Case Project 2 – Audio filter

|  |  |
| --- | --- |
| **Navn** | **Studienummer** |
| Naiyun Wu | 201405716 |
| Frederik Gadegaard | 201405625 |
| Maciej Lech | 20107672 |
| Jacob Aagaard | 201404442 |

Indhold

[Opgave 1 – Analyse af inputsignal 3](#_Toc445997417)

[Opgave 2 – IIR-notch filter design 5](#_Toc445997418)

[1. Skitsér pol-nul punkts diagram for filteret 5](#_Toc445997419)

[2. Udregn systemfunktion H(z) og opskriv differensligningen herudfra 6](#_Toc445997420)

[Systemfunktion H(z): 6](#_Toc445997421)

[Differensligning: 8](#_Toc445997422)

[3. Tegn signalgrafen for filteret på direkte form I. 9](#_Toc445997423)

[4. Bestem og plot frekvens-responset for filteret udfra H(z) – Er det et ideelt filter? 10](#_Toc445997424)

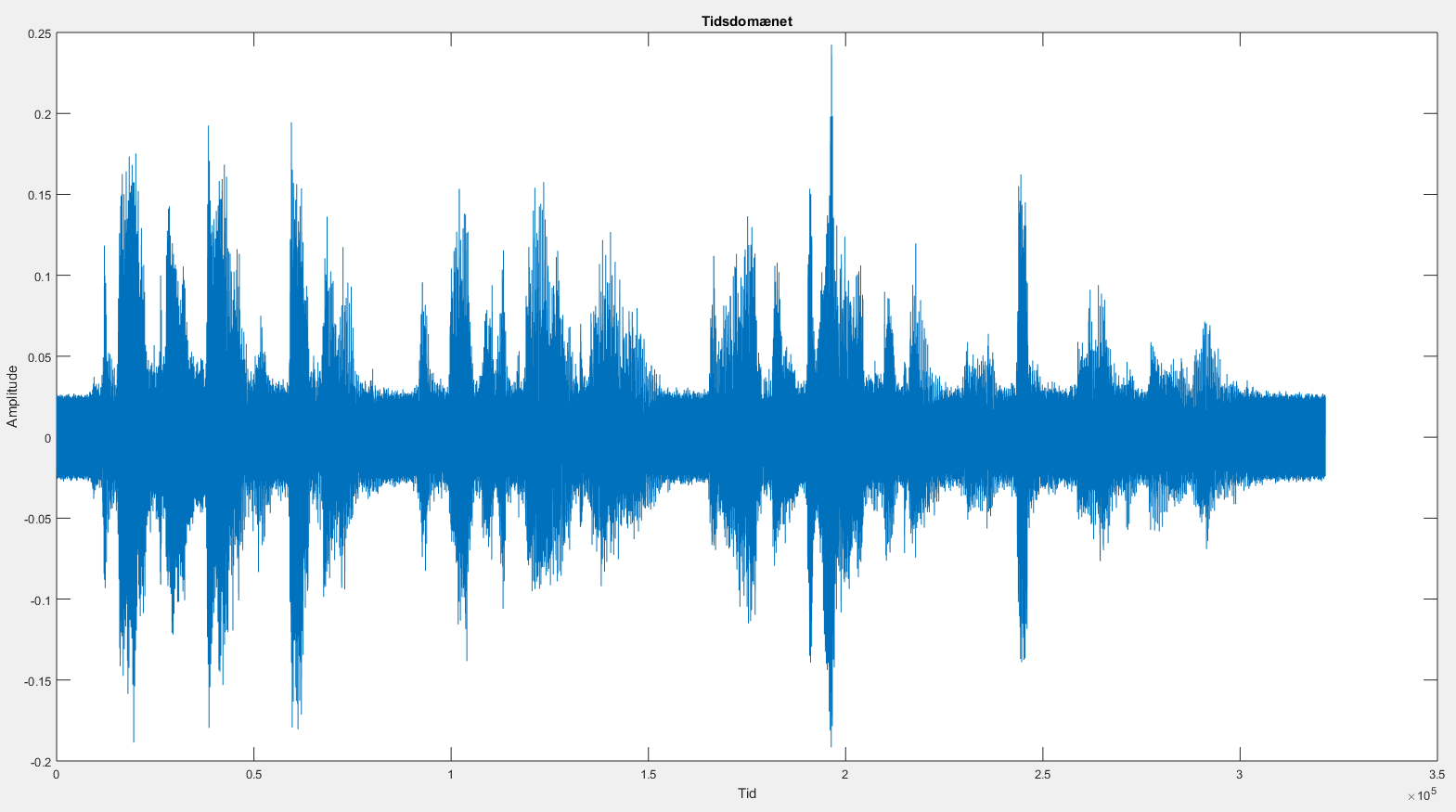
[5. Test at filteret virker ved at filtrere lydfilen i Matlab 10](#_Toc445997425)

[Opgave 3 – Implementering på blackfin processor 12](#_Toc445997426)

[Opgave 4 – Test af filteret 14](#_Toc445997427)

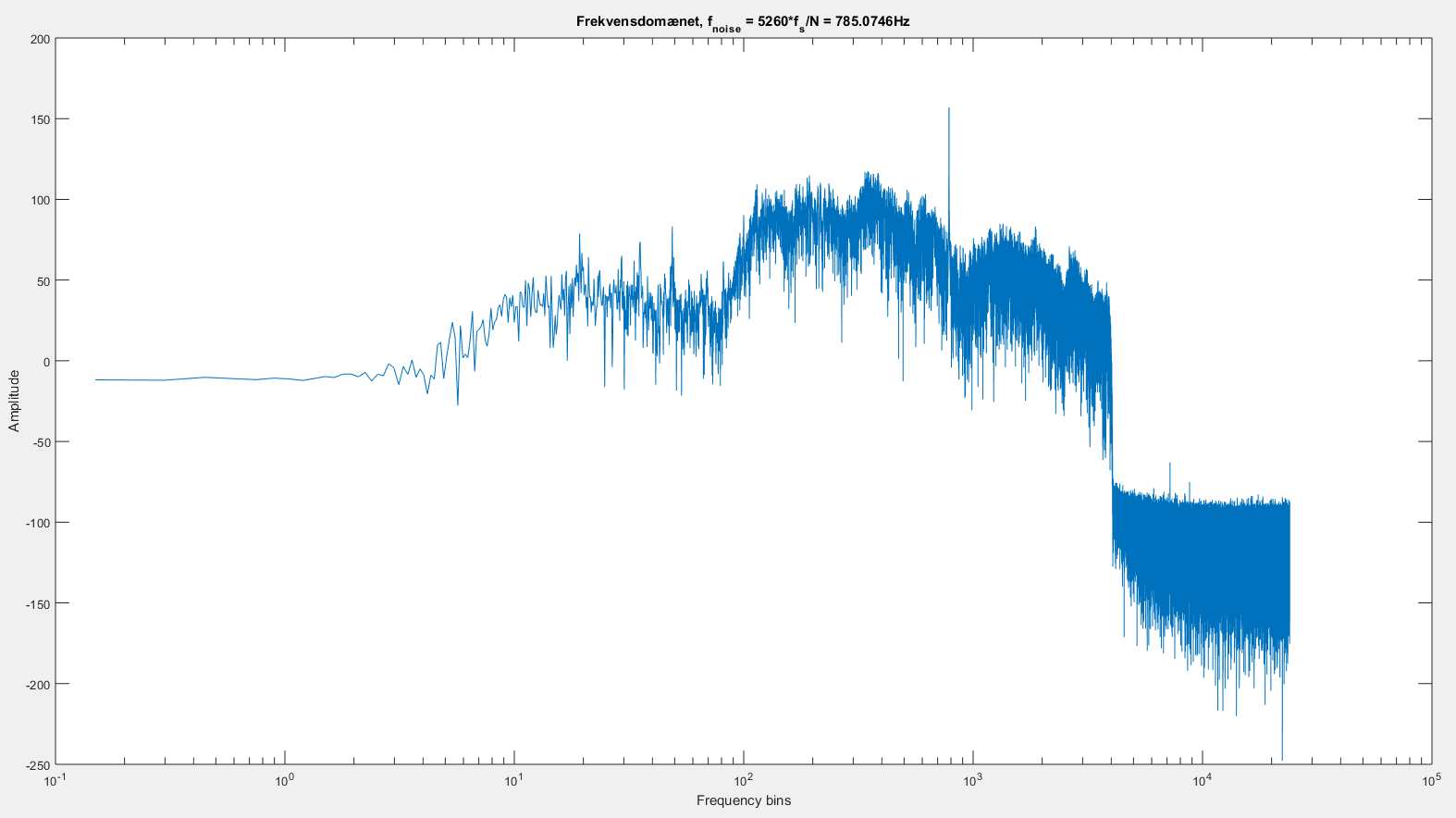
# Opgave 1 – Analyse af inputsignal

På Figur 1 ses signalet i tidsdomænet og det er tydeligt at se den forstyrrende tone hele tiden ligger i støjgulv neden under talesignalet.



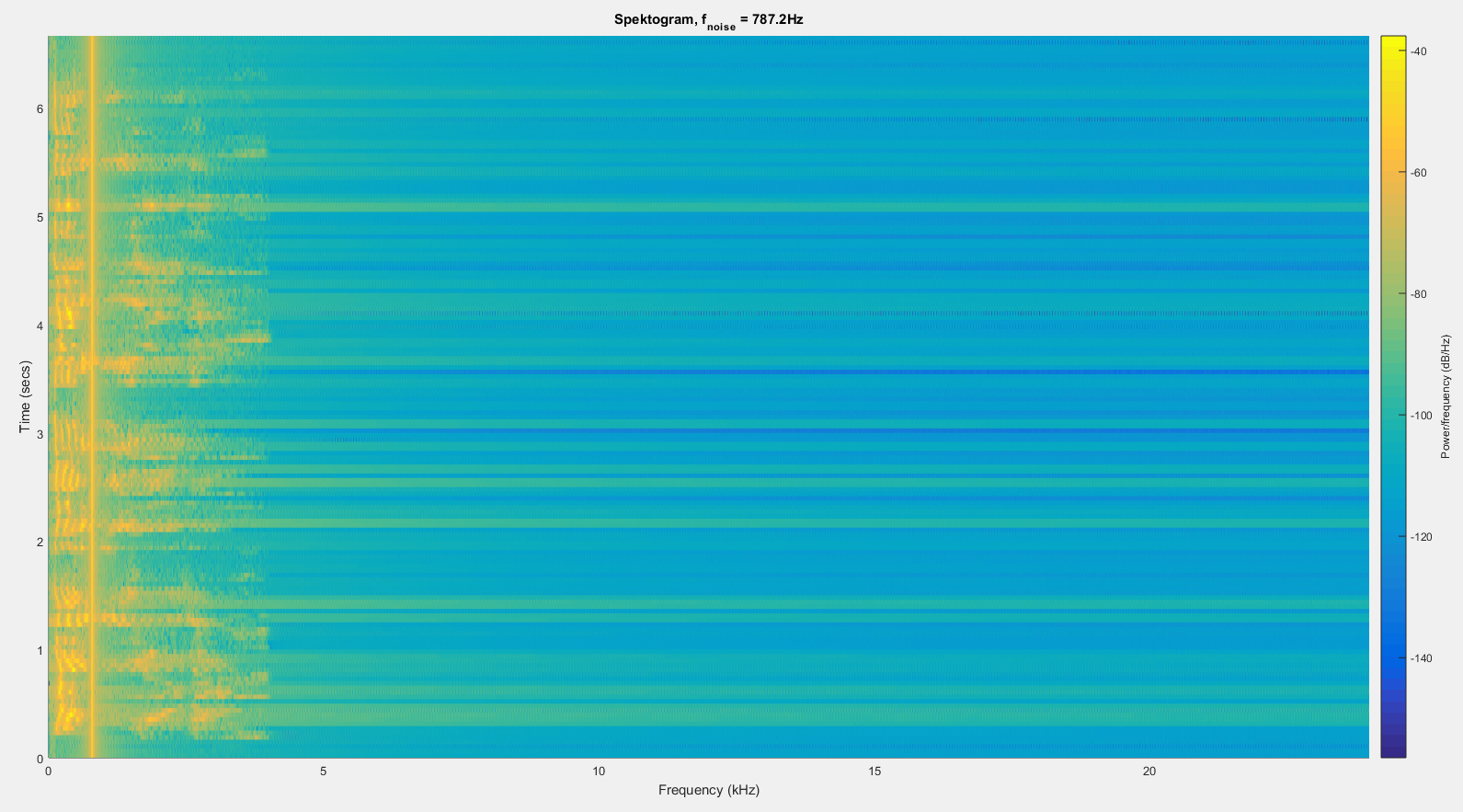
Figur tale\_tone i tidsdomænet

Når vi bevæger os over i frekvensdomænet, på Figur 2, ses den tydelige frekvens af støjen, der ligger ved bin nummer 5260, og svarer til frekvensen 785Hz, jf. nedenstående udregning:



Figur tale\_tone i frekvensdomænet

På Figur 3, spektogrammet, kan frekvensen bestemmes der hvor intensiteten er mest gul (og dermed, den frekvens, der har den højeste forstærkning). Det ses også tydeligt at det er en konstant tone, der ligger som baggrundsstøj igennem hele talesignalet - meget lig den oplevelse man får når man afspiller lydsignalet. Med den specifikke opløsning af spektogrammet, bestemmes frekvensen til . Meget tæt på frekvensen fundet i Figur 2.

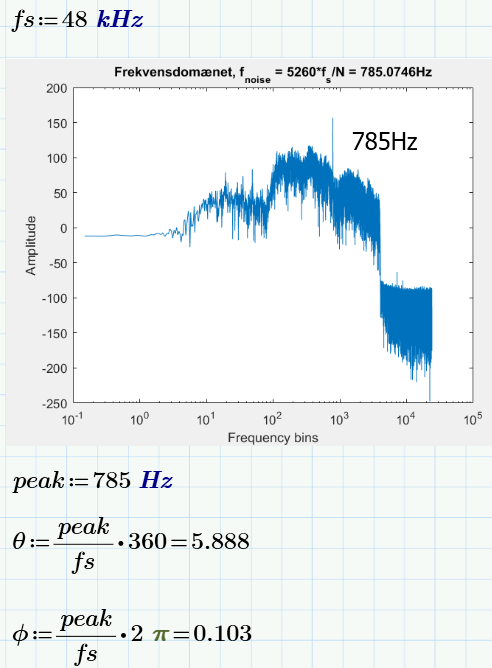


Figur tale\_tone i spectrogram

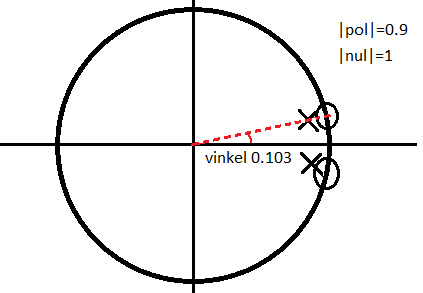
# Opgave 2 – IIR-notch filter design

## Skitsér pol-nul punkts diagram for filteret

Herunder i Figur 4 har vi i matcad beregnet os frem til den nomeret vinkelfrekvens som vores pol og nul punkt skal have i pol-nul punkts diagrammet i Figur 5.



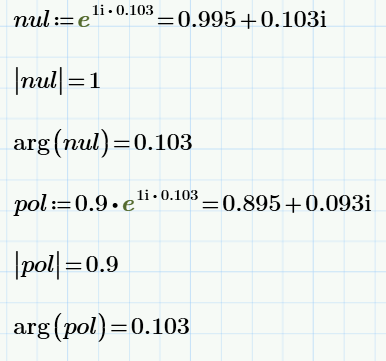
Figur udregning af den nomeret vinkelfrekvens i pol-nul punkts diagram



Figur Illustration af vores valgte poler og nul-punkter i et pol-nul punkts diagram

## Udregn systemfunktion H(z) og opskriv differensligningen herudfra

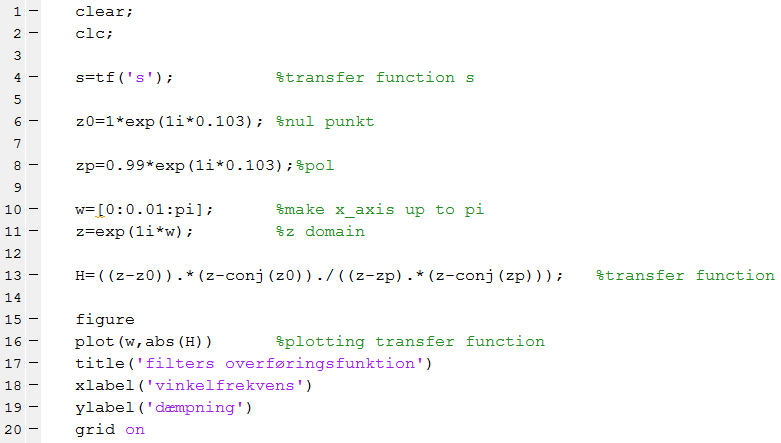
### Systemfunktion H(z):



Figur Udregning af poler og nul punkter på sumform

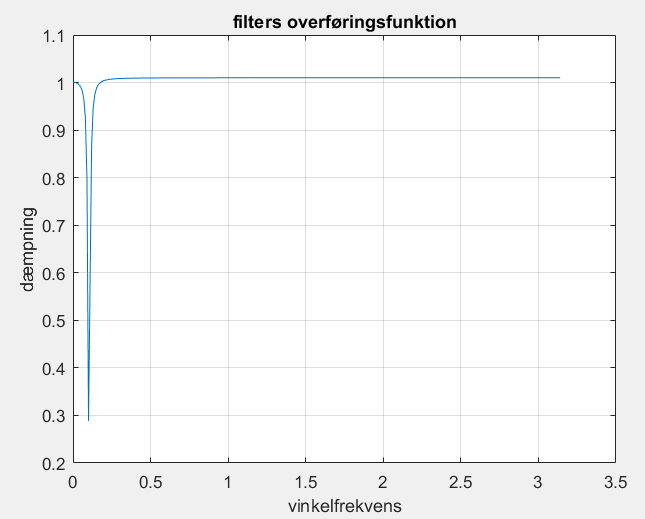
Herunder ses udregningen af overføringsfunktionen ud fra poler og nul punkter. Denne er implementeret i linje 13 i Figur 7.

Kodeudsnittet herunder viser hvordan vi i linje 10 har oprettet en vektor gående fra 0 til pi svarende til 0 til fs/2. Dette gøres for ikke at kigge på det spejlede billede også. Derefter defineres og overføringsfunktionen H plottes herefter i linje 16 som kan ses i Figur 8.



Figur Kodeudsnit af beregningen af overføringsfunktionen

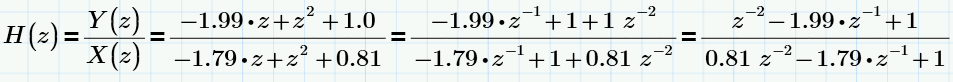
I Figur 8 ses vores IIR-notch filter som dæmper frekvenser ved .



Figur Overføringsfunktion H(z) plottet i linje 16 i Figur 7

### Differensligning:

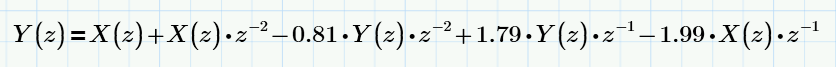
Herunder gennemgås udledningen af differensligningen ud fra overføringsfunktionen H(z):



Figur H(z) overføringsfunktion



Figur X(z) er ganget på begge sider af lighedstegnet



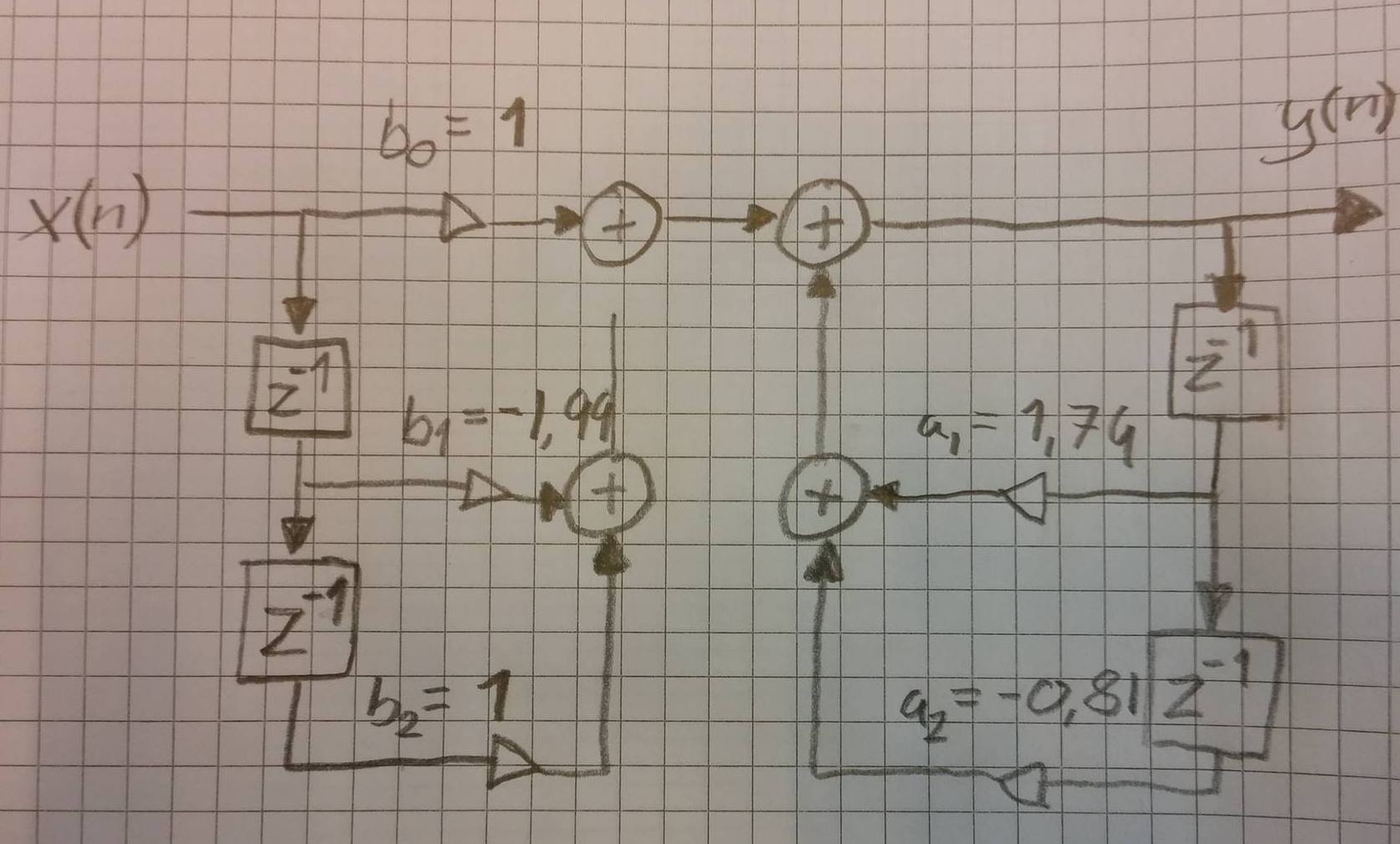
Figur Y(z) og X(z) er ganget ind i parentesen og Y(z) er isoleret



Figur Endelige differensligning

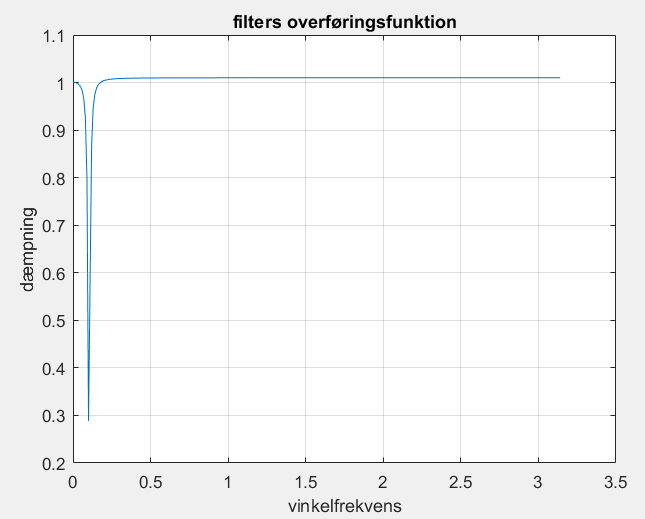
## Tegn signalgrafen for filteret på direkte form I.

Ved brug af filterkoefficienterne kan vi opstille signalgrafen på direkte form I, som ses på Figur 13. Den giver et overblik over selve udregningen i z-domænet fra input til output. Det tydeliggøre de forskellige delays og hvilken tilhørende koefficient som bliver ganget på.



Figur signalgraf skitseret på direkt form I

## Bestem og plot frekvens-responset for filteret udfra H(z) – Er det et ideelt filter?

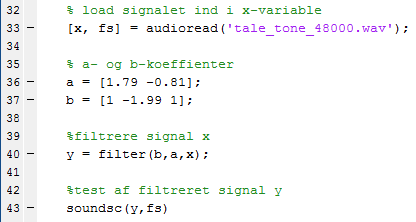


Figur Overføringsfunktion H(z)

Det er ikke en ideelt filter. Et ideelt filter vil have en uendelig orden og lodret knæk ved bandstop frekvensen. Dette ville vi aldrig kunne opnå i praksis på et digitalt filter da vi ikke kan have en uendelig orden.

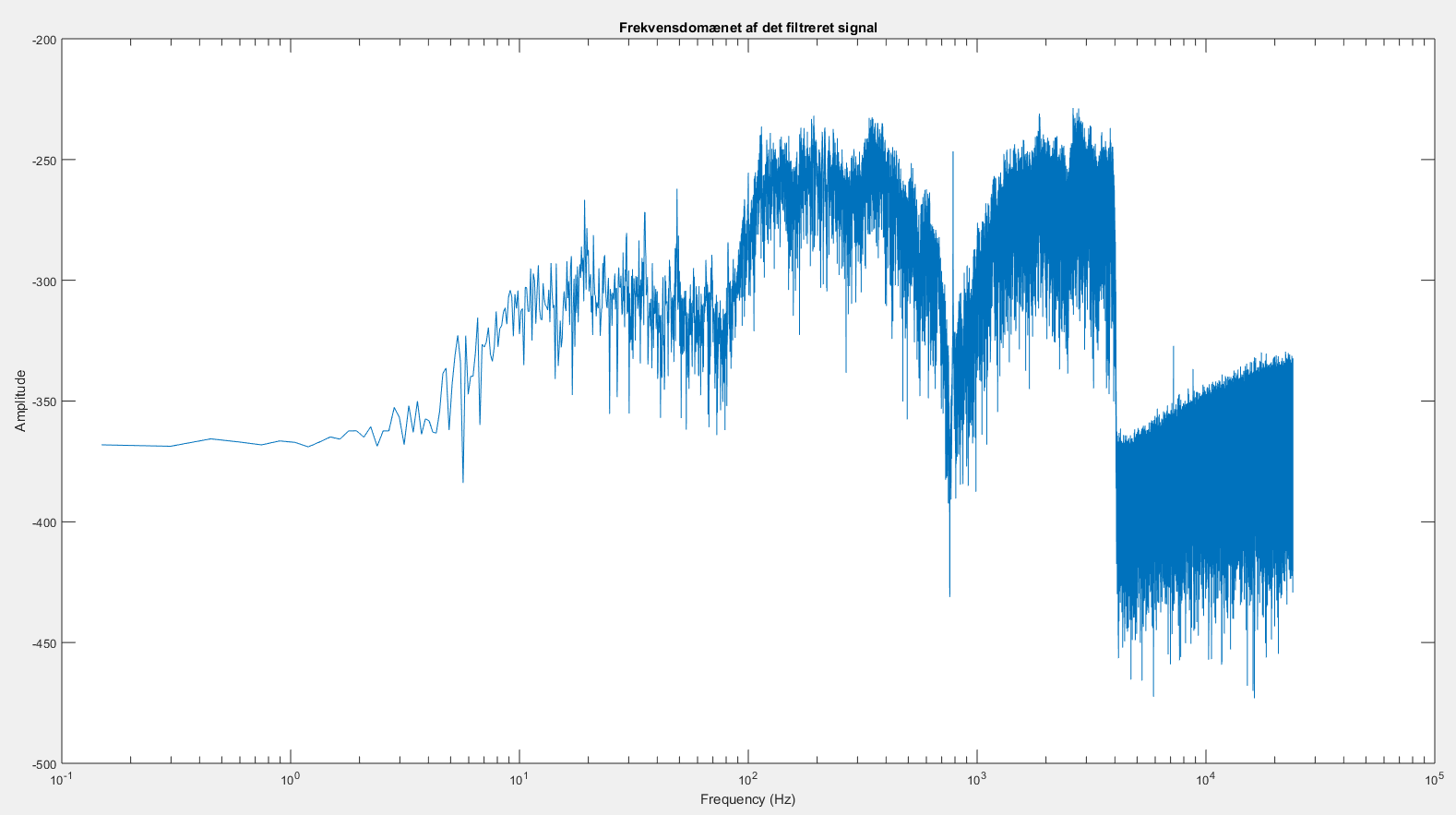
## Test at filteret virker ved at filtrere lydfilen i Matlab

Herunder i Figur 15 ses koden for testen af filteret. Vi anvender den indbyggede filter – funktion i matlab til at konstruere filteret ud fra filterkoefficienterne a og b. Vi lyttede til det filtreret lydsignal og kunne verificere at den støjende tone var filtreret væk.



Figur kodeudsnit af test af det filtreret lydsignal

Resultat af filterets påvirkning på signalet ses i Figur 16 herunder. Som det ses i forhold til frekvens-analysen i ##, er tonen på de 785 Hz dæmpet betydeligt. Her ses det også at vores filter ikke er ideelt og at det dæmper nærtliggende frekvenser en smule også.



Figur FIlteret lydsignal i frekvensdomænet

# Opgave 3 – Implementering på blackfin processor

Nedenunder ses koden for vores implementeret filter i funktionen myVolume.

1 **#include** "Talkthrough.h"

2 // Modify and insert your notch filter here!!!!

3

4 **short** xn=0, x1=0, x2=0;

5 **short** y1=0, y2=0;

6 **short** out;

7 **int** tmp\_out;

8

9 **short** **myVolume**(**short** x)

10 {

11 static **short** b0 = 8192; // Values from MATLAB, times 2^13

12 static **short** b1 = -16302;

13 static **short** b2 = 8192;

14 static **short** a1 = 14664;

15 static **short** a2 = -6636;

16

17 //Differensligning

18 tmp\_out = (**int**) b0\*x + (**int**) b1\*x1 + (**int**) b2\*x2 + (**int**) a2\*y2 + (**int**) a1\*y1;

19

20 out = tmp\_out >> 13; // Divide output by 2^13

21 y2 = y1; // y(n-2)

22 y1 = out; // y(n-1)

23

24 x2 = x1; // x(n-2)

25 x1 = x; // x(n-1)

26

27 return out;

28 }

For at implementere IIR notch filteret på Blackfin processoren, er det nødvendigt at overveje type konvertering nøje. Da det ikke er muligt for os at arbejde med floating-point variable, har vi på forhånd skaleret filterets koefficienter med en faktor 2^13. Disse værdier oprettes som short variable, da 16-bit er rigeligt til at opbevare disse i.

Differensligningens output lagres midlertidigt i en integer i linje 18, og selve beregningerne typecastes også til integers, for at sikre at der ikke sker overflow, når to short variable ganges sammen. Denne værdi shiftes derefter 13 pladser til højre, i linje 20, og lægges ind i variablen out af typen short. Derefter lægges den sidste output-værdi y1 ind i y2 for at implementere delay-funktionaliteten, y(n-2) og out lægges tilsvarende i y1, for at implementere y(n-1). Det samme gælder i linje 24 og 25 for delay af x-variablene.

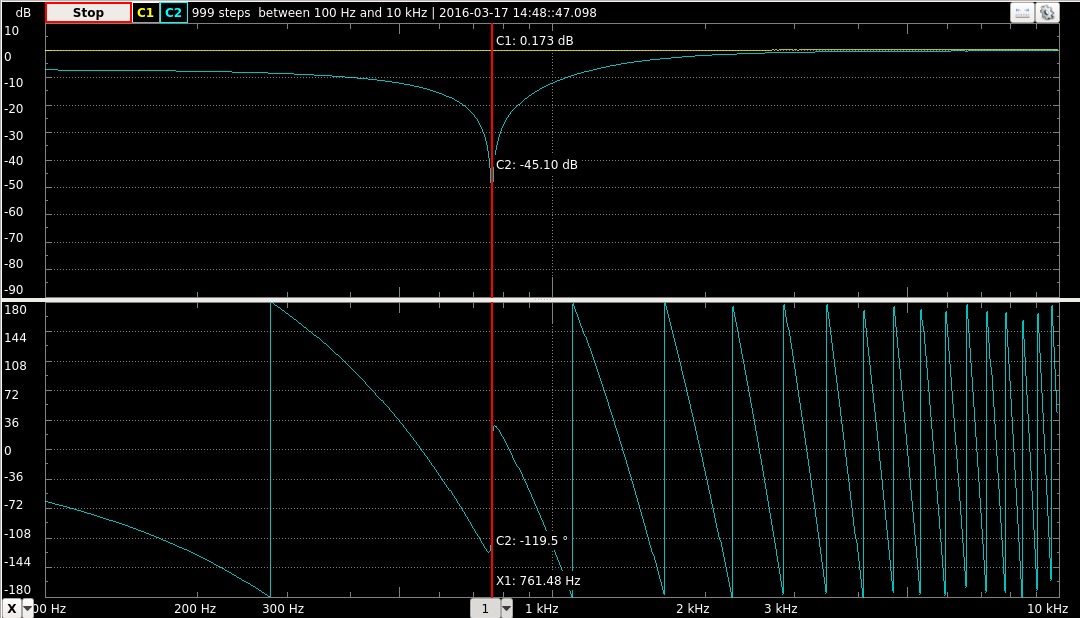
Til slut returneres out variablen.

Undervejs havde vi problemer med at vi hørte støj når alle koefficienter indgik i differensligningen, men vi prøvede at fjerne y-leddene, for at se effekten af et FIR notch filter. Det resulterede i at hele lydsignalet blev dæmpet, og at noget af tonen blev fjernet. Dernæst måtte vi debugge koden, for at finde ud af hvor problemet lå med y-leddene. Det viste sig at vi havde implementeret delay-funktionaliteten i forkert rækkefølge, hvilket resulterede i at outputtet fra differensligningen voksede konstant, altså havde vi fået lavet et ustabilt filter.

# Opgave 4 – Test af filteret

Vi lyttede til output-signalet direkte fra blackfin processoren og kunne verificere at den støjende tone var dæmpet væsentligt og næsten ikke længere hørbar.

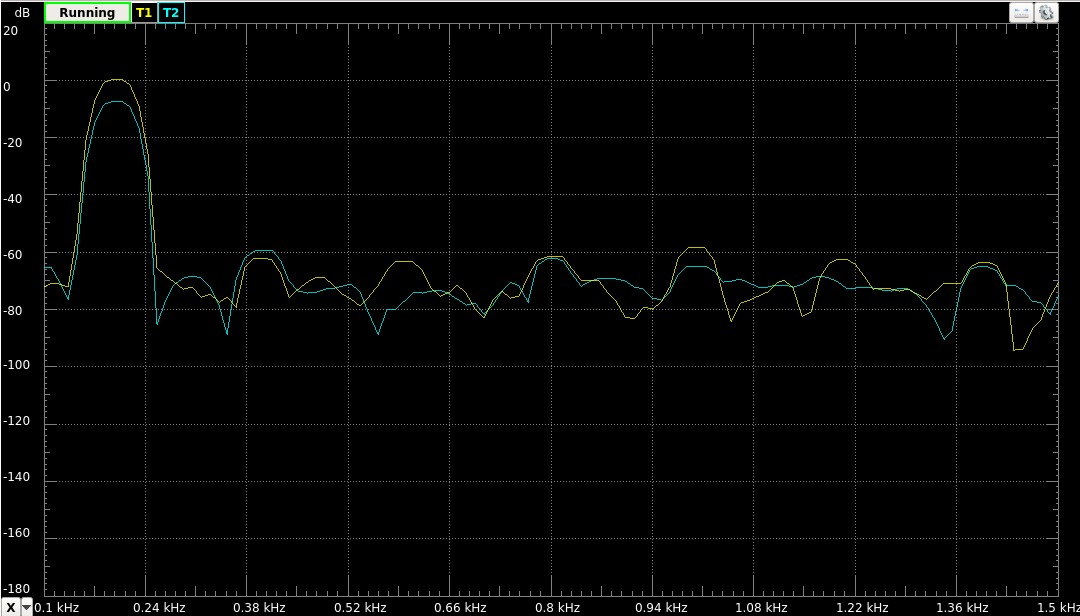
Her i Figur 17 ses en netværks analyse af det realiseret notch filter på blackfin processoren. Vi får en dæmpning på ca. 45 dB omkring det forventede område. Peaken er aflæst til 761 Hz hvor det teoretisk skulle være 785 Hz. Dette kan skyldes afrunding af filterkoefficienter eller usikkerhed i aflæsning på grafen.



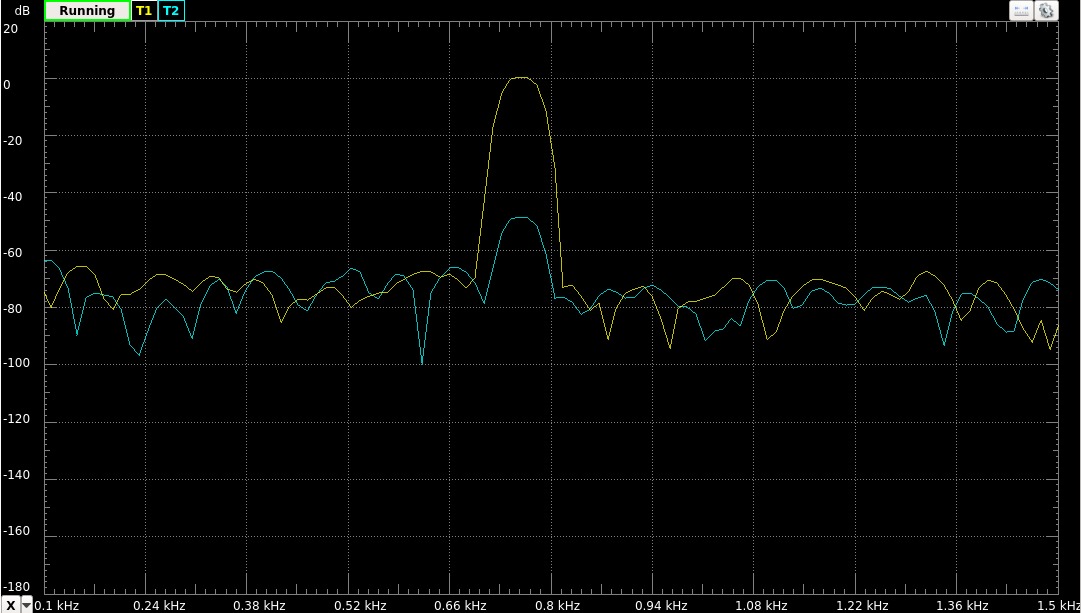
Figur Frekvens analyse filteret via Analog Discovery (Network Analyzer)

For at sammenligne med input-signalet har vi lavet en spectrum-analyse også, for at se forholdet i mellem input og output ved to forskellige frekvenser. Begge grafer er normeret efter input-signalet.

I Figur 18 havde input-signalet en frekvens på 200 Hz og i Figur 19 nedenunder havde input signalet en frekvens på 785 Hz. Som forventede observeres en lille dæmpning ved de 200 Hz på ca. 7 dB og en meget større dæmpning ved de 785 Hz på ca. 45 dB. Dette bekræfter IIR-notch filterets virkning.



Figur Inputsignal = 200 Hz



Figur Inputsignal = 785 Hz