
DSP mini projekt

ESD5 Digital Signal Processering 2024

Mini Projekt

Christian Lykke Jørgensen og Frederik Sofus Hansen

Copyright © Aalborg Universitet 2024

Simuleringer og udregninger er lavet i MatLab og Python med Sympy, SciPy Numpy og Matplotlib Bibliotekerne. Rapporten er formateret i L^AT_EX og skrevet i Overleaf



Electronik og Systemdesign

Aalborg Universitet

<http://www.aau.dk>

AALBORG UNIVERSITY

STUDENT REPORT

Titel:

DSP mini projekt

Tema:

Design af FIR filter med frekvens sample metoden.

Projektperiode:

Efterårssemestret 2024

Projektgruppe:

N/A

Deltager(e):

Christian Lykke Jørgensen
Frederik Sofus Hansen

Vejleder(e):

Peter Koch

Oplagstal: 1

Sidetal: 22

Afleveringsdato:

2 December 2024

Abstract:

I denne rapport vil:

- Frekvens sample metoden gennemgåes og beskrives matematisk/theoretisk.
- Metoden vil benyttes til at designe et LP filter med med 3dB knæk ved 1Khz.
- Der eksperimenteres med forskellige frekvens-sample værdier og deres effekt på transistions båndet.
- Sample-frekvens FIR filteret sammenlignes med vindue metoden.
- Sample-frekvens filteret effekt på et støjet signal blive simuleret.

Contents

1 Opgave 1	2
1.1 Lavpraktisk gennemgang af frekvenssamplingmetoden til design af et FIR filter:	2
1.1.1 Hvad er et FIR filter	2
1.1.2 Sample Frekvens Metoden	2
1.1.3 Sampling af frekvensresponset	3
1.1.4 Konstruktion af impulssvaret	3
2 Opgave 2	5
3 Opgave 3	9
4 Opgave 4	13
5 Punkt 5	18
A Appendix A - Making a Filter by the Book	22

1 | Opgave 1

"Der ønskes foretaget en teoretisk/matematisk gennemgang og beskrivelse af frekvens-sampling metoden til design af FIR-filtre"

Et FIR-filter baseret på frekvenssamplingmetoden adskiller sig fra traditionelle FIR-filtre ved, at man i stedet for at tage udgangspunkt i et ideelt filter i det kontinuerte frekvensdomæne, benytter frekvenssamples fra frekvensdomænet og bruger dem til at konstruere filterets impulsrespons. Generelt er metoden god til at designe filtre med meget bestemte frekvenskarakteristika, som ellers kan være svære at opnå igennem vinkelduesmetoden.

1.1 Lavpraktisk gennemgang af frekvenssamplingmetoden til design af et FIR filter:

1.1.1 Hvad er et FIR filter

Et FIR-filter, kort for finite impulse response, er som navnet hentyder et filter, som er brugbart, når impulsresponset af et filter ikke er uendeligt langt.

Implementationen for et diskret FIR filter ser således ud:

$$y[n] = \sum_{k=0}^{N-1} h[k] \cdot x[n-k] \quad (1.1)$$

hvor $h[]$ er et sample tog der beskriver impulsresponsen af filteret. $h[]$ bliver også ofte kaldt FIR-filterets "koefficienter" da de ganges på signalet i den diskrete implementation set i formel 1.1

1.1.2 Sample Frekvens Metoden

Ved denne design metode af et FIR-filter starter man med at beskrive det ønskede ideelle frekvensspektrum efter filteret. Vil man f.eks. have et lavpas filter ville funktionen kunne beskrives således:

$$H(e^{j\omega}) = \begin{cases} 1 & k < \omega_0 \\ 0 & k \geq \omega_0 \end{cases} \quad (1.2)$$

Eller hvis der ønskes et transistions bånd:

$$H(e^{j\omega}) = \begin{cases} 1 & k < \tau_0 \\ \frac{\omega}{\tau_0 - \tau_1} - \tau_0 & \tau_0 \leq k < \tau_1 \\ 0 & k \geq \tau_1 \end{cases} \quad (1.3)$$

Her er τ_0 og τ_1 lig med k værdien for start og stop punktet af transistions båndet.
Nu skal vi sample denne frekvens for dermed at kunne skabe det diskrete frekvenspektrum $H[k]$. Her skal vi dog huske at når vi går fra S planet i continuer tid til Z planet i diskret tid bliver frekvensspektrummet periodisk.

$$H(e^{j\omega}) = H(z)|_{z=e^{j\omega}} \quad (1.4)$$

Derfor skal man sørge for at ens filters sampling frekvens er $F_s > 2 \cdot \omega_0$ eller $F_s > 2 \cdot \tau_1$ i transisions bånds tilfældet. Da men ellers vil få aliasing, dette følger også shannon-nyquist theoremet.

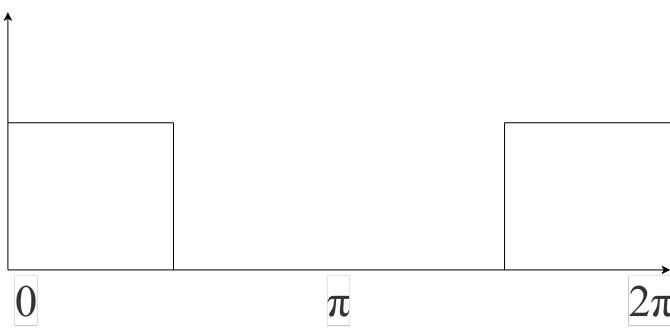


Figure 1.1: Et ideelt diskret lavpasfilter.

1.1.3 Sampling af frekvensresponset

Med det ønskede frekvensrespons på plads, skal antallet af samples N vælges. Ligesom med alt andet, vil at større antal samples resultere i bedre oplosning og større sikkerhed i det målte. Vi definerer det aktive sample som "k" og det totale antal samples som "N". Alle samples er selvfølgelig jævnt fordelt langs området fra $-\pi$ til π . Den matematiske forklaring er givet ved:

$$H[k] = H(e^{j\cdot\omega_k}) \quad (1.5)$$

Hvor:

$$\omega_k = \frac{2 \cdot \pi \cdot k}{N} \text{ og } k = 0, 1, 2, 3, \dots, N - 1 \quad (1.6)$$

Bemærk at $H[k]$ er frekvensresponset til det n-te sample og ω_k er de samplede frekvenser. Samplesne repræsenterer filterets målte respons ved bestemte frekvenser, og definerer derved hvordan filteret vil opføre sig i frekvensdomænet.

1.1.4 Konstruktion af impulssvaret

Efter at have samplet en given mængde værdier i frekvensdomænet, kan impulsresponsen konstrueres. Udbyttet af impulssvaret er filterets koefficienter i tidsdomænet. Impulsresponsen findes vha. den inverse DFT givet i formel 1.8 for n=0 til n=N-1. Den

inverse DFT har til formål at transformere alle samples fra frekvensdomænet, tilbage i tidsdomænet, så man kan få de tilhørende filterkoefficienter ud i sidste ende.

$$h[n] = \frac{1}{N} \sum_{k=0}^{N-1} H[k] \cdot e^{j \cdot \frac{2\pi}{N} \cdot k \cdot n} \quad (1.7)$$

Der ønskes dog ofte en lineær faserespons i filteret, da det både bliver lidt nemmere at regne med (ingen imaginær del), men mest fordi det skaber et filter der ikke forskyder eller forvrænger signalets fase. Et system med symmetrisk impulsrespons vil altid have linear fase da enhver fase forskydelse et sample kunne give ville blive modgjort af sit symmetrisk modsatte sample.

Dette kan gøres ved at forskyde alle samples med $\alpha = (N - 1)/2$ samples dermed omskrives DFT formlen til:

$$h[n] = \frac{1}{N} \sum_{k=0}^{N-1} H[k] \cdot e^{j \cdot \frac{2\pi}{N} \cdot k \cdot (n - \alpha)} \quad (1.8)$$

Da vi ved at vores signal er symmetrisk (altså at de første N-1 samples er lig de sidste N-1 samples) og at fasen nu er ligegyldig da den er spejlet omkring 0 kan vi gøre dette computationelt nemmere for os selv og skrive formlen om til.

$$h[n] = \frac{1}{N} \sum_{k=1}^{2/N-1} (2 \cdot |H[k]| \cdot \cos[2\pi \cdot k \cdot (n - \alpha)/N]) + H[0] \quad (1.9)$$

$H[0]$ kan summeres på til sidst da $\cos(0) = 1$ og skal ikke ganges med 2 da vi antager at mængden af coefficenter er ulige.

For filtre med et lige antal N kan denne formel bruges:

$$h[n] = \frac{1}{N} \cdot \sum_{k=1}^{(N-1)/2} 2 \cdot |H[k]| \cdot \cos[2\pi \cdot k \cdot (n - \alpha)/N] + 2 \cdot H[0] \quad (1.10)$$

2 | Opgave 2

"Metoden benyttes herefter til konstruktion af et LP-filter, som bedst muligt overholder de designspecifikationer for filteret, som blev introduceret ifm. opgaveregningen i kursets 4. forelæsning (opg. 1c)."

For at kunne designe et LP filter, skal det først specificeres. Specifikationerne givet i opgave 1c fra kursusgang 4 er:

- Samplefrekvens på 8 kHz
- 3dB knæk ved 1 kHz
- 0dB gain ved DC
- Mindst -1dB forstærkning ved 750 Hz
- Maksimalt -10dB forstærkning ved 1500 Hz

Lidt mere formaliseret kan det skrives som:

- Nyquistfrekvens: $f_N = \frac{f_s}{2} == 4\text{kHz}$
- Frekvensdomænet strækker sig over $[0, \pi]$, hvor π svarer til f_N , altså 4 kHz.
- 3dB knæk ved 1 kHz ($0.25f_N$ eller $\omega = 0.25\pi$)
- 0dB ved DC ($\omega = 0$)
- Mindst -1dB ved 0.750 kHz ($\omega = 0.1875\pi$)
- Maksimalt -10dB ved 1.500 kHz ($\omega = 0.375\pi$)
- N = arbitraert tal, 91 vælges for at få et skarpt peak i impulsresponsen sammen med en nogenlunde opløsning. Tallet skal helst være ulige, for at få en skarp sinc funktion, samt for optimal afbilledning af responset.
- Frekvensresponset er afgjort ved: $H(e^{j*\omega}) = \begin{cases} 1 & \text{if } \omega_k \leq 0.25\pi \\ 0 & \text{if } \omega_k > 0.375\pi \end{cases}$, og skal aftage gradvist derimellem.

Næste step er helt kort at skabe filteret i MATLAB eller lignende. Det gøres ved først at indsætte de kendte værdier som variable, bl.a. med et array som indeholder de frekvenser der er relevante for filterdesignet. -40 dB ved nyquistfrekvensen er hevet lidt ud af luften, da det er noget som minder om -40 dB pr dekade, hvis man antager ≈ 0 dB ved 400 Hz. Resten af værdierne kommer fra opgaveformuleringen.

```

1 % Parametre
2 N = 91;                                % Filterlaengde
3 fs = 8000;                               % Samplefrekvens i Hz
4 frequencies = [0, 750, 1000, 1500, fs/2]; % Frekvenser i Hz
5 gains_dB = [0, -1, -3, -10, -40];        % Gain i dB (krav)
6
7 % Konverter gain fra dB til lineaer skala
8 gains_linear = 10.^^(gains_dB / 20);

```

Derefter kan frekvenserne normeres, så de passer mellem 0 og 1 (Nyquist-normalisering), hvor 1 svarer til nyquistfrekvensen, hvilket er nødvendigt for at kunne bruge smarte funktioner, som FIR2 i MATLAB.

```

1 % Normer frekvenserne til [0, 1] (Nyquist-normalisering)
2 f_norm = frequencies / (fs / 2);

```

Efter det er der ikke så meget andet at gøre, end at sætte MATLAB til at lave matematik, før til sidst at plotte de ønskede figurer:

```

1 % Design impulssvaret med 'fir2'
2 h = fir2(N-1, f_norm, gains_linear);
3
4 % Beregn frekvensrespons til visualisering
5 [H, omega] = freqz(h, 1, 1024, 'whole', fs);
6
7 % Beregn frekvensrespons i dB
8 H_dB = 20 * log10(abs(H));
9 % Plot impulsresponsen
10 % Plot amplituderespons (lineaer skala)
11 % Plot amplituderespons (dB skala)
12 % Plot faserespons

```

De plottede grafer kan ses i de følgende figurer, 2.1, 2.2, 2.3, 2.4 og 2.5. Husk at oplosningen lige nu er ret god, så filteret passer meget godt med de ønskede værdier.

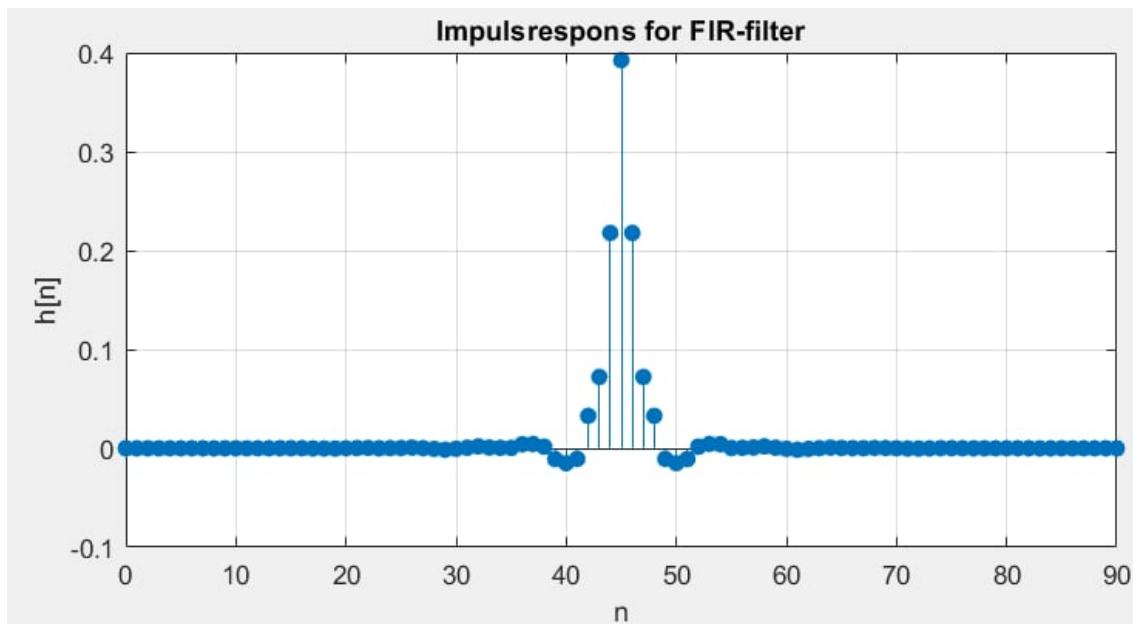


Figure 2.1: Impulsrespons ved 91 samples.

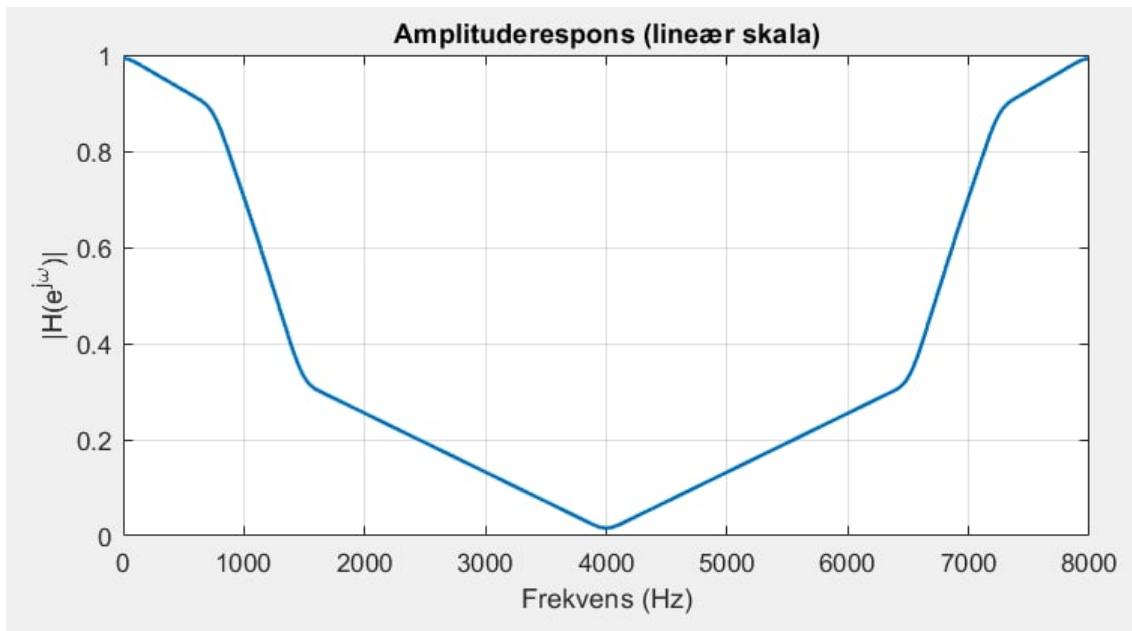


Figure 2.2: Lineær amplituderespons (det ideelle filter).

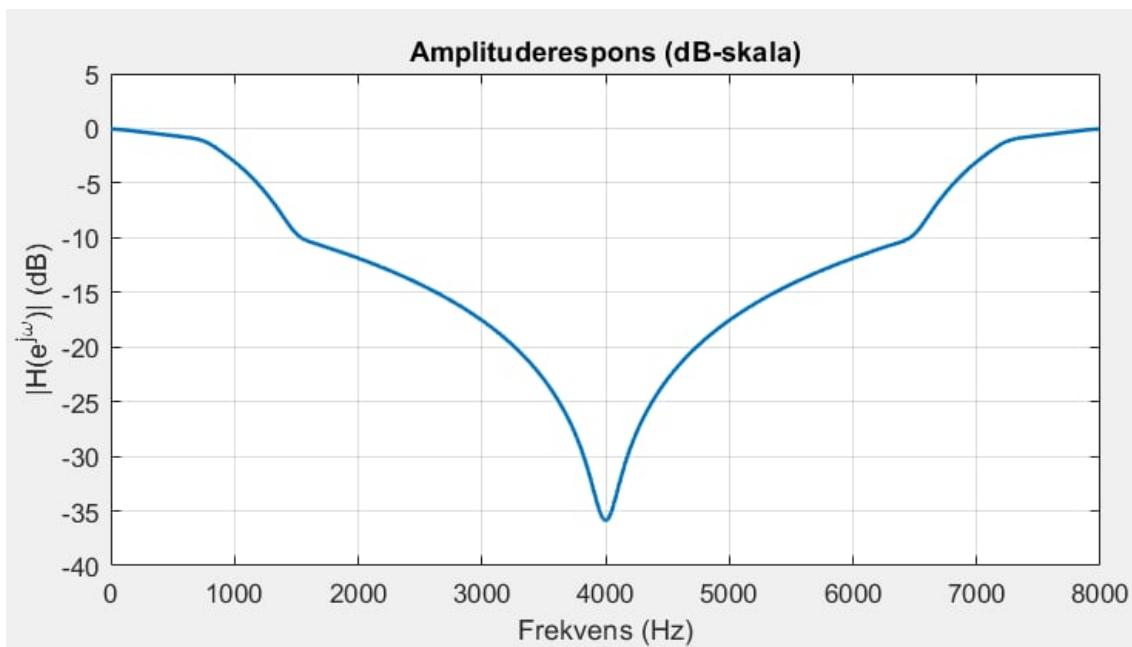


Figure 2.3: Amplituderespons ved 91 samples.

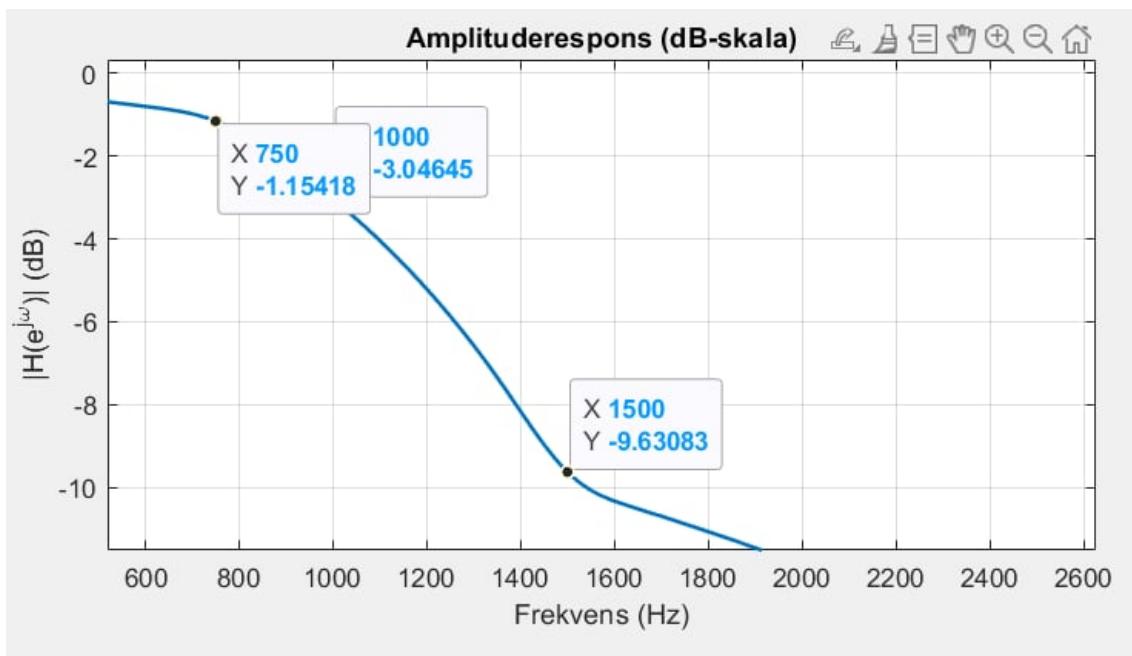


Figure 2.4: Amplituderesponse ved 91 samples, zoomet ind omkring de interessante værdier.

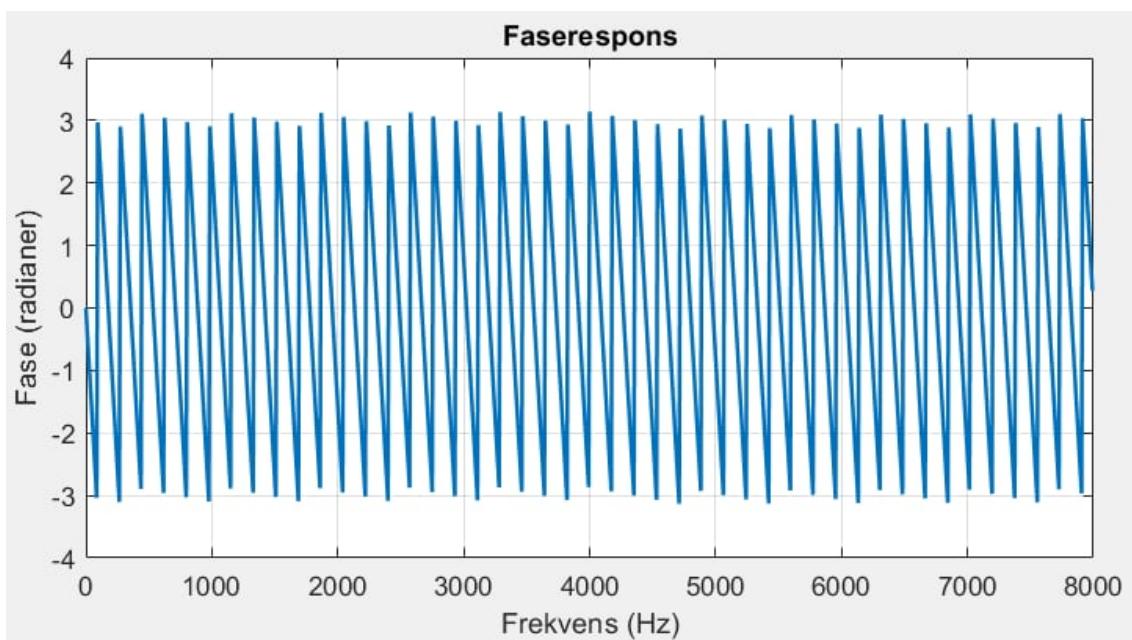


Figure 2.5: Faserespons ved 91 samples.

3 | Opgave 3

"Der eksperimenteres med og dokumenteres forskellige værdier af antal frekvens-samples i a) intervallet 0 til 2π , og b) transitionsbåndet mellem pas- og stop-bånd. Specielt pkt. b er interessant at undersøge."

For at kunne danne sig et indtryk af samplemængdens indflydelse, kan det være en fordel at kigge i I. Feachor's afsnit omhandlende frekvenssampling. Deri er samme figur som vist i figur 3.1, hvori man kan se indflydelsen, som antal samples i frekvensbåndet kan have. Des flere samples man har, des nemmere bliver det at beskrive pasbånd, transitionsbånd og stopbånd.

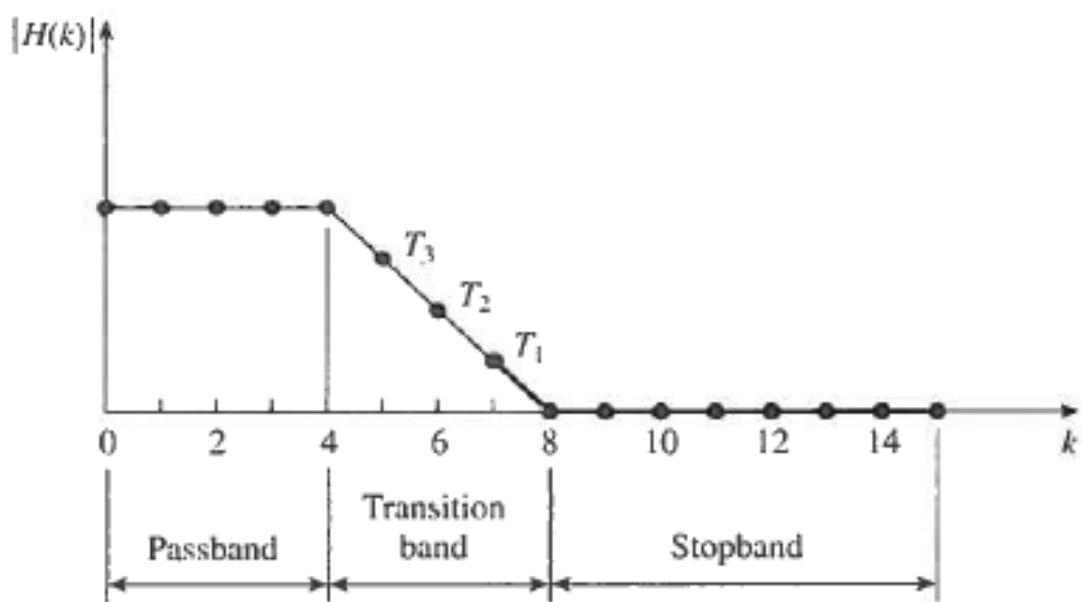


Figure 3.1: I. Feachors figur, som viser tydeligt, hvor mængden af samples i transitionsbåndet vil opløsningsgraden for beskrivelsen.

I de følgende figurer er filteret konstrueret på samme vis, som i kapitel 2. Eneste forandring er mængden af samples. Figur 3.2 til figur 3.6 er alle med $N=9$, og resultatet minder svagt om det vist i figur 2.4 til 2.5 med $N=91$.

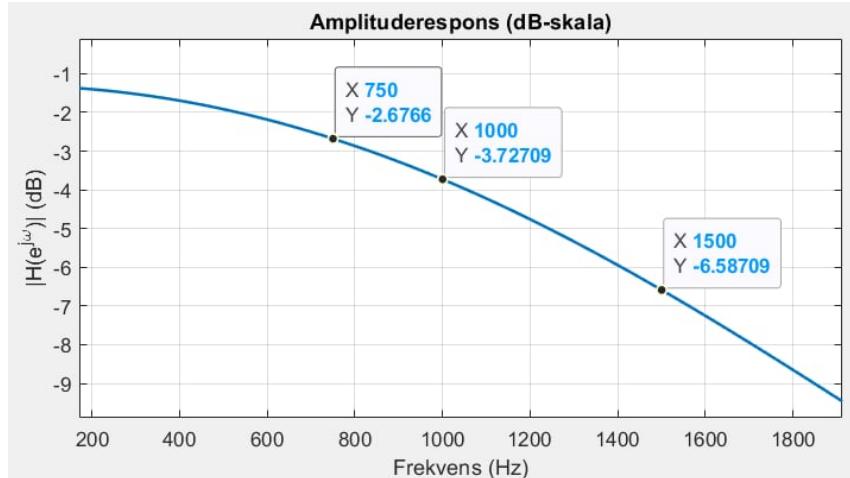


Figure 3.2: Amplituderespons ved N=9. Selvom filteret overholder kravene, er det stadig langt fra det ønskede.

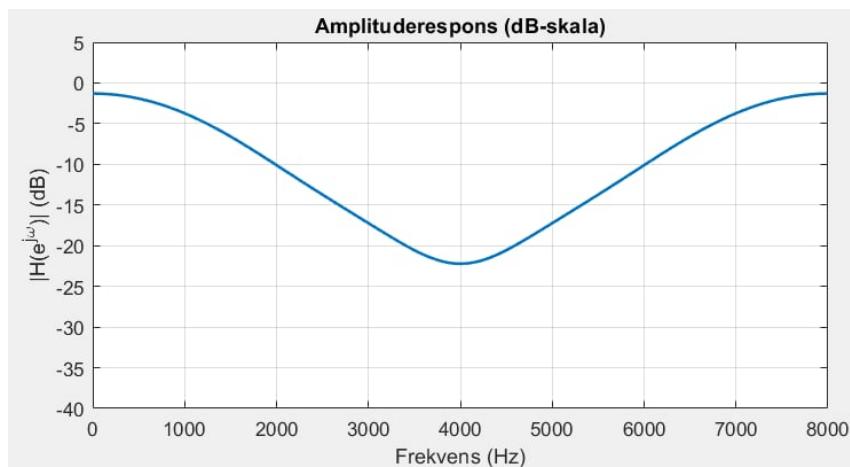


Figure 3.3: Amplituderespons ved N=9.

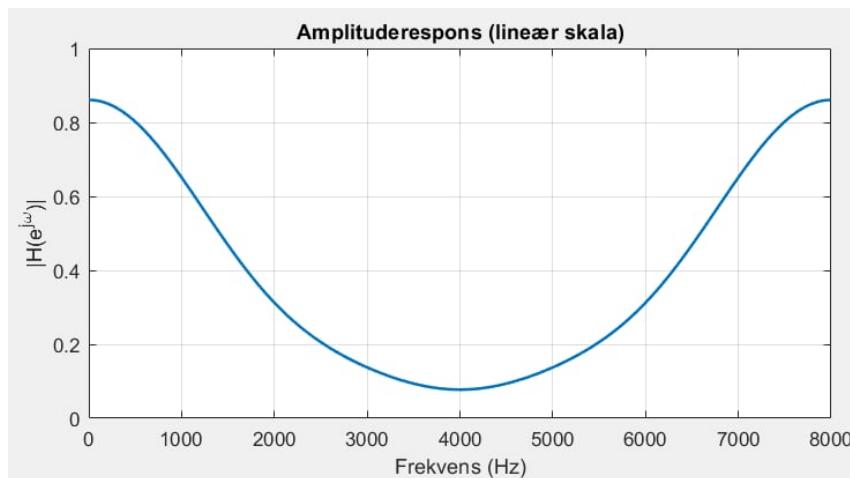


Figure 3.4: Lineær amplituderespons ved N=9.

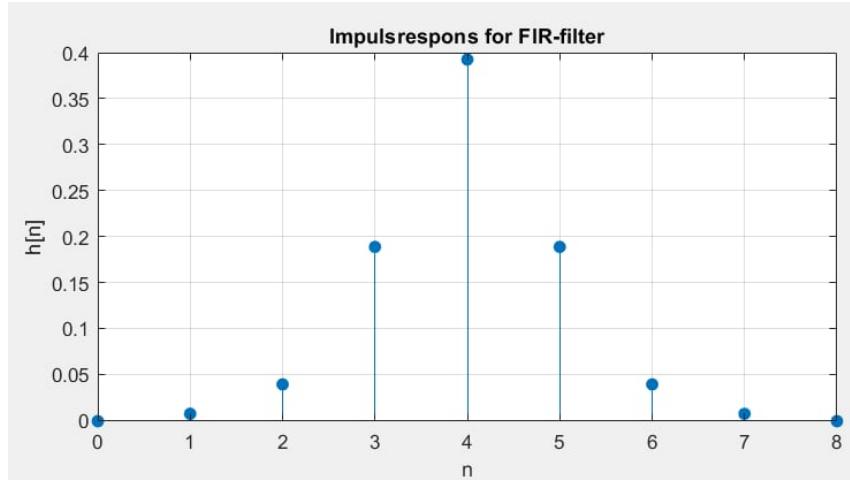


Figure 3.5: Impulsrespons ved N=9.

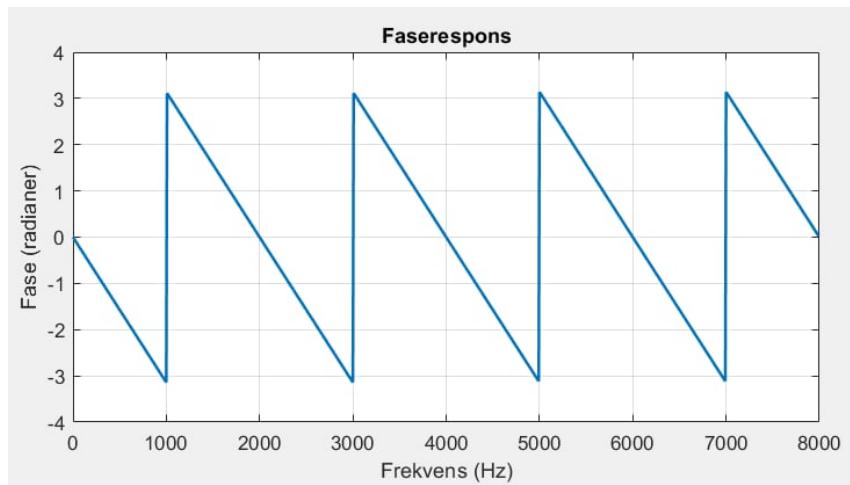


Figure 3.6: Faserespons ved N=9.

Hvis man vælger at forværre samplingen endnu mere, får man et endnu værre resultat. I figur 3.7 er filteret udført på ny, med hhv. N=3, 5 og 7. Det er ud fra dette tydeligt at konkludere, at mængden af samples påvirker filterets ydeevne ret voldsomt, ihvertfald ved lave sampleværdier.

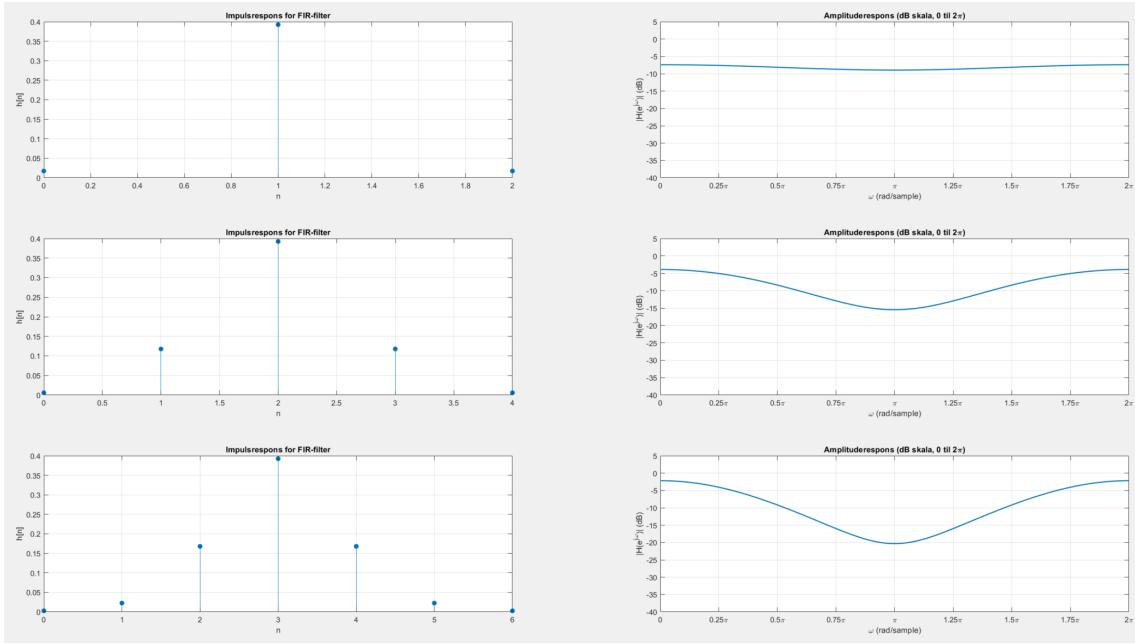


Figure 3.7: Som kan ses heri, bliver responsen i både amplitude og impuls plot markant forværret/utydelig, hvis man ikke har mere hhv. 3, 5 og 7 samples.

Som et ekstra forsøg for at finde et skæringspunkt mellem matematisk kompleksitet og ønsket amplituderesponsen, er amplituderesponsen for $N=9-35$ med sampleforøgelse på $N=2$ mellem hvert plot, vist i figur 3.8. Her er det tydeligt at se sampleforøgelse gør en markant større forskel ved lavere værdier, end høje. Et skæringspunkt for effektivitet kunne være omkring $N=25$, da amplituderesponsen herfor er meget tæt på den for $N=35$. Ikke desto mindre er der stadig lang vej til et resultat som vist ved $N=91$ i figur 2.3. Brugbarheden af dette, kan kun vurderes ud fra filterets endelige anvendelse.

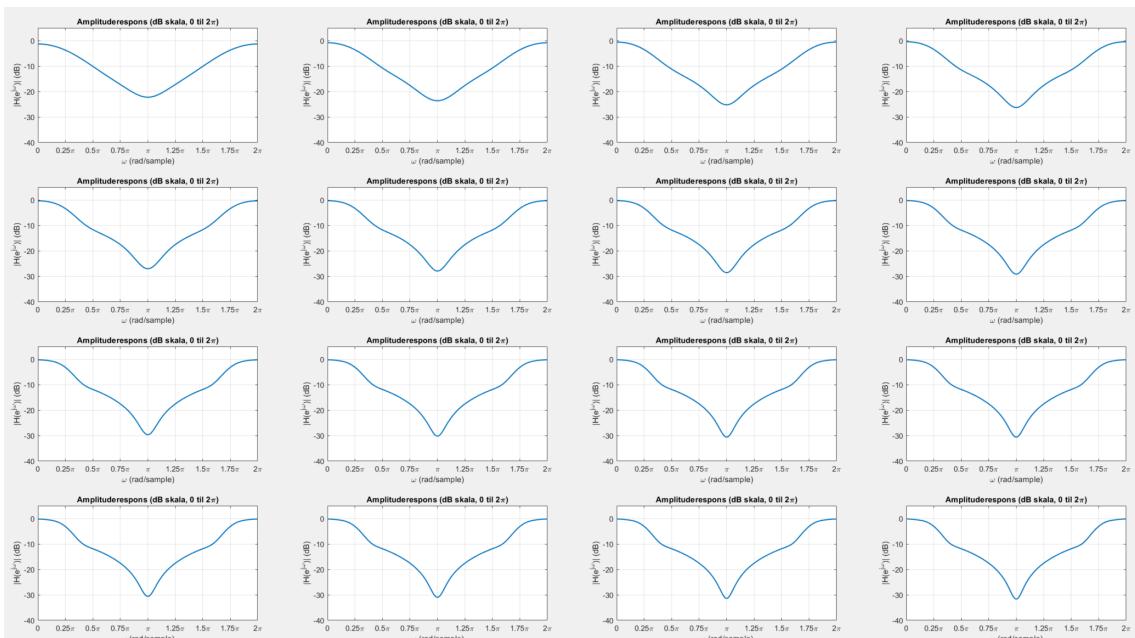


Figure 3.8: Oversigt over sampleforøgelse med $N=2$, fra $N=9$ til $N=35$. Øverste række indeholder $N=9, 11, 13, 15$, og så fremdeles gennem rækkerne nedad. Bemærk at fra $N=25$ er det begrænset hvor meget bedre filteret bliver.

4 | Opgave 4

"De fundne resultater sammenholdes med et filterdesign tilvejebragt vha. den ordinære vinduesmetode, hvor I naturligvis har frihed til at eksperimentere med alle de forskellige vinduesfunktioner og filter-ordener, som I måtte finde interessante/relevante."

For ikke at ignorere kompleksiteten ved højere filterordener, er samtlige vinduesfunktioner samt frekvenssamplingfilteret som udgangspunkt i dette afsnit med N=35, altså en orden på 35. På alle amplitudeplots er plottet RGB linjer, som skal vise de ønskede dæmpninger ved de givne frekvenser. Det betyder altså at krydsene vist med rød repræsenterer -1dB ved 750 Hz, grøn -3 dB ved 1000 Hz og blå maksimalt -10 dB ved 1500 Hz. Som kan ses i figur 4.1 er frekvenssamplingfilteret meget tæt på at være ideelt. Dette er gjort for nemmere at kunne evaluere både frekvenssamplingfilteret og samtlige vinduesfunktioners amplituderespons. I figur 4.1 er frekvenssamplingfilterets karakteristika vist.

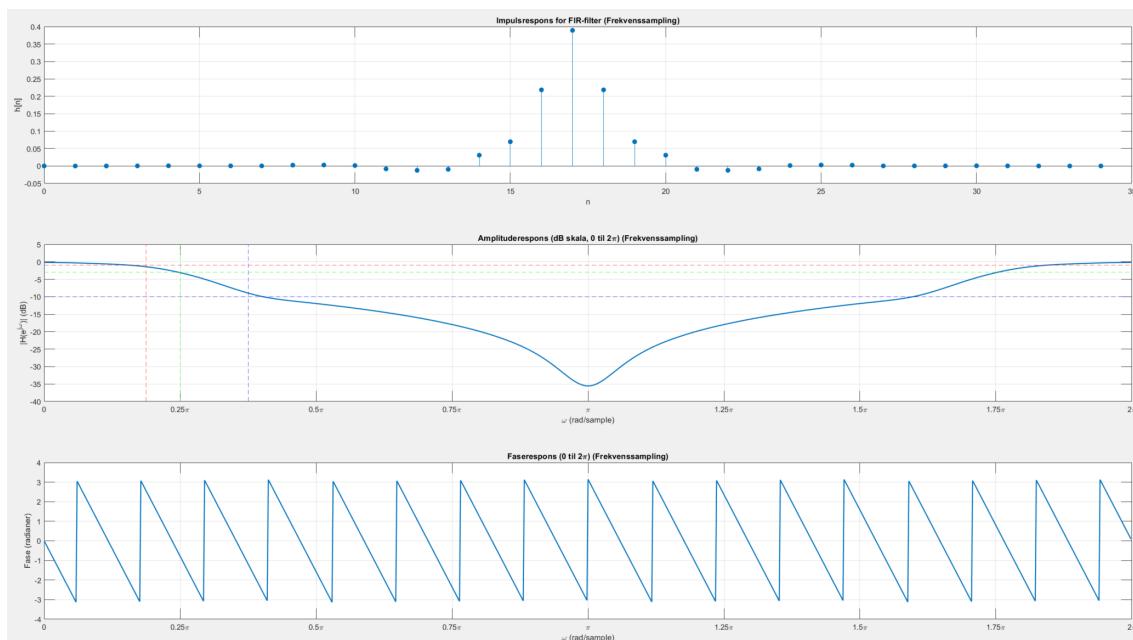


Figure 4.1: Sammenlignelig oversigt over impuls-, amplitude- og fase-respons for frekvenssamplingsfilteret ved N=35.

I figur 4.2 kan det rektangulære vindues karakteristika ses. Det skal bemærkes at den store mængde side lobes/ripples skyldes det rektangulære vindues brede gain, vist i impulsresponsen. Yderligere skal det bemærkes, at denne vinduesfunktion faktisk ikke overholder nogen af de ønskede dæmpninger. Ved 750 Hz er signalet tværtimod forstærket, og ved både 1000 og 1500 Hz er dæmpningen for kraftig. Altså er resultatet af denne vinduesfunktion ikke ønskværdig over frekvenssamplingmetoden.

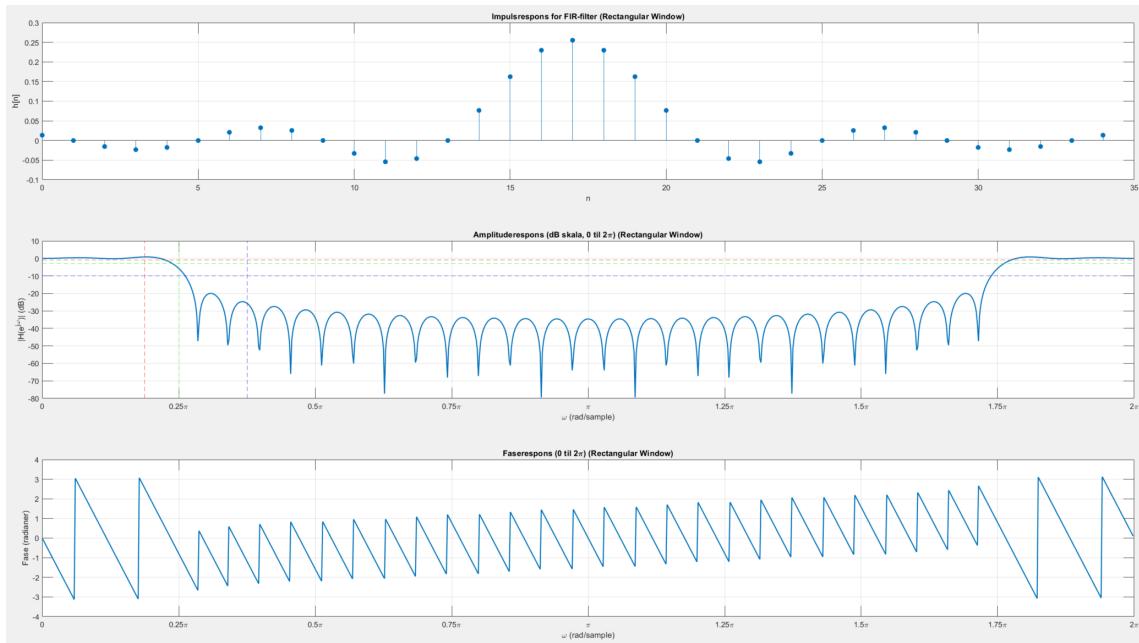


Figure 4.2: Oversigt over impuls-, amplitude- og faserespons for et rektangulært vindue ved $N=35$.

I figur 4.3 er et Kaiser vindue benyttet. Her er der igen kraftige ripples, som trækker dæmpningen ud af proportioner ved 1500 Hz. Denne gang er filteret dog tættere på de ønskede værdier ved hhv. 750 og 1000 Hz.

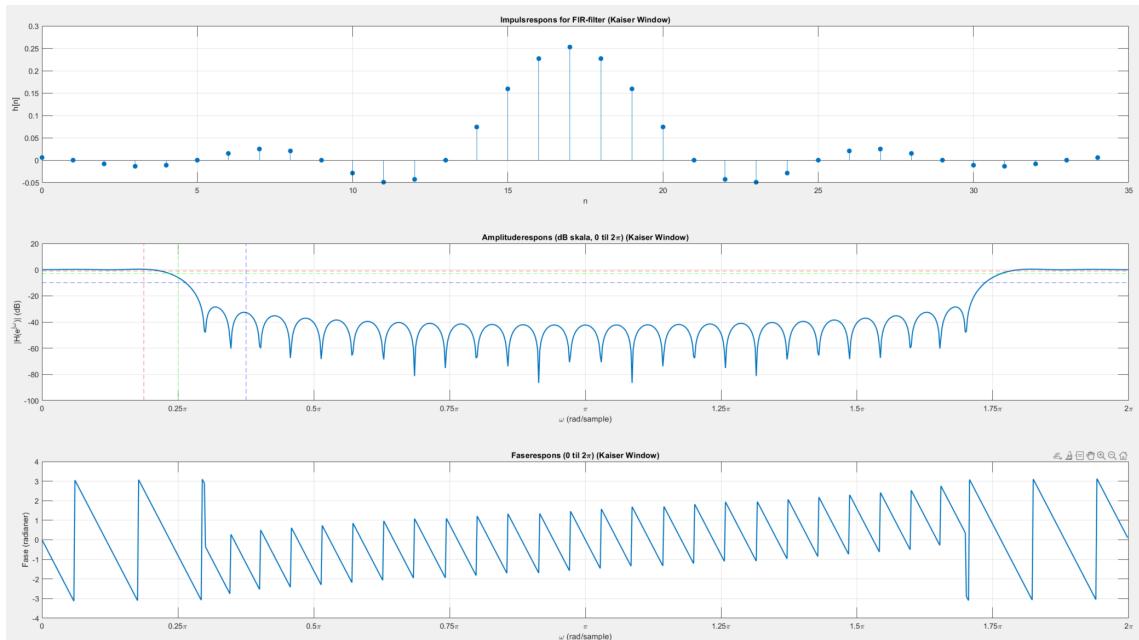


Figure 4.3: Oversigt over impuls-, amplitude- og faserespons for et Kaiser vindue ved $N=35$.

I figur 4.4 kan et Blackman vindue ses. Ligesom ved de forrige vinduer er der igen kraftige ripples til stede, til gengæld er der ikke noget grafisk synligt opsving i starten af vinduet. Yderligere skal det bemærkes at foruden lavpunkterne ved hvert ripple så dæmpes signalet gennemsnitligt med yderligere 20 dB fra 0.5π til π . Afsluttende er det værd at kigge på faseresponsen, da den er alt andet end kontinuert lineær. Til gengæld er faseresponsen stykvist lineær, i den forstand at omkring $\pm\pi$ er fasen lineær, men

omkring 0.375π bliver signalet ulineært. Dette passer dog meget godt overens, med det ønskede amplituderesponse, hvor dæmpningen er minimal frem til 0.25π .

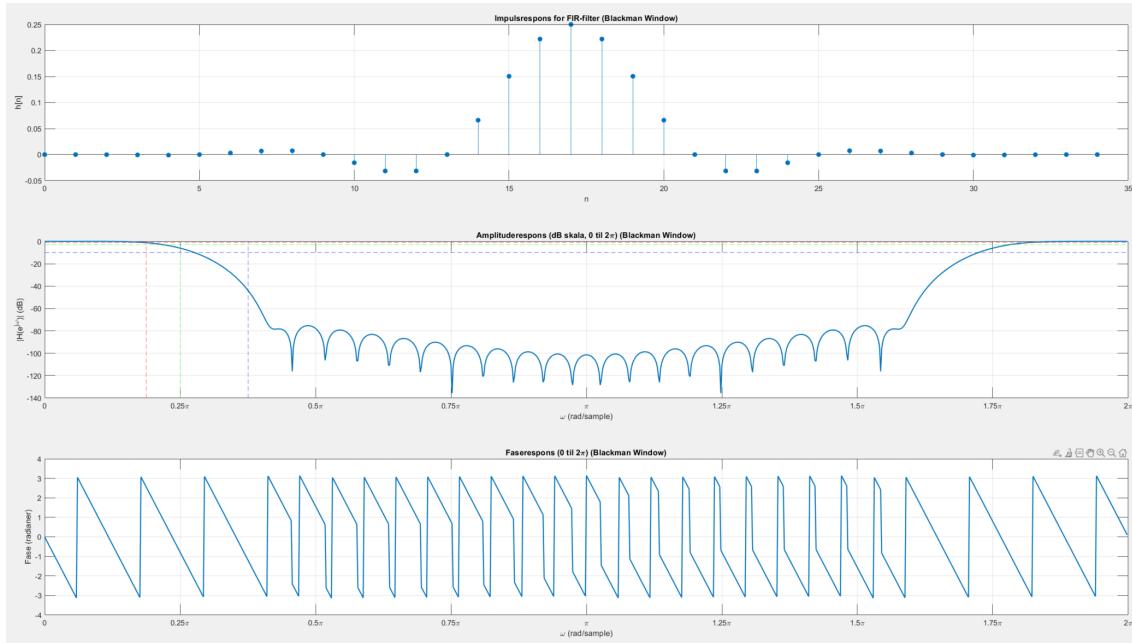


Figure 4.4: Oversigt over impuls-, amplitude- og fase-respons for et Blackman vindue ved $N=35$.

I figur 4.5 kan er det et Hanning vindue som er vist frem. Denne gang med et tilnærmelsesvist ønsket respons ved både 750 og 1000 Hz, men stadig ikke ideelt. Ligesom ved tidligere vinduesfunktioner er den relativt "smalle" main lobe skyld i, at signalet er dæmpet markant mere end de ønskede 10 dB ved 1500 Hz.

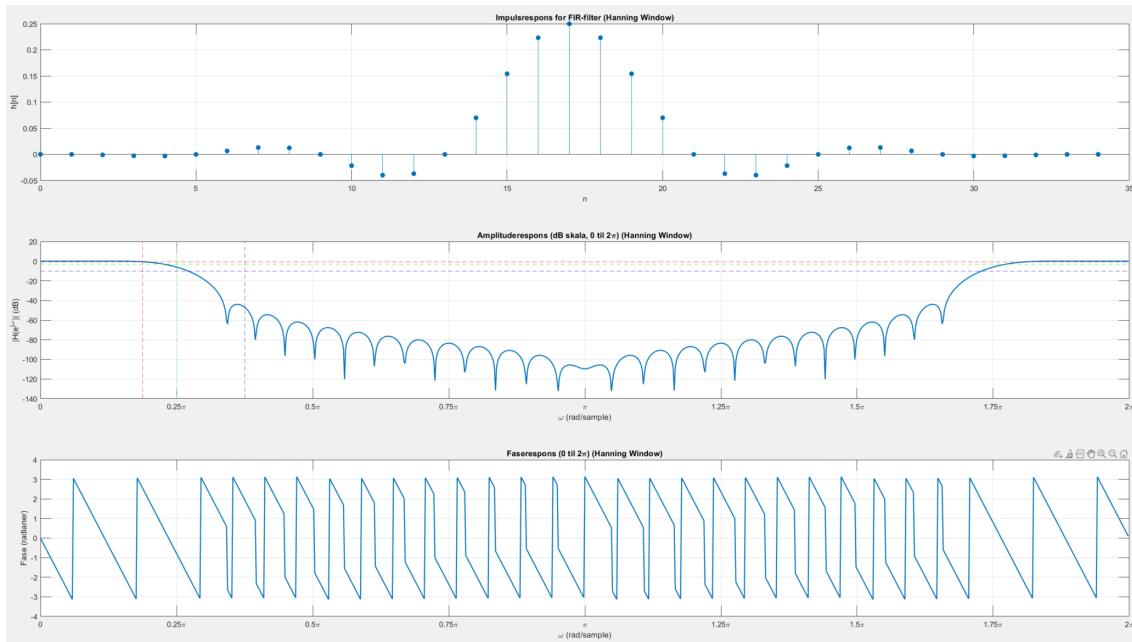


Figure 4.5: Oversigt over impuls-, amplitude- og fase-respons for et Hanning vindue ved $N=35$.

Ligesom ved Blackman vinduet bliver signalet kontinuert dæmpet mere og mere i stopbåndet, denne gang med ca 60 dB fra 0.375π til π . Denne gang er faseresponsen også langt fra at være kontinuert lineær. Ligesom før er responsen dog stykvist lineær. Hvis

man sammenholder faseresponsen med amplituderesponsen lige ovenover kan det ses, at ulineariteten begynder for alvor ved 2. side lobe.

I figur 4.6 benyttes et Hamming vindue. Modsat de forrige vinduer er der næsten ingen fortsat dæmpning til stede i stopbåndet, til gengæld er ca -40 dB allerede opnået, så afhængig af applikationen kan det være godt nok. Den bredere main lobe gør, at vinduet er tættere på den ønskede filtre end flere af de forrige vinduesmetode. Ikke desto mindre er det stadig langt fra det ønskede, som frekvenssamplingfilteret ellers kan opnå. Som ved forrige vinduer, er faseresponsen lineær i pasbåndet, men afvigende i stopbåndet.

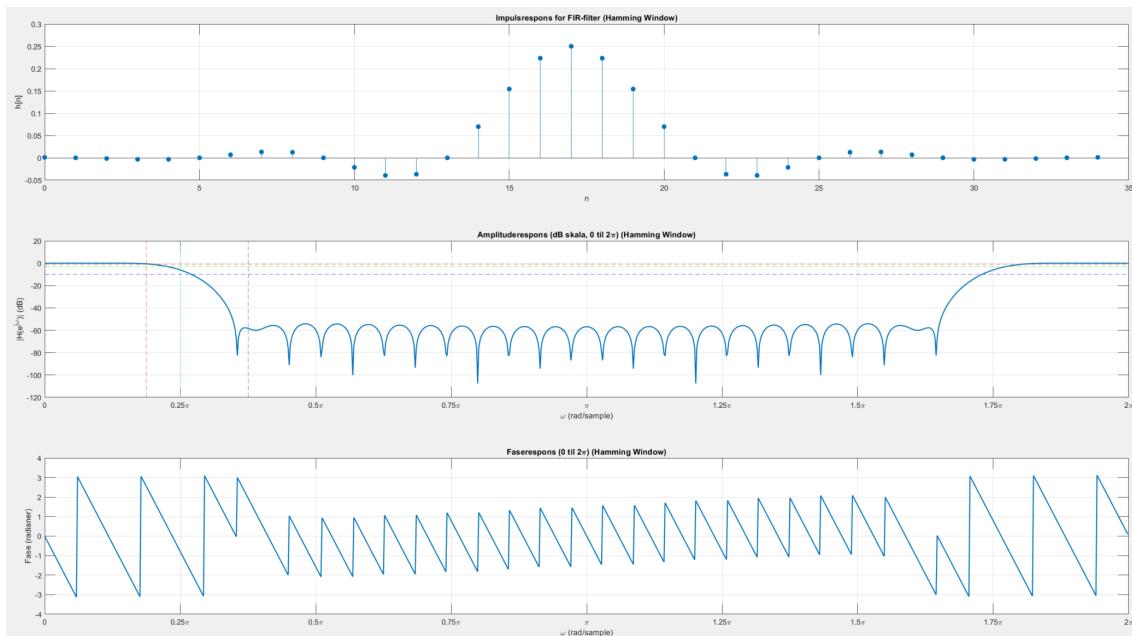


Figure 4.6: Oversigt over impuls-, amplitude- og fase-respons for et Hamming vindue ved $N=35$.

Figur 4.7 på næste side viser meget tydeligt, at hvis kravene til filteret er umulige at ændre, er det ene og alene frekvenssamplingmetoden, som kan resultere i den ønskede filtrering. Det ses tydeligt ved at bemærke positionen af krydsene med ønsket dæmpning for hver frekvens. Yderligere kan det bemærkes at frekvenssamplingmetoden har resulteret i et "blødt" filter, uden kraftige ripples/side lobes eller et ulinear faserespons nogen steder i frekvensspektret.

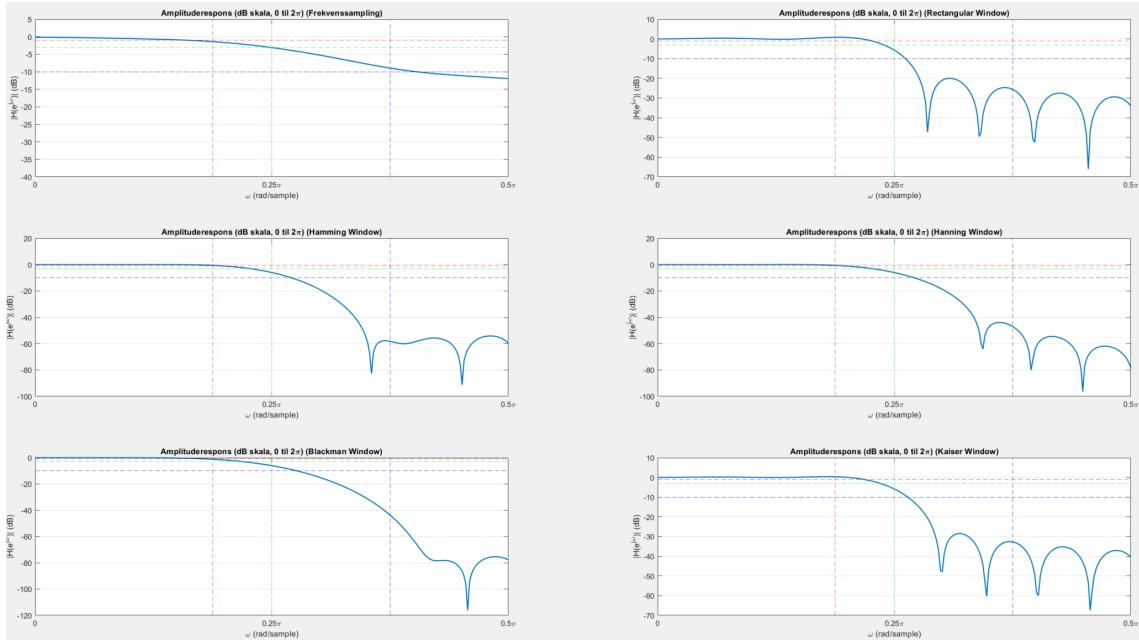


Figure 4.7: Zoomet version af amplituden ved $N=35$ for alle filtre/vinduer. Bemærk at det kun er frekvenssampling som reelt er tæt på de ønskede karakteristika.

I et forsøg på at gøre vinduesfunktionerne skarpere, er antallet af samples nu forøget med en faktor 5, så N nu er 175. Resultatet af dette er meget skarpere/smallere main lobes, som resulterer i næsten perfekt dæmpning ved 750 og 1000 Hz. Et smallere main lobe har dog den bivirkning, at ingen af vinduesfunktionerne er tæt på de maks 10 dB dæmpning ved 1500 Hz længere. De nærmeste er det rektangulære vindue og Hamming vinduet, som dæmper med ≈ 40 dB ved 1500 Hz.

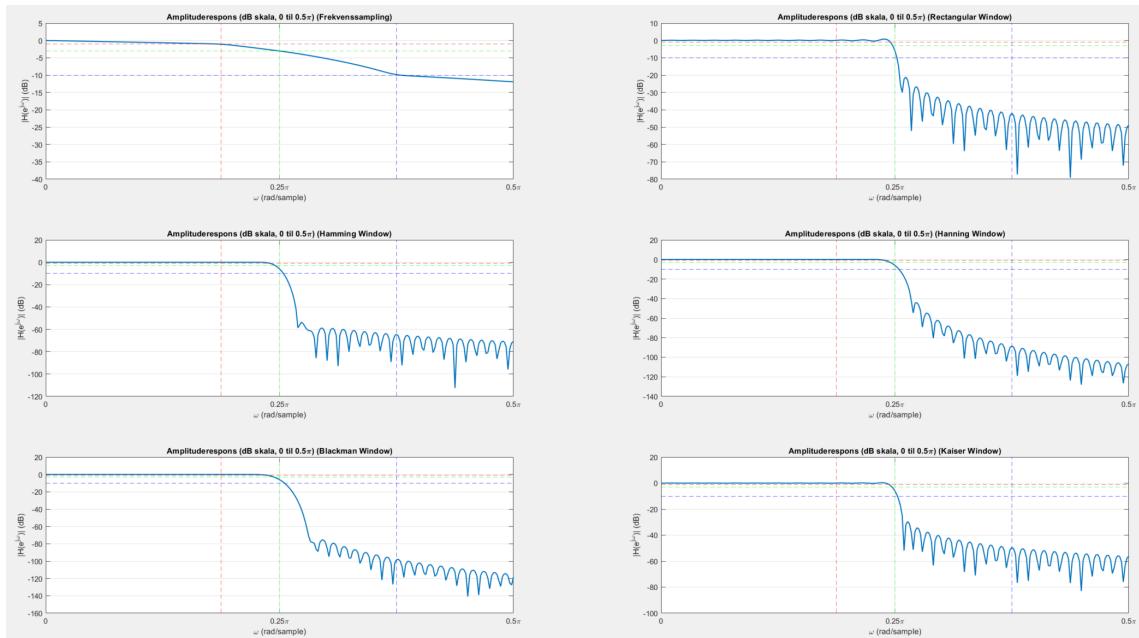


Figure 4.8: Zoomet version af amplituden ved $N=175$ for alle filtre/vinduer. Bemærk at frekvenssampling stadig er tættest på de ønskede værdier, på trods af en faktor 5 skalering i samples. Ikke desto mindre skal det bemærkes at de øvrige vinduer nærmer sig de ønskede værdier.

5 | Punkt 5

"Der foretages off-line simulering af filteret (Matlab, Python, ...), hvor dets evne til at "rense" et (selvkonstrueret) signal overlejret med støj illustreres."

For at kunne lave en simulering af filterets egenskaber, skal der ligesom i kapitel 2 initialiseres en håndfuld variabler, samt beregnes impulsrespons, frekvensrespons osv. Det gøres i første omgang ved:

```
1 % Parametre til frekvenssamplingsfilter
2 N = 35;                                     % Filterlaengde
3 fs = 8000;                                    % Samplefrekvens i Hz
4 frequencies = [0, 750, 1000, 1500, fs/2];    % Frekvenser i Hz
5 gains_dB = [0, -1, -3, -10, -40];           % Gain i dB (krav)

6
7 % Konverter gain fra dB til lineaer skala
8 gains_linear = 10.^^(gains_dB / 20);

9
10 % Normerer frekvenserne til [0, 1] (Nyquist-normalisering)
11 f_norm = frequencies / (fs / 2);

12
13 % Design impulsresponsen med 'fir2'
14 h = fir2(N-1, f_norm, gains_linear);

15
16 % Beregn frekvensrespons fra 0 til 2*pi (hele omraadet)
17 [H, omega] = freqz(h, 1, 1024, 'whole', fs);

18
19 % Skab et grundsignal til test
20 duration = 0.05;                             % Signalets varighed (kort for
                                                 % overskuelighed)
21 x_noisy = generateNoisySignal(fs, duration);

22
23 % Benyt filteret paa signalet
24 y_filtered = conv(x_noisy, h, 'same');        % Fold signalet med filteret
                                                 % uden at forlaenge signalet
```

Derefter skabes en funktion som kan lave noget "højfrekvent" støj på signalet:

```
1 function x_noisy = generateNoisySignal(fs, duration)
2 % Parametre for signalet
3 f_passband = 500;                            % Frekvens i pasbaandet
4 amplitude_passband = 1;                      % Amplitude for
                                                 % pasbaandssignalet
5
6 % Genererer tidsvektor
7 t = 0:1/fs:duration - 1/fs;
8
9 % Skab pasbaandskomponenten (ren 500 Hz sinusboelge)
10 signal_passband = amplitude_passband * sin(2 * pi * f_passband * t);
11
12 % Skab højfrekvent støj (frekvenser over 1500 Hz)
```

```

13 high_freqs = [1750, 2000, 2500, 3000]; % Stoejfrekvenser i stopbaandet
14 noise = 0; % Initialiser stoej
15
16 for f_noise = high_freqs
17     noise = noise + 0.5 * sin(2 * pi * f_noise * t); % Tilfoej hver
18         højfrekvente komponent
19 end
20
21 % Kombiner signal og stoej
22 x_noisy = signal_passband + noise;

```

Til sidst skal signalerne samt deres FFT'er plottes. Dette er vist i figur 5.1. I figuren kan det hurtigt synes af, at filteret ingen nævneværdig indvirkning har. Ikke desto mindre, dæmper filteret lige præcis det, som det er blevet specificeret til.

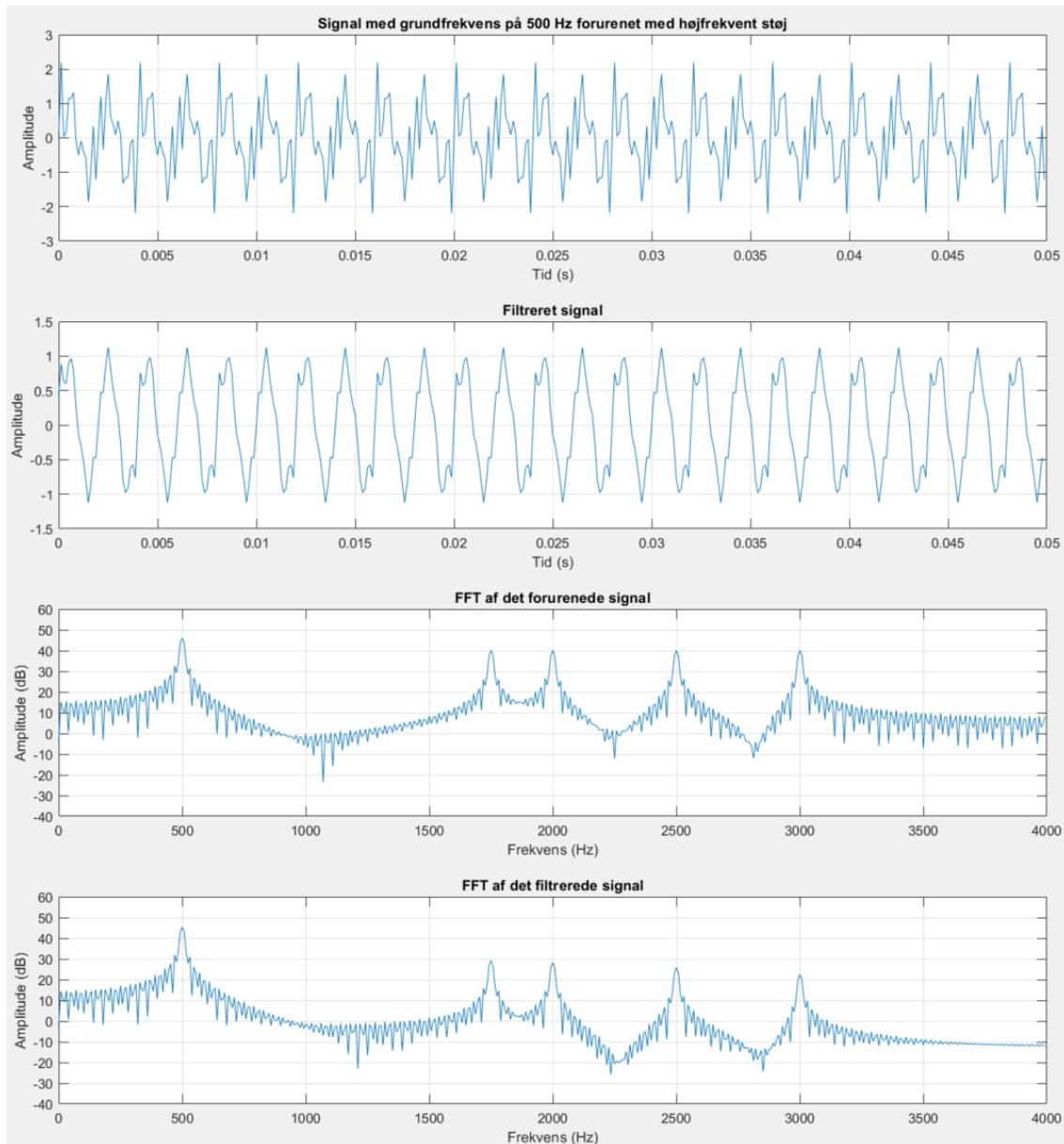


Figure 5.1: Som det kan ses på det filtrerede signal samt på den filtrerede FFT, er det begrænset hvor meget filteret reelt fjerner.

Hvis man kigger i figur 5.2, kan det grafisk aflæses, at det er meget begrænset hvor dæmpet signalet reelt bliver ved de givne "højfrekvente" signaler. Derfor kan man ved

f.eks. at aflæse ca -14 dB ved 2500 Hz i 5.2 godt se, at de tinder, som er til stede i FFT'en af det filtrerede signal, rent faktisk er dæmpet passende ifht. amplituderesponsen. Derfor kan man konkludere at filteret reelt filtrere, som specificeret af opgaven. Om det så er et anvendeligt filter i nogen henseender er en anden historie.

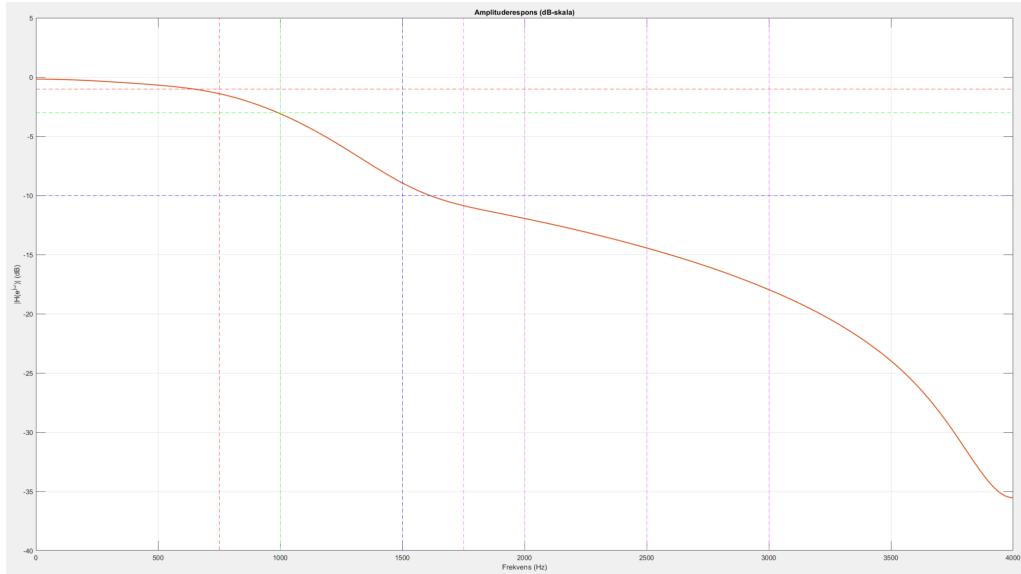


Figure 5.2: Når man kigger på filterets amplituderespons ved de støjende frekvenser (markeret med magenta), 1750 Hz, 2000 Hz, 2500 Hz og 3000 Hz, giver resultaterne for FFT'en vist i figur 5.1 god mening.

Hvis man absolut gerne vil have fjernet de tinder, som er til stede i FFT'en, kan filteres specifikation ændres. For pædagogikkens skyld, er nogen helt tilfældige, men meget voldsomme værdier valgt. Nu lyder specifikationen:

```

1 % Parametre til frekvenssamplingsfilter
2 N = 91;                                     % Filterlaengde
3 fs = 80000;                                    % Samplefrekvens i Hz
4 frequencies = [0, 750, 1000, 1500, 8000, fs/2]; % Frekvenser i Hz
5 gains_dB = [0, -1, -3, -10, -80, -80];        % Gain i dB (krav)

```

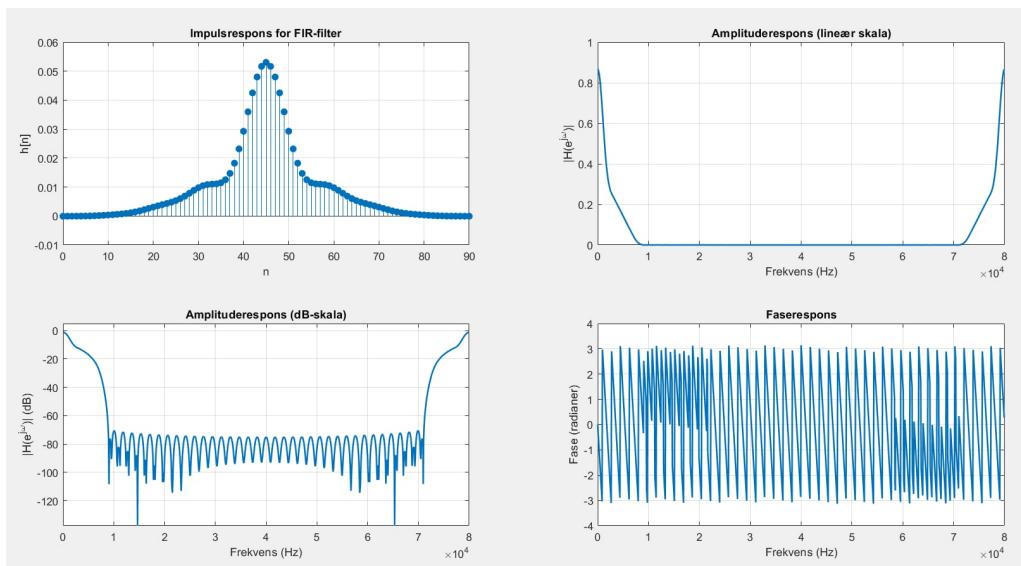


Figure 5.3: Et meget eksperimentalt frekvenssampling filter. I figuren ses impulsresponsen, amplituderesponsen (lineær og dB) samt faseresponsen.

Hvilket giver et filter, som matcher figur 5.3. Her er stadig tale om et frekvenssampling-filter, bare med samplingfrekvensen skruet op til 80 kHz, samtidig med at den ønskede dæmpning ved hhv. 8 kHz og nyquistraten (40 kHz) er sat til 80 dB. Det giver ydermere et filter, som pludselig viser side lobes/ripples, og derfor også en ikke 100% lineær faserespons.

Med det eksperimentielle filter kan man dog se, at de forurenende frekvenser i store træk er filtreret fra, som vist i figur 5.4

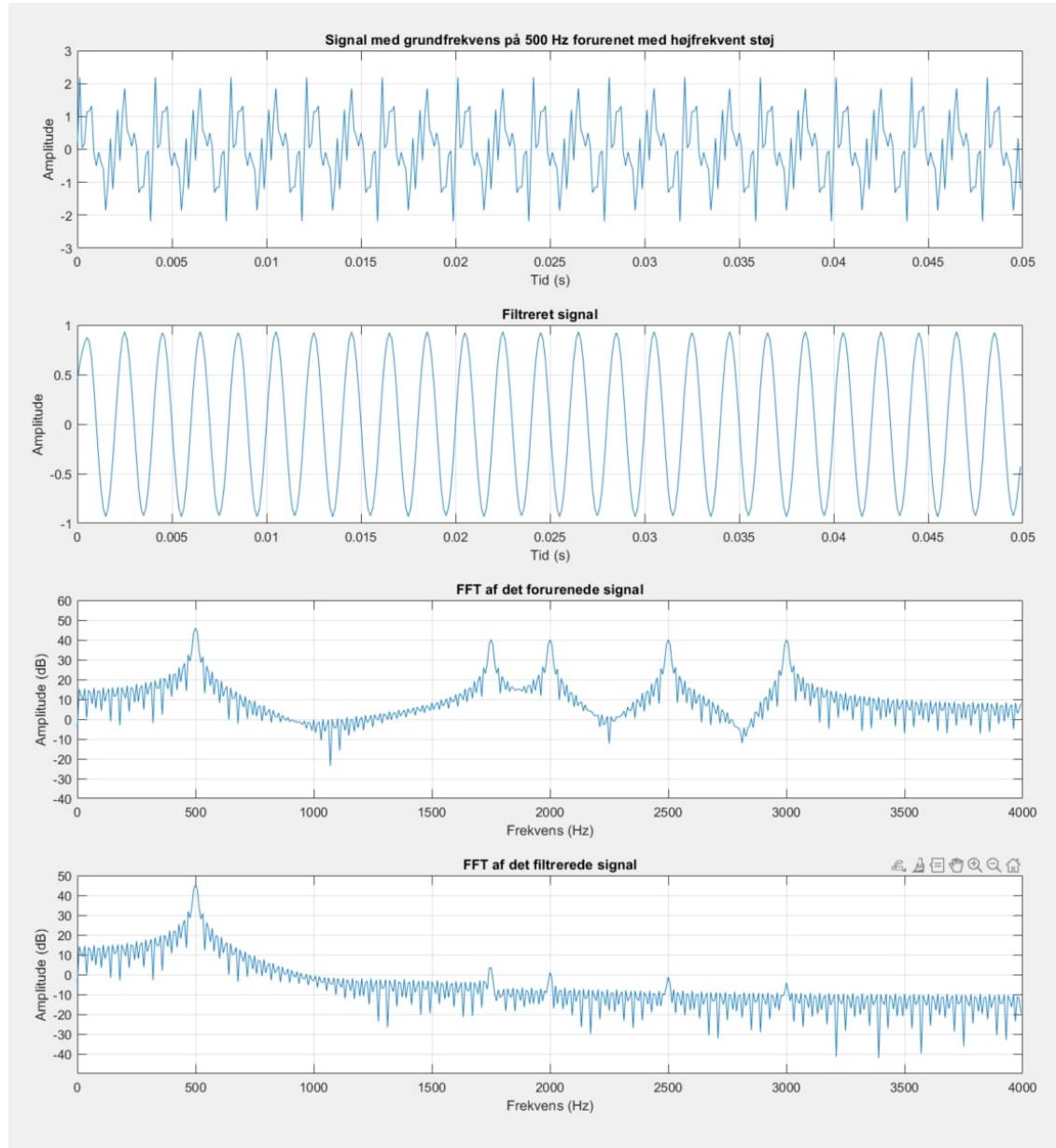


Figure 5.4: Nu lykkes det at fjerne størstedelen af de højfrekvente signaler, så sinusen på 500 Hz er tydelig igen.

A | Appendix A - Making a Filter by the Book

To ensure that the frequency filter made in MATLAB was correct, it was first tested with the specifications made in the book by I. Feachor, with a passband from 0-5 kHz, sampling frequency of 18 kHz and a filter length of N=9. The frequency sampling method made in MATLAB yielded exact replicas of the results shown in the book. In figure A.1 the result can be seen.

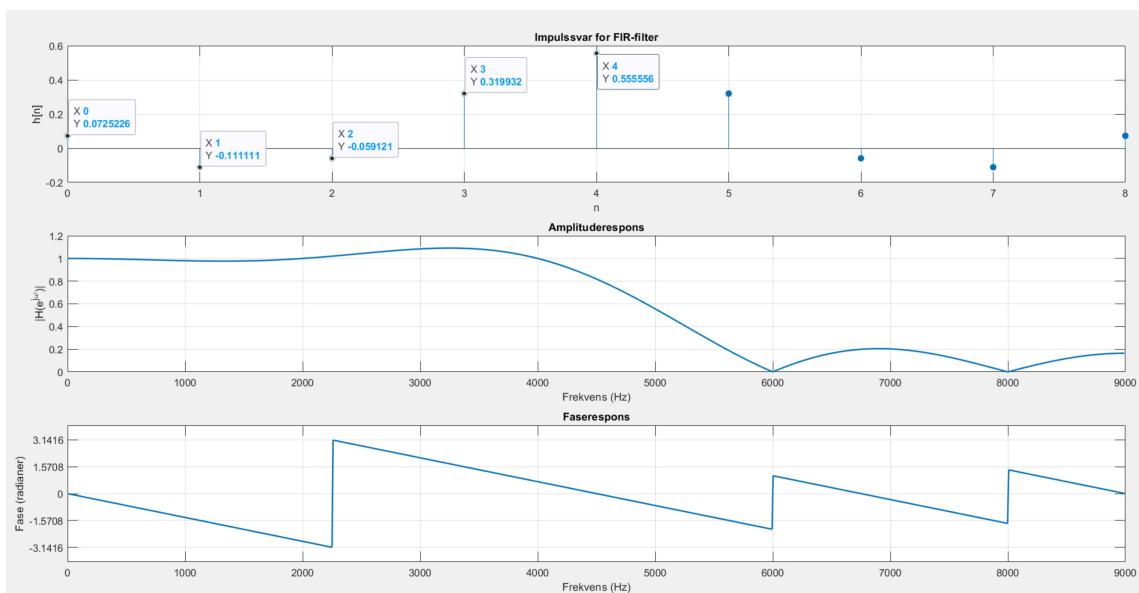


Figure A.1: MATLAB result from our developed frequency sampling method, based on the specifications made in the book.