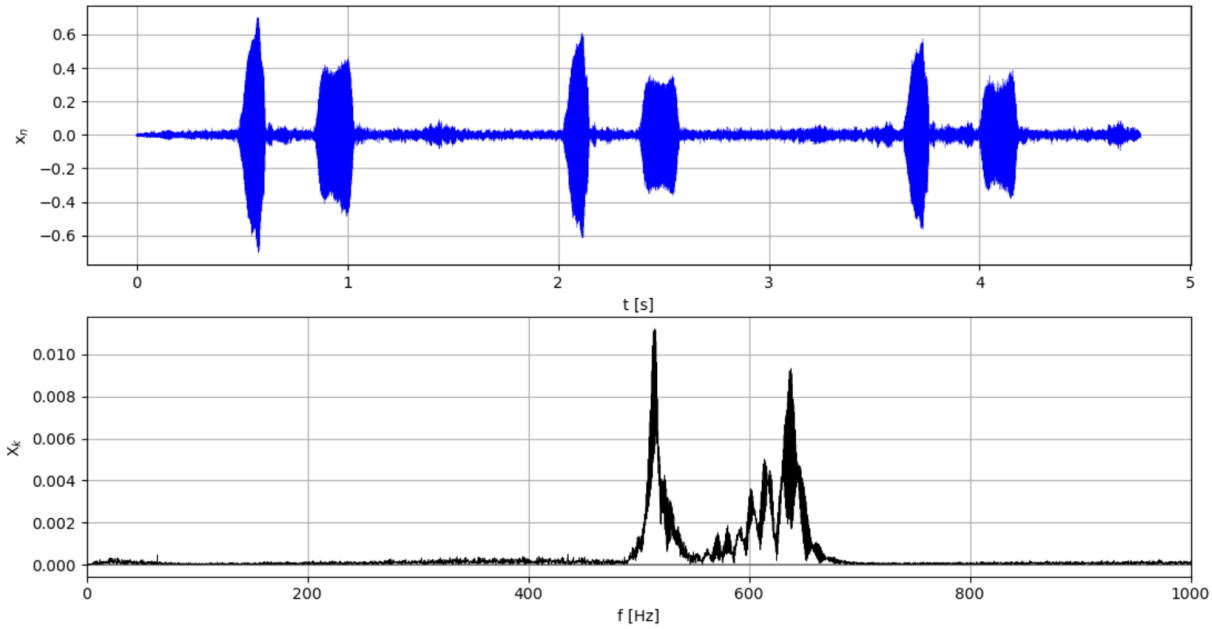


Oppgave 3c



FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

Oppgave 4a



I dene oppgaven for å finne frekvens området til wavleteanalysen mp vi ta FFT på lydopptaket, og fra graffen ser vi at 500Hz til circa 650Hz er frekvensen vi bør ta wavelet analyse i.

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.

Oppgave 4C) se kode nede, var ikke helt ferdig med den

FYS2130 regneoppgaver 6
Vibishan Raveendrarajah

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, 3c:

Oppgave 3C funker ikke helt, vet ikke hva som er feil. (Oppgave 3c er med oppgave 3b koden)

```
import numpy as np
import matplotlib.pyplot as plt
from numpy import sin, exp, pi, log10

A1, A2 = 1, 1.7
f1, f2 = 100, 160
t1, t2 = 0.20, 0.60
sigma1, sigma2 = 0.05, 0.10
T = 1
fs = 1000
N = int(T*fs)
dt = T/N
t = np.linspace(0,T,N)
f = A1*sin(2*pi*f1*t)*exp(-((t-t1)/sigma1)**2) + A2*sin(2*pi*f2*t)*exp(-((t-
t2)/sigma2)**2)

def wavelet(tn,tk,wa,K):
    cplx = np.complex(0,1)
    C = 0.798*wa/(fs*K)
    return C*(exp(-cplx*wa*(tn-tk))-exp(-K**2)) * exp(-wa*2*(tn-
tk)**2/(2*K)**2)

def wavelet_transform(tn,tk,wa,K):
    gamma = 0
    for n in range(len(t)):
        gamma += f[n]*np.conjugate(wavelet(tn[n],tk,wa,K))
    return gamma

def wavelet_diagram(tn,tk,wa,K):
    a, b = np.meshgrid(tk,wa)
    A = wavelet_transform(tn,a, b, K)
    plt.pcolormesh(a, b/(2*pi), abs(A))

K = 6
wa = np.logspace(log10(80), log10(200), 100)*2*pi
tk = np.linspace(0, T, 1000)

plt.subplot(2,1,1)
wavelet_diagram(t,t,wa,6)
plt.title('K = 6')
plt.xlabel('t (s)')
plt.ylabel('frekvens Hz')

plt.subplot(2,1,2)
wavelet_diagram(t,t,wa,60)
```

```
plt.title('K = 60')
plt.xlabel('t (s)')
plt.ylabel('frekvens Hz')

plt.show()
```

Opggave 4a (bruker mye av kode fra forelesningen)

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

# t series
f_s, data = wavfile.read('cuckoo.wav')
x_n = data[:,0]
N = data.shape[0]
print(f_s)
T = N/f_s
dt = 1/f_s
t1 = 0.4
t2 = 1.1

dt = 1 / f_s
t = dt*np.linspace(0., N, data.shape[0])

#t_n = t[np.int_(t1*f_s):np.int_(t2*f_s):N]

# calc DFT via FFT
X_k = (1/N)*np.fft.fft(x_n)
freq = np.fft.fftfreq(N, dt)

#short-t fourier transform
t_STFT, t_STFT, Fxx = signal.stft(x_n, f_s, nperseg=2000)

fig, ax = plt.subplots(2,1, figsize = (13, 8))
ax[0].grid(1)
ax[1].grid(1)
ax[0].plot(t, x_n, color='blue', linestyle='solid', linewidth=0.2)
ax[0].set_xlabel('t [s]')
ax[0].set_ylabel('x$_n$')
#
ax[1].plot(freq, np.abs(X_k[:N]), color='black', linewidth=0.5)
ax[1].set_xlabel('f [Hz]')
ax[1].set_ylabel('X$_k$')
ax[1].set_xlim([0, 1000])
plt.show()
```

```
plt.pcolormesh(t_STFT, f_STFT, np.abs(Fxx), shading='gouraud')
plt.xlabel('t [s]')
plt.ylabel('f [Hz]')
plt.ylim([0, 1000])
```

Oppgave 4c) funker ikke ☹

```
import numpy as np
import matplotlib.pyplot as plt
from numpy import sin, exp, pi, log10


f_s , data = wavfile.read('cuckoo.wav')
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]



def wavelet(tn,tk,wa,K):
    cplx = np.complex(0,1)
    C = 0.798*wa/(fs*K)
    return C*(exp(-cplx*wa*(tn-tk))-exp(-K**2)) * exp(-wa*2*(tn-
tk)**2/(2*K)**2)

def wavelet_transform(tn,tk,wa,K):
    gamma = 0
    for n in range(len(t)):
        gamma += f[n]*np.conjugate(wavelet(tn[n],tk,wa,K))
    return gamma

def wavelet_diagram(tn,tk,wa,K):
    a, b = np.meshgrid(tk,wa)
    A = wavelet_transform(tn,a, b, K)
    plt.pcolormesh(a, b/(2*pi), abs(A))
K = 6
wa = np.logspace(log10(80), log10(200), 100)*2*pi
wavelet_diagram(t,t,wa,60)
plt.title('K = 60')
plt.xlabel('t (s)')
plt.ylabel('frekvens Hz')
plt.show()
```