CE - Bjørn Magnus Hoddevik

April 29, 2022

1 Hvordan kan vi simmulere støydempingen til en vegg?

1.0.1 Skrevet av Bjørn Magnus Hoddevik

1.1 Problemstilling

Om vi har en lydfil, hvilke antagelser og data trenger vi for å kunne "realistisk" simmulere en vegg. Lar det seg i det heletatt løses?

1.2 Oppgaven

Vi har likninger for hvordan bølger blir reflektert av en overflate og blir absorbert avhengig av impedans. Tanken er å lag et program som tar en lydfil og sender den gjennom en simulert vegg med ulik impedans avhengig av materialets egenskaper.

For hva som blir reflektert av veggen skriver vi:

$$R = \frac{A_r}{A_i} = \frac{(z_a - z_b)^2}{(z_a + z_b)^2} \tag{1}$$

og det som går igjennom:

$$T = \frac{A_t}{A_i} = \frac{4 z_a z_b}{(z_a + z_b)^2} \tag{2}$$

Hvor A er amplituden til t det som går igjennom, r reflektert og i bølgen før den treffer veggen. z er impedansen til de ulike materialene. Vi har flere ulike mulige konsekvenser avhengig av svaret på disse likningene.

Vi starter med å importere nødvendige biblioteker:

[1]:
```
import matplotlib.pyplot as plt
import numpy as np
import scipy.signal
from scipy.io import wavfile
```

Deretter trenger vi å uttrykke de to likningene over:

[2]:
```
def calc_R(za, zb):
    R = (za-zb)**2/(za+zb)**2
    return R

def calc_T(za, zb):
    T = 4*za*zb/(za+zb)**2
```

```
    return T
```

Videre trenger vi å finne ulike impedanser til ulike materialer vi ønsker å teste ut:

[3]:
```
Air = 413
Wood = 15.7e6
Brick = 7.4e6
Marble = 10.5e6
Glass = 13e6
```

Disse verdiene er avhengig av mye annet, som for eksempel temperatur, men fungerer for å gi oss et godt intrykk av hvordan lyden vil høres ut etter at den treffer veggen.

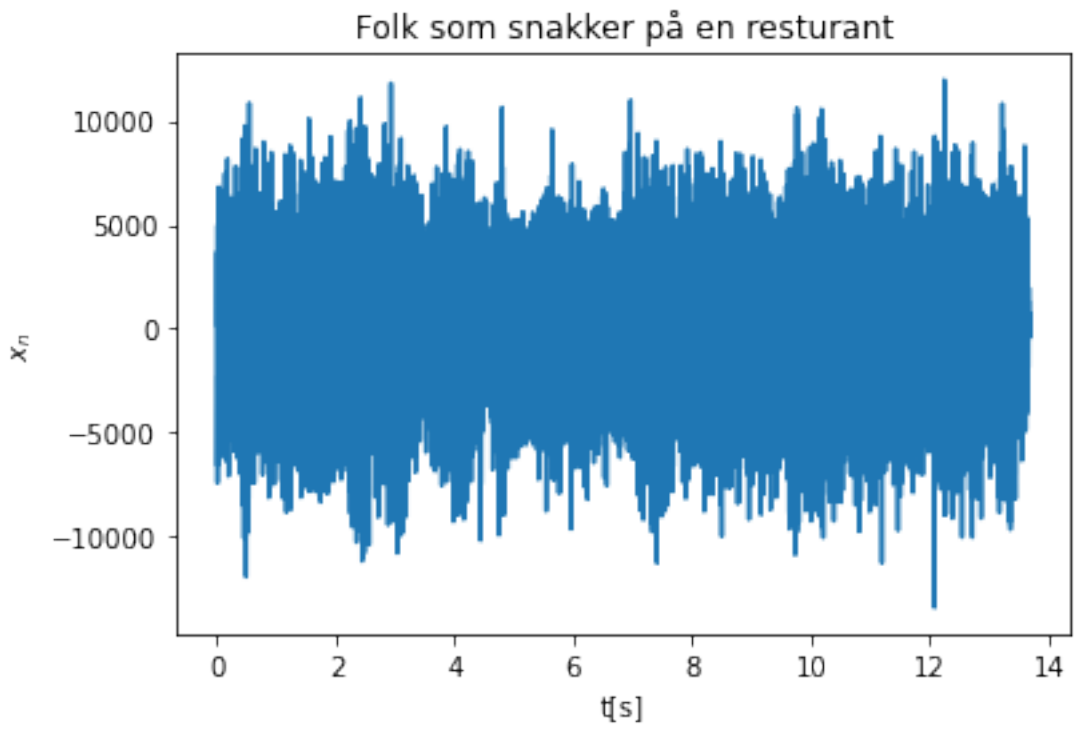
Videre trenger vi en lydfil med det vi ønsker å høre på. Jeg har valgt et lydklipp med bakgrunnstøy fra en resturant.

[4]:
```
fs, data = wavfile.read("people_talking.wav")

people_talking = data[:, 0]

N = len(people_talking)
T = N/fs
t = np.linspace(0, T, N)
```

Vi kan se på hvordan lydfilen ser ut først:

[5]:
```
plt.plot(t, people_talking)
plt.title("Folk som snakker på en resturant")
plt.xlabel("t[s]")
plt.ylabel("$x_n$")
```

[5]:
```
Text(0, 0.5, '$x_n$')
```



Vi ser at filen er sånn ca. 13-14sekunder lang uten mye karakteristiske egenskaper.

Videre trenger vi en funksjon som tar inn to ulike akustiske impedanser og en lydfil og deretter retunerer en endret lydfil. Vi ønsker også å se hvor stor prosentandel som blir reflektert og går igjennom.

[6]:
```
def wall(x, za, zb):
    Ai = np.copy(x)
    T = calc_T(za, zb)
    R = calc_R(za, zb)
    print(f"Veggen reflekterer R={abs(100*R):.2f}% og tar imot T={abs(100*T):.
 ↪2f}%")
    return np.asarray([R*Ai, T*Ai])
```

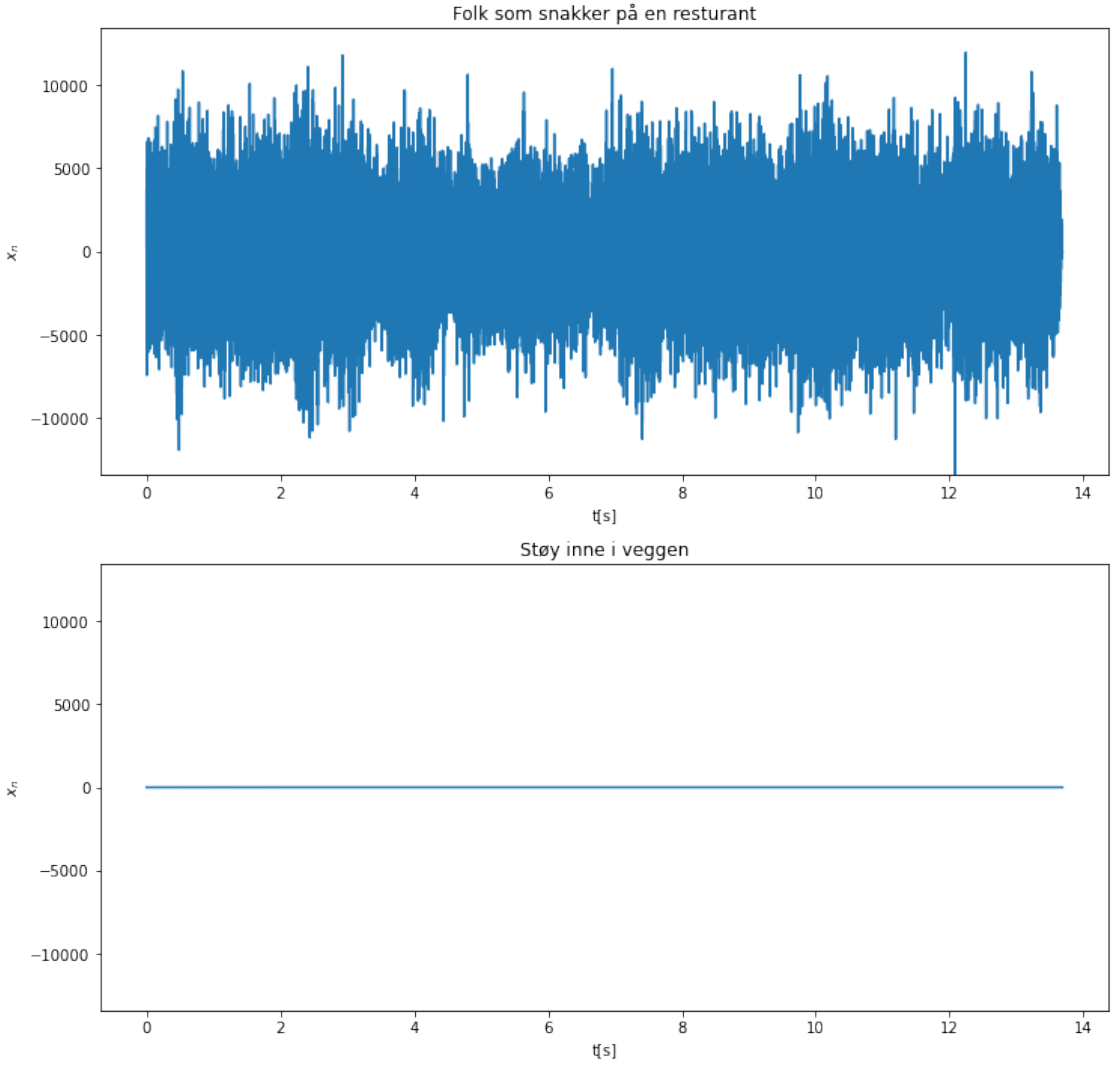
Tester først ut at lyden går igjennom en trevegg fra luft:

[7]:
```
people_talking_WoodAir = wall(people_talking, Air, Wood)
```

```
Veggen reflekterer R=99.99% og tar imot T=0.01%
```

[8]:
```
people_talking_WoodAir_max = np.max(abs(people_talking_WoodAir))

text = ["Folk som snakker på en resturant", "Støy inne i veggen"]
figs, ax = plt.subplots(2, 1, figsize=(12, 12))
for i in range(len(ax)):
```

```
    ax[i].plot(t, people_talking_WoodAir[i])
    ax[i].set_ylim(-people_talking_WoodAir_max, people_talking_WoodAir_max)
    ax[i].set_title(text[i])
    ax[i].set_xlabel("t[s]")
    ax[i].set_ylabel("$x_n$")
```



Som du ser er det ikke mye som skjer her, det er fordi begge grafene deler samme y-akse, så de små endringene i nederste graf er vanskelig å legge merke til. Men vi må huske på at vi fremdeles er inne i veggen, vi må ut igjen.

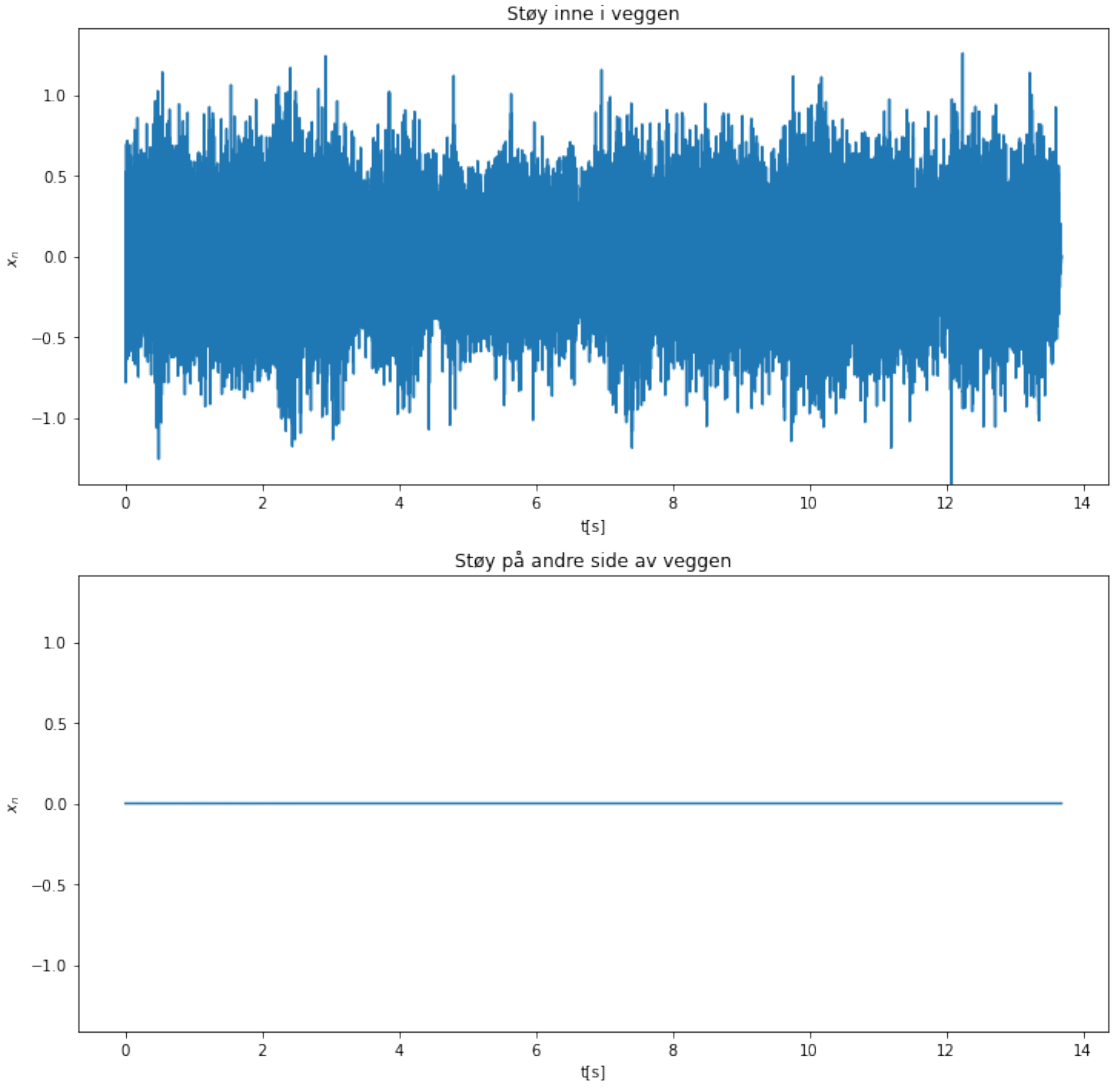
[9]:
```
people_talking_AirWood = wall(people_talking_WoodAir[1], Wood, Air)
```

```
Veggen reflekterer R=99.99% og tar imot T=0.01%
```

Vi plotter det igjen og ser forksjellen og deretter lager det om til en fil for å høre hvordan det har

endret seg:

[10]:
```
people_talking_AirWood_max = np.max(abs(people_talking_AirWood))

text = ["Støy inne i veggen", "Støy på andre side av veggen"]
figs, ax = plt.subplots(2, 1, figsize=(12, 12))
for i in range(len(ax)):
    ax[i].plot(t, people_talking_AirWood[i])
    ax[i].set_ylim(-people_talking_AirWood_max, people_talking_AirWood_max)
    ax[i].set_title(text[i])
    ax[i].set_xlabel("t[s]")
    ax[i].set_ylabel("$x_n$")
wavfile.write(f"people_talking_AirWood.wav", fs, people_talking.
 ↪astype("float32"))
```

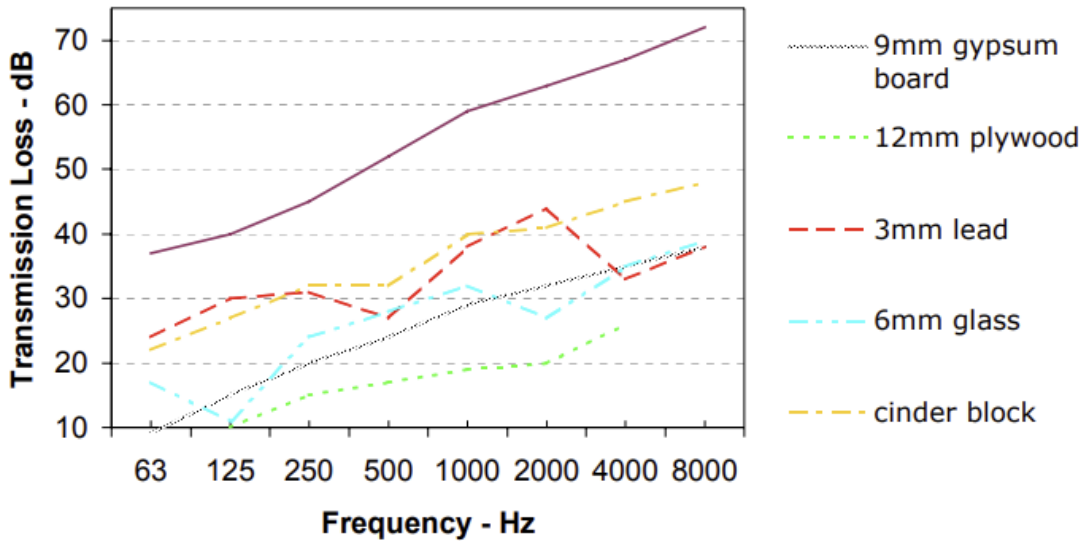


Så her ser vi at restene av bølgen som treffer veggen, slik vi har simulert det, ikke kan være grunnen til at vi hører folk igjennom veggene. Dette kan være på grunn av to grunner, modellen vår er for urealistisk og hadde vært bedre om vi hadde modelert den bedre, eller fordi det er et annet fysisk fenomen som fører til at vi hører gjennom veggene. Jeg lener meg mot den andre, ettersom måten vi beregnet dette var kun avhengig av den akustiske impedansen til materialene og brydde seg ikke om tykkelsen på veggene eller noe i nærheten. Dessuten er forskjellen mellom impedansen til tre og luft så forskjellige at det meste av bølgen vil uansett bli reflektert. Dette vil også være tilfelle for overgangen fra et hvert vegg materiale til luft, som vi ser tydelig under:

[11]:
```
wall_imp = [Wood, Brick, Marble, Glass]
wall_mat = ["Tre", "Murstein", "Marmor", "Glass"]
for imp, mat in zip(wall_imp, wall_mat):
    print(f"For materialet `{mat}` får vi at:")
    people_talking_wall = wall(people_talking, Air, imp)
    print("\n")
```

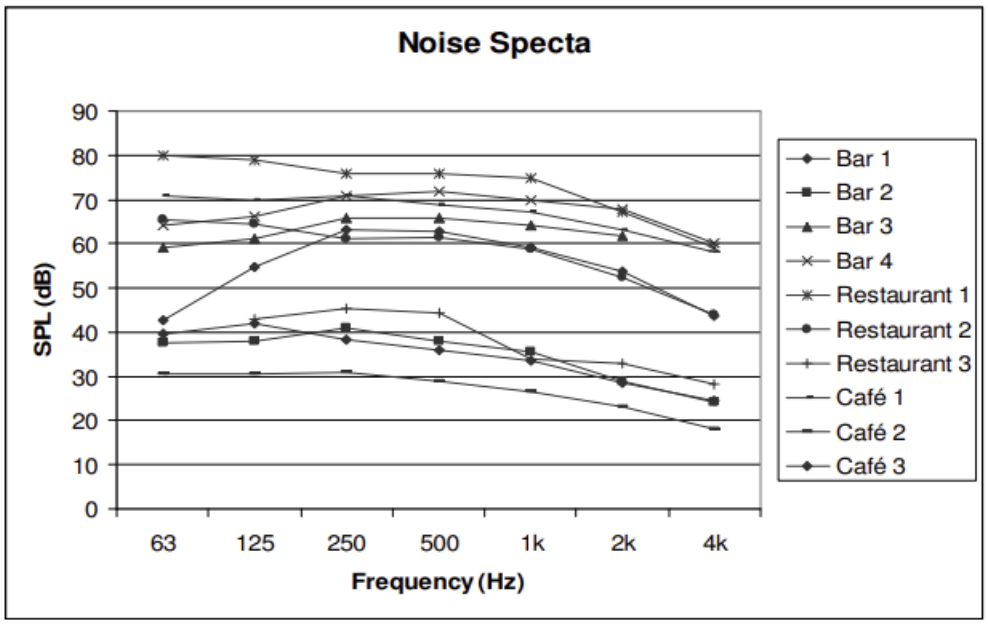
```
For materialet `Tre` får vi at:
Veggen reflekterer R=99.99% og tar imot T=0.01%


For materialet `Murstein` får vi at:
Veggen reflekterer R=99.98% og tar imot T=0.02%


For materialet `Marmor` får vi at:
Veggen reflekterer R=99.98% og tar imot T=0.02%


For materialet `Glass` får vi at:
Veggen reflekterer R=99.99% og tar imot T=0.01%
```

Så vi trenger altså en annen modell for å finne ut hvordan vi kan simulere en vegg på en resturant. For å gjøre dette ser vi på "STL", eller sound transmision loss, for ulike materialer. Denne modellen vil ta inn i betraktning både tykkelse på veggen, materiale og frekvensen til lyden som treffer veggen. Ulike materialer vil nemmelig være mer effektive mot lavere frekvenser enn de vil være til høyere frekvenser for eksempel, som vist i grafen under:



Vår modell vil ikke være like komplisert som over, men viser likevel et godt utgangspunkt. Fremgangsmåten blir da som følgende. Vi trenger å beregne volumet til en hver frekvens, deretter bruke den til å kalkulere STL og til slutt senke volumet og lagre den ferdige filen igjen.

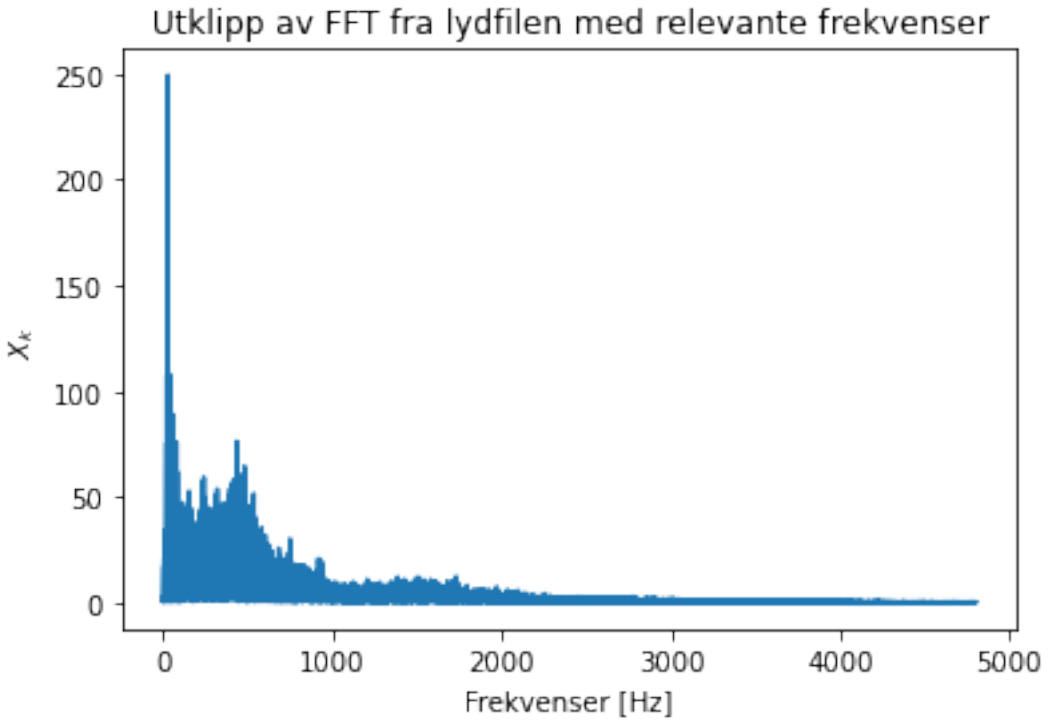
Vi starter med å ta for oss beregningen av volum, som vi ønsker å beregne i SPL, hvor $I_0 = 10^{-12} W/m^2$ og$I = p^2/\rho c = p^2/z$. Så vi mangler altså trykket for å finne en måte å kunne beregne volum. Dette viser seg å være svært vanskelig uten en referanse wav, som er tatt opp med samme utstyr og samme sted hvor volumet til tonen spilt av må være kjent fra før. Det har ikke jeg tilgang til og må heller derfor finne en annen måte å finne volum. I følge denne artikkelen har det gjennomsnittlig blitt målt volum på rundt 70dB(SPL), som du finner ved figur 1 også vist under.



Videre bruker vi Resturant 1 som utgangspunkt for våre målinger. Vi antar at volumet vokser lineært fra punkt til punkt. Når vi skal arbeide med frekvenser trenger vi også en FFT av filen.

[12]:
```
X_k = np.fft.fft(people_talking/N)
freq = np.fft.fftfreq(N, 1/fs)

plt.plot(freq[(0 < freq) & (freq < 4800)], abs(X_k[(0 < freq) & (freq < 4800)]))
plt.title("Utklipp av FFT fra lydfilen med relevante frekvenser")
plt.xlabel("Frekvenser [Hz]")
plt.ylabel("$X_k$")
```

[12]:
```
Text(0, 0.5, '$X_k$')
```



Legger inn punktene fra resturant 1.

[13]:
```
dbSPL_ref = [0, 80, 79, 77, 77, 75, 69, 60]
freq_ref = [0, 63, 125, 250, 500, 1000, 2000, 4000]
```

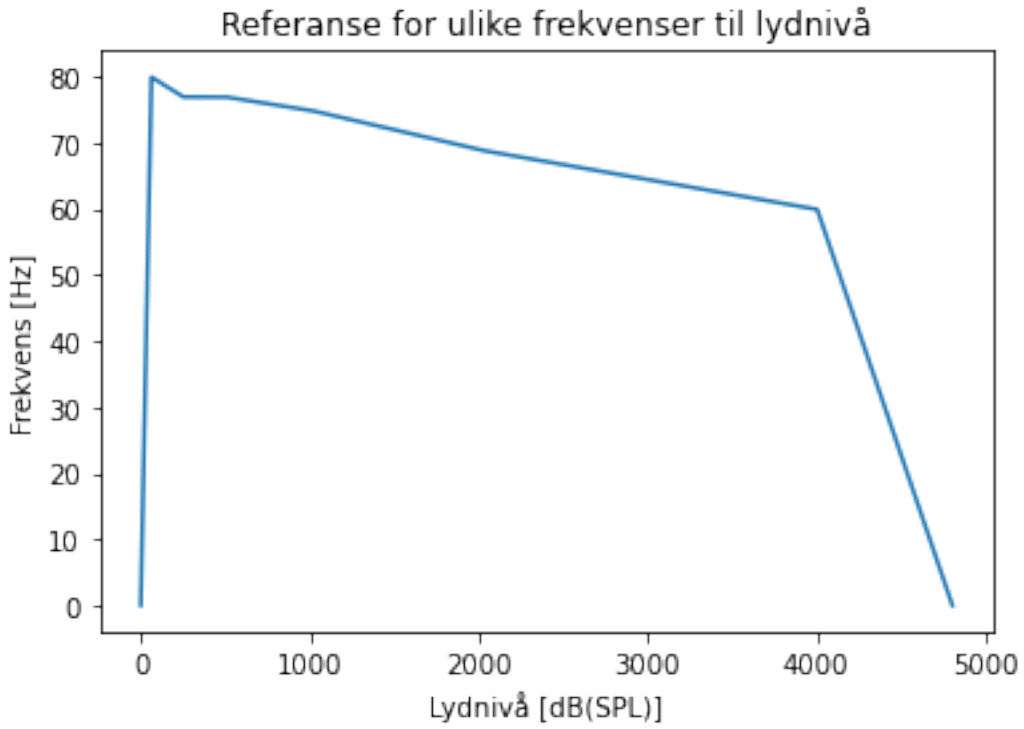
Videre trenger vi å lage en referanse array med hva lydnivået burde være for ulike frekvenser.

[14]:
```
ref = np.zeros(N)
for i in range(len(dbSPL_ref)-1):
    tmp = np.linspace(dbSPL_ref[i], dbSPL_ref[i+1], len(freq[(freq_ref[i] <=␣
 ↪freq) & (freq < freq_ref[i+1])]))
    ref[(freq_ref[i] <= freq) & (freq < freq_ref[i+1])] = tmp
ref[(freq_ref[i+1] <= freq) & (freq < 4800)] = np.linspace(dbSPL_ref[i+1], 0,␣
 ↪len(freq[(freq_ref[i+1] <= freq) & (freq < 4800)]))
```

```
ref[freq <= 0] = np.flip(ref[(0 <= freq)])
```

[15]:
```
plt.plot(freq[(0 <= freq) & (freq <= 4800)], ref[(0 <= freq) & (freq <= 4800)])
plt.title("Referanse for ulike frekvenser til lydnivå")
plt.xlabel("Lydnivå [dB(SPL)]")
plt.ylabel("Frekvens [Hz]")
```

[15]:
```
Text(0, 0.5, 'Frekvens [Hz]')
```



Over kan du se hvordan vår referanse ser ut, hvor vi har gjort et par antagelser, at lydnivået er null ved frekvensen 0 og at den er 0 ved frekvensen 4800.

Videre må vi finne STL gitt ved:

$$STL = 10 \log\left(\frac{1}{\tau}\right) \tag{3}$$

hvor τ er forholdet mellom intensiteten som treffer veggen og den som går igjennom om vist her. Videre antar vi at kun en fjerdedel av intensiteten til lydkilden treffer veggen (kun interessert i den ene veggen av fire).

[16]:
```
I_0 = 1e-12 #W/m^2

def I(dB):
    return I_0*10**(dB/10)
```

Så for funksjonen over sender vi inn lydnivået og får ut itensiteten ved hjelp av SPL. Men hvordan kalkulerer vi dB på hver av rommene. Her kommer funksjonen vår hvor vi bruker impedans inn igjen. Først kan vi enkelt kalkulere lydnivået i resturantern, uten å ta hensyn til refleksjon av veggene. For å finne lydnivået i rommet vårt bruker vi den transmiterte lyden fra resturanten, men vi legger til et trinn til ved den første metoden vår. Lyden fra resturanten vil nemlig treffe veggen mer enn en gang. Så om vi bestemmer hvor bredt rommet er (vi antar altså at all lydvektoren står normalt på veggen) kan vi bruke at lyd avtar inverst proposjonalt med avstand r^2 fra kilden. Slik at:

$$I_{tot} = I_1 + \sum_{i=1}^{\infty} \frac{I_1}{i(2d)^2} \tag{4}$$

Hvor d er bredden på rommet, I_1 er intensiteten i resturanten. Her blir det også antatt at intensiteten som reflekterer er den samme som treffer, som er en liten forenkling.

[17]:
```
def I_tot(I, d, tol = 1e-10, iters = 1000):
    s = I
    for i in range(1, iters):
        s += I/(i*(2*d)**2)
    return s
```

Velger at $d = 10m$

[18]:
```
d = 10 #m
```

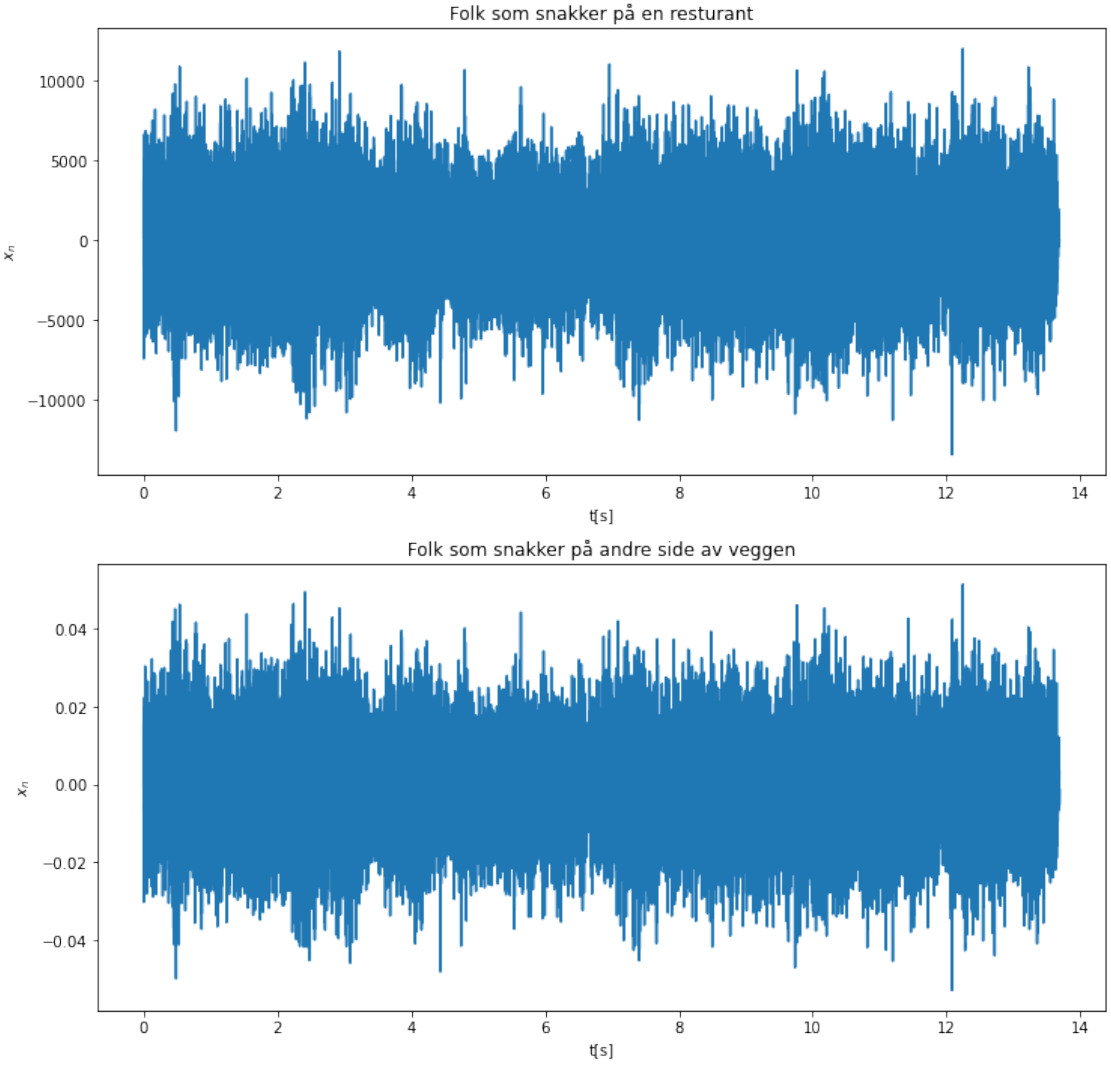
Beregner det nye frekvensdomenet til rommet vårt.

[19]:
```
dB = X_k/max(X_k)*ref

I_1 = I(dB)/4 #deler på fire siden vi ser på kun en vegg
I_tot_1 = I_tot(I_1, d)
_, I_2 = wall(I_tot_1, Air, Brick)


tau = I_2/I_1
STL = 10*np.log(1/tau)
new_dB = dB - STL

new_X_k = new_dB/max(new_dB)*X_k
```

Veggen reflekterer R=99.98% og tar imot T=0.02%

[20]:
```
people_talking_other_side = np.real(np.fft.ifft(new_X_k))
plt.figure(figsize=(12,12))
plt.subplot(2, 1, 1)
plt.plot(t, people_talking)
plt.title("Folk som snakker på en resturant")
plt.xlabel("t[s]")
plt.ylabel("$x_n$")
plt.subplot(2, 1, 2)
```

```
plt.plot(t, people_talking_other_side)
plt.title("Folk som snakker på andre side av veggen")
plt.xlabel("t[s]")
plt.ylabel("$x_n$")
wavfile.write(f"people_talking_other_side.wav", fs, people_talking_other_side.
 ↪astype("float32"))
```

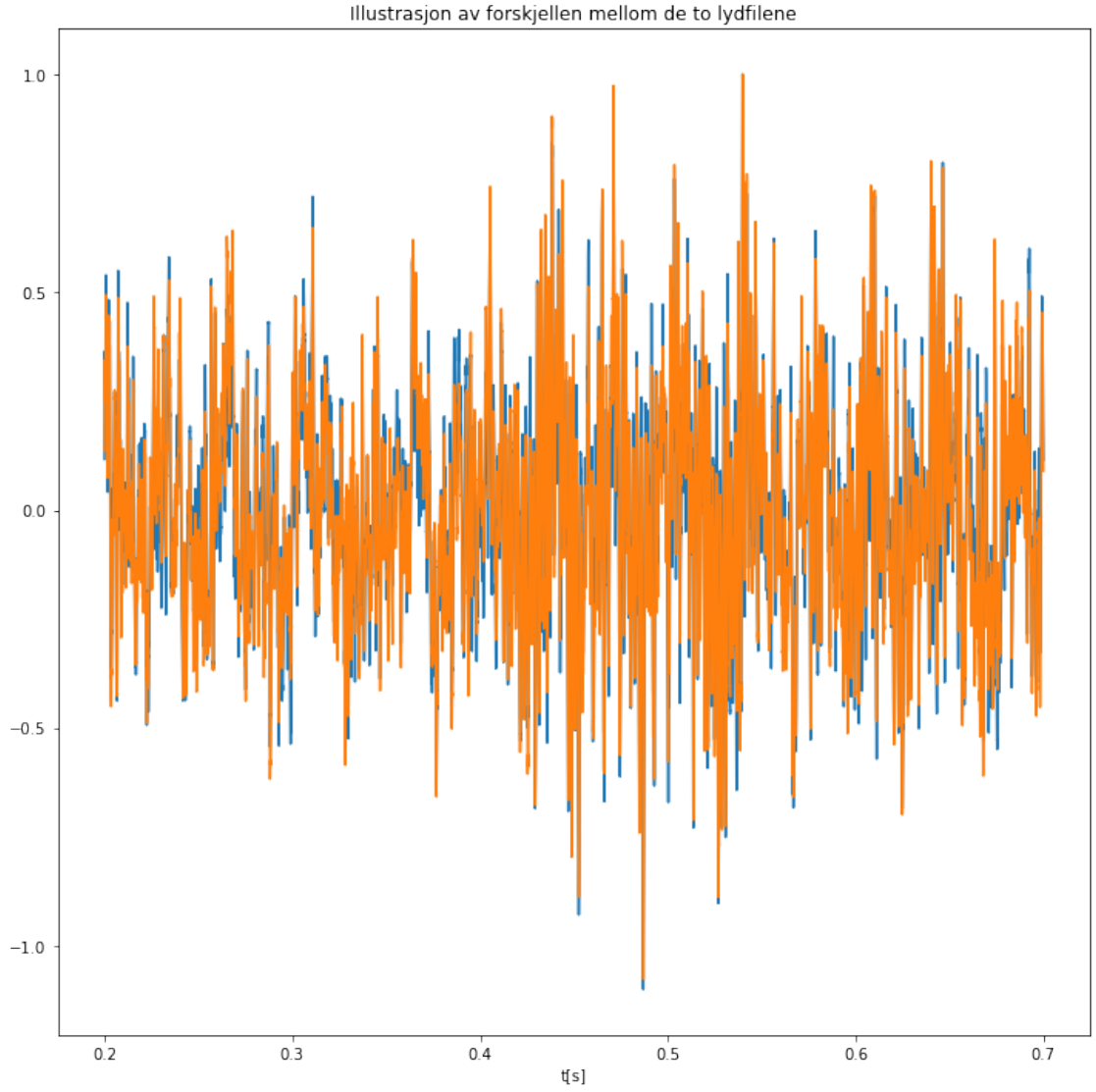


Her er det to viktige observasjoner. Den ene er den store forskjellen i amplitude, og den andre er at de ikke er nøyaktige kopier av hverande (selv om man ser bort ifra amplitude). Som blir tydelig i figuren under:

[21]:
```
plt.figure(figsize=(12,12))
plt.plot(t[(0.2 < t) & (t < 0.7)], people_talking[(0.2 < t) & (t < 0.7)]/
 ↪max(people_talking[(0.2 < t) & (t < 0.7)]))
```

```
plt.plot(t[(0.2 < t) & (t < 0.7)], people_talking_other_side[(0.2 < t) & (t < 0.
↪7)]/max(people_talking_other_side[(0.2 < t) & (t < 0.7)]))
plt.title("Illustrasjon av forskjellen mellom de to lydfilene")
plt.xlabel("t[s]")
```

[21]: `Text(0.5, 0, 't[s]')`



Videre forbedringer til moddelen vår kunne vært å ta hensyn til tykkelsen på veggen, ved å ha et bedre uttryk for avtagelsen av intensiteten inne i veggen. Lyd er veldig komplisert og derfor vanskelig å modulere ordentilig uten en drøss med antagelser.

1.3 Konklusjon

For å konkludere, det er mulig å simmulere hva lyden høres ut som på andre siden av veggen til en resturant. Men det lar seg ikke gjøres ved kun en enkelt hvilken som helst lydfil. Man trenger en referanse for å kunne konvertere over til $dB(SPL)$ som er nødvendig for hvordan modellen vår er bygget opp. For å gjøre det enklere for oss selv antar vi at vi er i et diffust lyd felt, at vi befinner oss i en gjennomsnittlig resturant, rommet er $d = 10m$ bredt og retningen til lyden står normalt på veggen.