Spektografisk analyse av “Für Elise” ved bruk av Fourier-transformasjoner

ING2501-1 24H Matematiske Metoder 2

Adrianne Bendiksen, Brede Solberg,
Stian Loddengaard, Askil Uppstad, & Andreas Kragseth

03.november 2024
Cyberingeniørskolen 2024

Sammendrag

I dette prosjektet analyseres lydsignaler fra musikkstykket Für Elise ved hjelp av Short-Time Fourier Transform (STFT) og Fast Fourier Transform (FFT) for å identifisere musikalske noter og forstå frekvensvariasjoner over tid. STFT ble valgt som hovedmetode fordi den kombinerer tids- og frekvensanalyse, som er nødvendig for å fange de dynamiske endringene i musikk. Gjennom ulike eksperimenter ble Hanning-vinduet identifisert som den mest passende vindusfunksjonen, da det ga en god balanse mellom tids- og frekvensoppløsning og reduserte støy i analysen.

Et viktig eksperiment i prosjektet var å legge til og fjerne en 3000 Hz tone fra lydsignalet for å demonstrere hvordan frekvensfiltrering kan brukes til å isolere og fjerne uønskede lyder. Resultatene viste at STFT og DFT kan fjerne spesifikke frekvenser uten å påvirke andre deler av signalet, noe som er nyttig for lydredigering.

Prosjektet belyser både styrker og begrensninger ved STFT i noteidentifikasjon, særlig ved analyser av komplekse polyfoniske signaler der noter overlapper. Videre forskning kan utforske alternative metoder som wavelet-transformasjon, som gir tilpasningsdyktig oppløsning, og justering av vindusparametere for å forbedre STFTs presisjon i slike analyser. Oppgavens funn har relevans for lydbehandling i musikkteknologi, særlig innen automatisert noteidentifikasjon og støyreduksjon.

Innholdsfortegnelse

1 Innledning

I dette prosjektet har vi som mål å utforske Short-Time Fourier Transform (STFT) for å analysere lydsignaler, med fokus på gjenkjenning av musikalske noter. Fouriertransformasjonen er et essensielt verktøy innen signalbehandling som gjør det mulig å bryte ned et signal i dets frekvenskomponenter. Mens standard Fouriertransformasjon gir et overordnet bilde av frekvensinnholdet, tar den ikke hensyn til hvordan frekvensene varierer over tid. STFT løser dette ved å analysere små tidsvinduer av signalet, noe som gir mulighet til å studere både tids- og frekvensdimensjoner samtidig. Dette er spesielt nyttig for komplekse lydsignaler, slik som musikk, der frekvensene endres dynamisk over tid.

For å gjennomføre denne analysen har vi valgt "Für Elise", et kjent musikkstykke komponert av Ludwig van Beethoven. "Für Elise" er velegnet fordi det inneholder både enkle noter og komplekse akkorder, noe som gir oss en god mulighet til å teste STFTs effektivitet.

STFT er mye brukt i musikkteknologi for å identifisere noter, noe som krever en nøyaktig forståelse av hvordan frekvenser varierer i tid. Prosjektet vårt utforsker STFTs effektivitet for denne oppgaven. Vi vil teste ulike vindusparametere og støyfiltrering for å evaluere STFTs nøyaktighet i noteidentifikasjon. Dette vil gi oss innsikt i metodens presisjon, robusthet og begrensninger.

1.1 Problemstilling

Hovedproblemstillingen dette prosjektet tar for seg er:
"Hvordan kan Short-Time Fourier Transform (STFT) og Fast Fourier Transform (FFT) brukes til å identifisere og modifisere frekvenser i lydsignaler fra et kjent musikkstykke, og hvilke faktorer påvirker nøyaktigheten av noteidentifikasjonen?"

Målet med prosjektet er å utvikle en matematisk metode for å identifisere noter fra lydsignaler ved hjelp av STFT, og å analysere hvilke teknikker og parametere som gir best resultat for nøyaktig gjenkjenning. Gjennom prosjektet vil vi utføre en praktisk analyse ved hjelp av et klipp fra den kjente sangen "Für Elise". Dette klippet vil fungere som et konkret eksempel der vi kan teste STFT-metodens evne til å identifisere noter i et ekte musikkstykke. Vi vil spesielt fokusere på hvordan valg av matematiske parametere, som vindustype og -størrelse, samt støyfiltrering, påvirker nøyaktigheten av analysen. Gjennom vår tilnærming vil vi også vurdere metodens begrensninger og utforske muligheter for videre forbedringer.

For å svare på hovedproblemstillingen vil vi fokusere på tre underproblemstillinger:

1. *Hvordan påvirker valg av vindustype og -størrelse nøyaktigheten i noteidentifikasjonen ved bruk av STFT?*
2. *Hvordan kan en uønsket note fjærnes fra et lydsignal vet hjelp av DFT?*
3. *Hvordan presterer STFT når det gjelder å identifisere noter fra et ekte musikkstykke som inneholder komplekse lyder og overlappende frekvenser?*

1.2 Metode og Tilnærming

For å oppnå målene vil vi bruke STFT til å analysere ulike lydfiler og generere spektrogrammer som viser hvordan frekvensinnholdet varierer over tid. Dette gir en visuell og kvantitativ måte å forstå lyden på, og vil hjelpe oss med å identifisere hvilke noter som spilles. Prosjektet implementeres ved hjelp av Python-biblioteker som librosa, numpy og matplotlib, som gjør det mulig å lese inn lydfiler, utføre Fourieranalyse, og presentere resultatene grafisk. Vi vil eksperimentere med forskjellige vindustørrelser og parametere for STFT, samt analysere effekten av disse innstillingene på gjenkjenningsnøyaktigheten. Gjennom praktiske eksperimenter med lydfiler vil vi utføre egne analyser for å trekke konklusjoner om hvilke faktorer som er avgjørende for presis

noteidentifikasjon. Dette prosjektet er derfor mer enn en teoretisk gjennomgang av STFT; det inkluderer også våre egne vurderinger og funn, som vil bidra til en dypere forståelse av metodens styrker og svakheter.

1.3 Rapportens Struktur

Rapporten vil være strukturert som følger: Først vil vi gjennomgå relevant teori om Fouriertransformasjon, FFT og STFT, som vil gi det nødvendige grunnlaget for å forstå de matematiske prinsippene bak lydanalysemetodene vi bruker. Deretter vil vi beskrive metodene vi har benyttet for å samle inn og behandle data, inkludert detaljer om datainnsamling, implementeringen av STFT, og eksperimentering med ulike vindusparametere. I resultatseksjonen vil vi presentere våre analyser, og fremheve hvordan forskjellige parameterinnstillinger påvirker nøyaktigheten av noteidentifikasjonen. Diskusjonsdelen vil tolke resultatene, sammenligne STFT med andre potensielle frekvensanalysemetoder, og vurdere både styrker og svakheter ved tilnærmingen vår. Til slutt vil vi oppsummere hovedfunnene våre i konklusjonen, diskutere mulige begrensninger, og peke på fremtidige forbedringsområder og potensial for videre forskning.

2 Teori

2.1 Fouriertransformasjon

Fouriertransformasjonen er en matematisk integraltransformasjon som benyttes til å analysere og dekomponere signaler i deres grunnleggende frekvenskomponenter. Ved hjelp av fouriertransfo ,rmasjonen kan man identifisere mengden og typen av hver frekvens som bygger opp et signal.

Fouriertransformasjonen konverterer et signal fra tidsdomenet, hvor informasjonen uttrykkes over tid, til frekvensdomenet, hvor signalets frekvensinnhold kan analyseres. Dette gjør det enklere å forstå og manipulere signaler basert på deres frekvensinnhold, noe som kan være avgjørende for for eksempel filtrering, støydemping eller spektralanalyse.

I motsetning til fourierserien, som benyttes for å representere periodiske funksjoner, kan fouriertransformasjonen anvendes på ikke-periodiske signaler. Dette betyr at den er anvendbar for analyse av et bredt spekter av signaltyper, inkludert enkeltstående impulser og andre ikke-periodiske signaler.

[1]

2.1.1 Fourierserie

En fourierserie uttrykker en periodisk funksjon som en sum av trigonometriske funksjoner. For en periodisk funksjon $f(x)$ med periode T, kan vi representere funksjonen som en uendelig sum av eksponensielle (trigonometriske) funksjoner gjennom fourierserien:

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{i2\pi\frac{n}{T}x} \tag{1}$$

der c_n er fourierkoeffisientene som bestemmer amplituden til hver frekvenskomponent $\frac{n}{T}$. Disse koeffisientene kan beregnes med:

$$c_n = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} f(x)e^{-i2\pi\frac{n}{T}x}dx \tag{2}$$

fourierkoeffisienten c_n angir styrken til den diskrete frekvensen $\frac{n}{T}$ i funksjonen.

2.1.2 Fouriertransformasjon

For å komme frem til et uttrykk for fouriertransformasjonen kan man ta utgangspunkt i fourierkoeffisienten c_n vist i Formel 2. c_n forteller om hvor mye av en frekvens som finnes i et periodisk signal. Når perioden T øker og nærmer seg uendelig, blir frekvensene $\frac{n}{T} = \xi$ så tette at de danner et kontinuerlig spekter. Dette gjør funksjonen ikke-periodisk, og summen i fourierserien går over til et integral. Vi kan da definere den kontinuerlige fouriertransformasjonen for en funksjon $f(x)$, der frekvensen ξ varierer kontinuerlig som:

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)e^{-i2\pi\xi x} \tag{3}$$

Her gir $\hat{f}(\xi)$ amplituden til frekvens ξ i funksjonen $f(x)$ over et kontinuerlig frekvensspekter.

Om man ser på definisjonen av fourierserien gitt i Formel 1, kan man sette inn $\hat{f}(\xi)$ for c_n og si at periode T går mot uendelig. Dette gir et uttykk som transformerer en funksjon $\hat{f}(\xi)$ fra frekvensdomenet til tidsdomenet, en invers fouriertransformasjon.

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\xi) e^{i2\pi\xi x} \tag{4}$$

Denne formelen viser hvordan vi kan rekonstruere $f(x)$ ved å integrere over alle frekvenskomponenter ξ.

Disse transformasjonene kan også skrives på en kortere form:

$$\begin{aligned} \hat{f}(\xi) &= \mathcal{F}\{f(x)\} \\ f(x) &= \mathcal{F}\left\{\hat{f}(\xi)\right\} \end{aligned} \tag{5}$$

[1]

2.1.3 Fouriertransformasjon for vinkelfrekvens

Når x representerer tid t, kan man sette inn vinkelfrekvens ω definert ved $\omega = 2\pi\xi$ i definisjonen av fouriertransformasjonen vist i Formel 3.

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx \tag{6}$$

Vi navngir denne formelen $F(\omega)$ for å skape en klar forskjell fra formelen $\hat{f}(\xi)$ som har frekvensen ξ som argument. [1]

2.2 Diskre fouriertransformasjon (DFT)

Definisjonen av fouriertransformasjonen gitt i Formel 3 antar at man har en kontinuerlig funksjon som representerer dataen man jobber utifra, noe man ikke får av faktiske måleinstrumenter.

En diskre fouriertransformasjon tar inn N datapunkter og gir amplituden til forskjellige diskre frekvenser som finnes i signalet. Siden vi ikke kan finne et kontinuerlig frekvensspekter med diskre data, fordeles frekvensspekteret i N like store deler. Dette gir ξ_0, frekvensforskjellen mellom hver frekvensindeks.

$$\xi_0 = \frac{k}{N} \tag{7}$$

$$\xi = \xi_0 n \tag{8}$$

Der N er antallet datapunkter og k er frekvensindeksen. Om man setter dette inn i definisjonen for fourier transform gitt i Formel 3 og endrer fra integrasjon til summering, får man definisjonen for DFT:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi \frac{k}{N} n} \tag{9}$$

Der X_k forteller i hvilken grad frekvensen gitt ved frekvensindeks k er i signalet.

[2]

2.2.1 Fast fourier transform (FFT)

FFT, eller *Fast Fourier Transform*, er en optimalisert form av diskret fouriertransformasjon (DFT). Siden trigonometriske funksjoner har en periodisk natur, vil mange av beregningene i en DFT resultere i like verdier på flere punkter. I en vanlig DFT med N målingspunkter, må vi gjennomføre N^2 beregninger – noe som blir uoverkommelig når vi håndterer store datamengder. FFT løser dette

ved å gjenbruke tidligere resultater i stedet for å beregne dem på nytt, og dermed reduseres antallet nødvendige operasjoner drastisk. [3]

La oss anta at vi har N målingspunkter der N kan deles som $N = N_1 N_2$ altså N_1 mindre DFTer av størrelse N_2. Selv om det finnes flere typer FFT-algoritmer, skal vi fokusere på *Cooley-Tukey-algoritmen*, som er den mest brukte.

Cooley-Tukey-algoritmen deler datasettet inn i punkter med henholdsvis jevne og odde indekser, hvor vi noterer jevne indekser som x_{2m} og odde indekser som x_{2m+1}. Denne delingen reduserer mengden beregninger vi må utføre, da hver beregning kun trenger å gjøre halvparten av operasjonene.

$$X_k = \underbrace{\sum_{m=0}^{\frac{N}{2}-1} x_{2m} e^{-\frac{2\pi i}{\frac{N}{2}} mk}}_{E_k} + e^{-\frac{2\pi 2}{N} k} \underbrace{\sum_{m=0}^{\frac{N}{2}-1} x_{2m-1} e^{-\frac{2\pi i}{\frac{N}{2}} mk}}_{O_k} = E_k + O_k e^{-\frac{2\pi i}{N} k} \tag{10}$$

Algoritmen utfører denne oppdelingen rekursivt til vi ender opp med ledd som bare inneholder ett datapunkt. Denne prosessen gir FFT en kompleksitet på $O(N \log_2(N))$[1] sammenlignet med $O(N^2)$ for en tradisjonell DFT. Den logaritmiske faktoren i basen 2 kommer av FFT-algoritmens rekursive oppdeling – datasettet halveres på hvert trinn, noe som gir logaritmisk vekst.

La oss se på en sammenligning av beregningene ved $N = 100$ datapunkter:

Ved 100 datapunkter kan man allerede se den store beregningsmessige forskjellen:

$$\frac{100 \cdot \log_2(100)}{100^2} \approx 0.066 \tag{11}$$

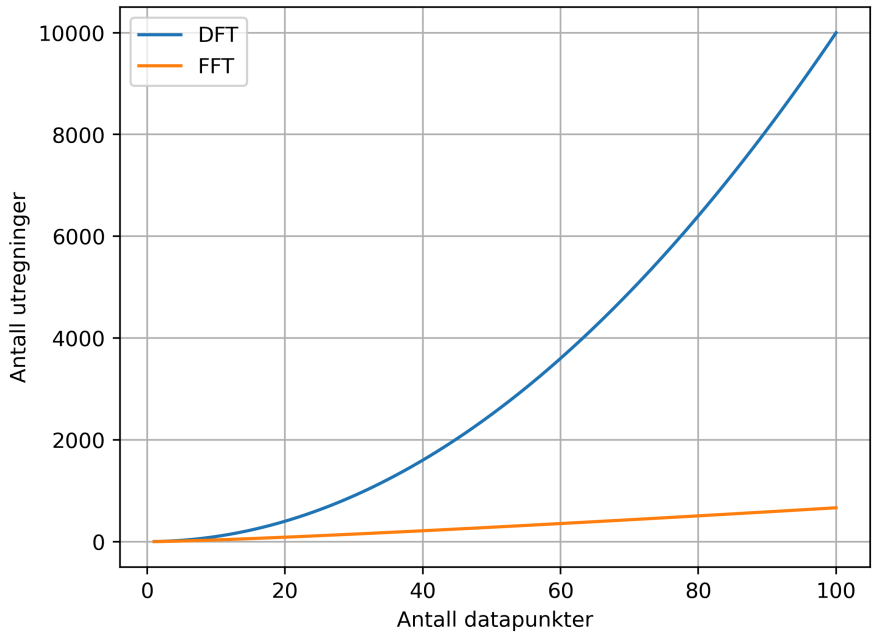
Med FFT trenger vi altså bare å utføre omtrent 6.6% av beregningene som kreves for en vanlig DFT med 100 datapunkter – en forskjell som blir enda større ved større datasett.

Til sammenligning har en CD en samplingsfrekvens på 44.1 kHz, noe som betyr at ett sekund med lyd består av 44100 datapunkter. Ved slike datamengder gir FFT en betydelig ytelsesfordel fremfor DFT.

$$\frac{44100 \cdot \log_2(44100)}{44100^2} \approx 0.00035 \tag{12}$$

Altså vil en FFT på dataen for ett sekund av en CD kreve 0.035% av utregningene som kreves av normal DFT.

[1] Antar at leser har forståelse for beskrivelse av tidskompleksitet gjennom $O(N)$



Figur 1: Antall utregninger som gjøres for DFT og FFT i forhold til hver verdi

Over, i Figur 1 vises en graf over hvor mange utregninger som trengs for å få DFT og FFT for 1 til 100 punkter. Grunnen til at figuren ikke viser flere punkter er at grafen for FFT ville sett ut som en flat linje i bunnen av figuren i forhold til DFT.

2.3 Short-time fouriertransformasjon (STFT)

STFT er en utvidelse av fouriertransformasjonen som lar oss se hvordan frekvensinnholdet i et signal endrer seg over tid. For å gjøre dette splittes signalet i små tidsvinduer, for så å utføre en fouriertransformasjon på hvert enkelt vindu. Ved å gjøre dette får vi se hvilke frekvenser som er i signalet i hvert tidsvindu.

2.3.1 Kontinuerlig STFT

Ved kontinuerlig tid STFT utføres fouriertransformasjonen på et kontinuerlig signal som deles inn i små tidsvinduer.

Formelen for Kontinuerlig STFT:

$$X(t, w) = \int_{-\infty}^{\infty} x(\tau) w(\tau - t) e^{-jw\tau} d\tau \tag{13}$$

$x(\tau)$ er det kontinuerlige inngangssignalet som analyseres.

$w(\tau - t)$ er vindusfunksjonen som bestemmer en liten tidsbit av signalet for analyse.

e^{jwt} er kjernen til fouriertransformasjonen og utfører denne på det begrensede signalet.

2.3.2 Diskret STFT

Ved diskret STFT analyseres signaler som er diskretisert for å få informasjon om frekvensinnholdet i forskjellige tidsintervaller. Diskretisert signaler (f.eks. digitale signaler som lydfiler) er signaler som er definert på bestemte punkter i tid. Når et signal diskretiseres tas målinger av signalet ved bestemte tidspunkter og representeres ved verdier. Diskret STFT brukes i flere praktiske

applikasjoner, som musikkgjenkjenning, taleprosessering og spektrumanalyse av digitale signaler. Musikkgjenkjenningsprogrammet Shazam er basert på diskret STFT.

Formelen for diskret STFT:

$$X(n, w) = \sum_{m=-\infty}^{\infty} x[m] \times w[m-n] \times e^{-jwm} \tag{14}$$

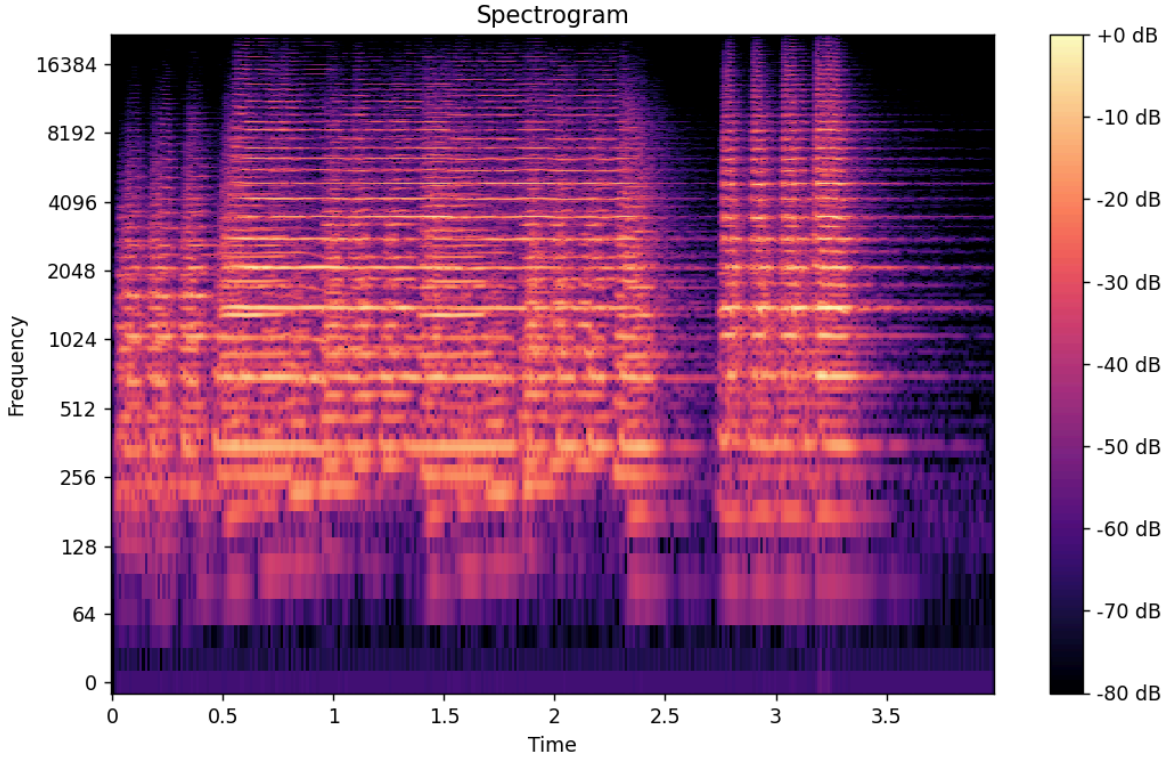
$x[n]$ er det diskrete inngangssignalet.

$w[m-n]$ er vindusfunksjonen som bestemmer et tidsvindu av signalet som skal analyseres. Størrelsen på dette vinduet vil påvirke oppløsningen i tid og frekvens. Et kort vindu vil gi god tidsoppløsning, men dårlig frekvensoppløsning, mens et langt vindu vil gi god frekvensoppløsning, men dårlig tidsoppløsning. Med dette menes hvor nøyaktig vi kan bestemme når en bestemt frekvens skjer og hvor nøyaktig vi kan bestemme hvilke frekvenser som er til stede i signalet.

e^{-jwm} er den komplekse eksponentielle funkjsonen som utfører fouriertransformasjonen på signalet.

$X(n, w)$ er resultatet av STFT og viser både tidsindeksen n og frekvenskomponenten w

For å utføre diskret STFT er det noen steg som må tas. Først velges et kort tidsvindu som skal analyseres, deretter forskyves tidsvinduet for å gjøre en ny diskret fouriertransformasjon. For hver gang tidsintervallet forskyves tas det en DFT for å finne frekvensinnholdet i vinduet. Til slutt samles alle DFT-resultatene for hvert tidsvindu og settes sammen til tids-frekvens-representasjon av signalet. Et eksempel på en slik representasjon er spektrogram. Spektrogrammet tar resultatene for hvert tidsvindu og viser signalet over tid med frekvens på vertikal akse, tid på horisontal akse, og intensiteten eller styrken av hver frekvens representeres med farge.



Figur 2:

2.4 Alternative metoder for Frekvensanalyse

I tillegg til Short-time fouriertransformasjon finnes det flere metoder for frekvensanalyse, som gir forskjellige fordeler og ulemper ved frekvensanalyse. To andre ofte brukte metoder er Wavelet-transformasjon og Empirical Mode Decomposition (EMD). Disse metodene har noen fordeler og ulemper som kan være mer verdifulle i andre situasjoner.

Wavelet-transformasjonen bruker en annen tilnærming enn STFT for å dele opp signaler i tid og frekvens. Mens STFT har et fast tidsvindu, tilpasser Wavelet-transformasjonen seg til frekvensene i signalet. Høyfrekvente deler av signalet analyseres med korte tidsvinduer, mens lavfrekvente deler får lengre vinduer. Dette gjør den godt egnet for signaler med raskt skiftende frekvenser.

Formelen for Wavelet-transformasjonen:

$$w_{x(a,b)} = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t)\psi^* \left(\frac{t-b}{a} \right) dt \tag{15}$$

$x(t)$ er selve signalet som analyseres. a og b justerer wavelet-funksjonen for å tilpasse seg høyfrekvente (korte vinduer) eller lavfrekvente (lange vinduer) deler av signalet. Wavelet transformasjonen er spesielt nyttig for analyse av musikk med slagverk hvor frekvensene endres raskt. Denne metoden kan være kompleks å implementere. [4]

EMD er en metode som automatisk deler opp signalet i flere frekvenslag, kalt Intrinsic Mode Functions (IMF). I stedet for å bruke faste tidsvinduer som i STFT eller wavelets, finner EMD naturlige frekvenser i signalet.

Formelen for EMD:

$$x(t) = \sum_{k=1}^{n} IMF_k(t) + r(t) \tag{16}$$

$IMF_k(t)$ er hver av de naturlige frekvensene som EMD finner i signalet. $r(t)$ er en rest som inneholder de laveste frekvensene etter at alle IMFer er funnet. EMD tilpasser seg signalets egen struktur, noe som gjør det nyttig ved analyse av komplekse naturlige lyder som tale, fuglesang og andre lyder fra naturen. Denne metoden kan være følsom for støy, krevende å implementere og tolke. [5]

2.5 Andre vindustyper

I frekvensanalyse, spesielt ved bruk av STFT har valget av vindusfunksjon stor påvirkning. Vindusfunksjon er en matematisk funksjon som brukes til å dele opp signalet i mindre biter for å bedre analysere hvordan frekvensen varierer over tid. Ulike vindusfunksjoner påvirker hvordan STFT presenterer frekvensinnholdet i et signal, spesielt i forhold til tidsoppløsning og frekvensoppløsning.

Hamming-vinduet er et vindu som gir en god balanse mellom tids- og frekvensoppløsning. Vinduet har lav sidelobestøy. Sidelobestøy er uønskede frekvenskomponenter som oppstår under analysen, noe som kan gjøre det vanskelig å skille frekvenser som ligger tett inntil hverandre. Dette vinduet er bra for signaler hvor det trengs en god balanse mellom de to oppløsningene, men det er ikke optimalt for svært presis frekvensanalyse, siden det er litt bredere i frekvensspekteret enn noen andre vinduer.

Formelen for Hamming-vinduet:

$$w[n] = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1} \right) \tag{17}$$

Hanning-vinduet ligner Hamming-vinduet, men i forhold til Hamming-vinduet som ikke går helt til null ved endene av vinduet, gjør Hanning-vinduet dette. Dette gjør vinduet spesielt godt egnet for jevn overgang mellom tidssegmenter. Hanning-vinduet gir god balanse mellom tidsoppløsning og frekvensoppløsning, samt lav sidelobestøy, men det gjør ingen av disse tingene like bra som Hamming-vinduet.

Formelen for Hanning-vinduet:

$$w[n] = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) \tag{18}$$

Det er dette vinduet vi har brukt gjennom vår frekvensanalyse.

Gaussiske vinduer følger formen til en Gauss-kurve. Vinduet har en justerbar tids- og frekvensoppløsning ved hjelp av parameteren σ. Gausiske vinduer har svært lav sidelobestøy, og er god for avansert frekvensanalyse, som musikk med raskt skiftende frekvenser.

Formelen for Gauss-vinduet:

$$w[n] = e^{-\frac{1}{2}\left(\frac{n-\frac{N}{2}}{\sigma\frac{N}{2}}\right)^2} \tag{19}$$

Blackman-vinduet har enda lavere sidelobestøy enn Hamming. Dette gjør at Blackman-vinduet er et godt valg for analyse hvor frekvensnøyaktighet er spesielt viktig. Vinduet gir dårligere tidsoppløsning sammenlignet med andre vinuer med smalere tidsvindu.

Formelen for Blackman-vinduet:

$$w[n] = 0.42 - 0.5\cos\left(\frac{2\pi n}{N-1}\right) + 0.08\cos\left(\frac{4\pi n}{N-1}\right) \tag{20}$$

Kaiser-vinduet er svært fleksibel siden det har en parameter β som kan justeres for å balansere mellom tidsoppløsning og frekvensoppløsning. Lavere verdier av β gir høyere tidsoppløsning, mens høyere verdier gir bedre frekvensoppløsning. Ulempen med dette vinduet er at det kan kreve mye eksperimentering for å velge riktig β-verdi.

Formelen for Kaiser-vinduet:

$$w[n] = \frac{I_0\left(\beta\sqrt{1 - \left(\frac{2n}{N-1} - 1\right)^2}\right)}{I_0(\beta)} \tag{21}$$

[6]

3 Metode

3.1 Datainnsamling og Forberedelse av Lydfiler

I dette prosjektet fokuserte vi på et kort tidsintervall fra starten av den populære sangen "Für Elise" som hovedeksempel for vår analyse. Dette spesifikke klippet ble valgt fordi det er gjenkjennelig og består av enkle, tydelige noter som spilles én om gangen. Denne tilnærmingen bidro til å holde prosjektet håndterbart og unngå kompleksitet, samtidig som det tillot oss å teste STFT-metodens evne til å identifisere noter i en enkel musikalsk kontekst.

Vi brukte en .wav-lydfil med en samplingsrate på 44,1 kHz, noe som sikret at analysen oppfylte kravene til korrekt representasjon og presisjon. Standardisering av samplingfrekvensen er avgjørende fordi Fourier-transformasjonen krever en korrekt representasjon av kontinuerlige signaler som diskrete samples. Når samplingfrekvensen er for lav, kan aliasing-effekter oppstå, der høyfrekvente komponenter ikke kan gjenkjennes korrekt. For å unngå aliasing sørget vi for å følge Nyquist-kriteriet, som sier at samplingfrekvensen må være minst dobbelt så høy som den høyeste frekvensen i signalet:
$f_s \geq 2f_{\max}$
hvor f_s er samplingsfrekvensen og $2f_{\max}$ er den høyeste frekvensen i signalet. Ved å standardisere på 44,1 kHz kan vi analysere frekvenser opp til omtrent 22,05 kHz, noe som dekker det meste av det hørbare spekteret. Denne samplingsfrekvensen gir tilstrekkelig presisjon i frekvensdomenet for musikk, slik at både lave og høye frekvenser kan analyseres korrekt uten risiko for aliasing, og resultatene fra STFT blir pålitelige og nøyaktige.

3.2 Implementering av STFT og FFT

For å utføre en detaljert analyse av lydsignalet implementerte vi både Short-Time Fourier Transform (STFT) og Fast Fourier Transform (FFT). STFT ble brukt for å dekomponere signalet i frekvenskomponenter over tid, noe som gir en dynamisk representasjon ved å inkludere en tidsdimensjon. Dette gjør det mulig å observere hvordan frekvensene i signalet varierer over tid, noe som er spesielt nyttig for analyse av musikk der frekvensene endres kontinuerlig.

Mens den vanlige Fourier-transformasjonen tar et signal $x(t)$ og konverterer det til et frekvensspektrum $X(f)$, benytter STFT en glidende vindusfunksjon $w(\tau - t)$ til å fokusere på korte tidsintervaller av signalet. Dette betyr at signalet blir delt opp i små tidsvinduer, og en Fourier-transformasjon blir utført på hvert av disse vinduene separat. På denne måten kan vi analysere frekvensinnholdet lokalt i tid, i motsetning til en global frekvensanalyse. Den matematiske formuleringen av STFT, både for kontinuerlig og diskret tid, er beskrevet i teoridelen (se seksjon 2.3.1 og 2.3.2).

I vår implementasjon leste vi inn lydsignalet ved hjelp av Python-biblioteket scipy.io.wavfile. Signalet ble deretter segmentert i små tidsvinduer med forhåndsdefinerte størrelser, og hver del ble analysert individuelt ved bruk av en valgt vindusfunksjon. For å utføre spektralanalyse brukte vi scipy.signal.stft, som muliggjør Short-Time Fourier Transform (STFT) for å dekomponere signalet i frekvenskomponenter over tid. FFT ble brukt i tillegg for å generere et globalt amplitudespektrum av hele signalet, noe som gir en helhetlig oversikt over hvilke frekvenser som er til stede i lydfilen, men uten tidsinformasjon.

Under vises koden som ble brukt:

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

```
from scipy.signal import stft
from scipy.io.wavfile import write

def plot_magnitude():
    plt.figure(figsize=(10, 6))
    plt.plot(frequencies[:n//2], np.abs(DFT_combined[:n//2]))
    plt.title('Amplitude Spekter av WAV filen')
    plt.xlabel('Frekvens (Hz)')
    plt.ylabel('Amplitude')
    plt.xlim(0, 4000)
    plt.grid()
    plt.show()

def plot_spectogram():
    plt.figure(figsize=(12, 6))
    plt.pcolormesh(times, frequencies, np.abs(Zxx), shading='gouraud')
    plt.colorbar(label='Styrke')
    plt.title('Spektrogram av WAV filen')
    plt.xlabel('Tid [s]')
    plt.ylabel('Frekvens [Hz]')
    plt.ylim(0, 4000)  # Limiter frekvensaksen til 0–4000 Hz
    plt.show()

def save_as_wav(filename, sample_rate, data):
    # Konverter data til riktig datatype (16-bit PCM)
    scaled_data = np.int16(data / np.max(np.abs(data)) * 32767)
    write(filename, sample_rate, scaled_data)

    save_as_wav("results\Med 3kHz).wav", sample_rate, combined_data)


def main():
    # Load the .wav file
    sample_rate, data = wavfile.read("data\Til elise.wav")

    # Om audioen er stereo (2 kanaler) velger vi kunn 1 kanal for enkelhet
    if len(data.shape) == 2:
       data = data[:, 0]

    # Number of samples in the audio
    n = len(data)

    # Compute the DFT using numpy
    DFT_data = np.fft.fft(data)

    # Frequency vector
    frequencies = np.fft.fftfreq(n, 1/sample_rate)

    # Compute the STFT
    frequencies, times, Zxx = stft(data, fs=sample_rate, window='hann',
nperseg=1024, noverlap=512)

    plot_spectogram()
```

```
if __name__ == "__main__":
    main()
```

For diskrete signaler brukte vi den diskret-tid STFT-metoden som beskrevet i teoridelen (se formel 9 i seksjon 2.3.2). Denne metoden gjør det mulig å beregne et lokalt frekvensspektrum for hvert vindu, noe som gir informasjon om hvilke frekvenser som er tilstede i det aktuelle tidsintervallet.

Ved å bruke STFT på denne måten, kunne vi lage et spektrogram som visualiserer frekvensinnholdet over tid. Spektrogrammet viser tid langs x-aksen, frekvens langs y-aksen, og fargeintensiteten representerer amplituden, altså mengden til de ulike frekvensene. Denne fremgangsmåten gjør det mulig å observere hvordan musikalske noter dukker opp, varer, og eventuelt overlapper hverandre i musikklippet.

Gjennom denne tilnærmingen klarte vi å dekomponere lydsignalet på en måte som gjorde det mulig å analysere energifordelingen over frekvensene gjennom hele musikkklippet, noe som ga et detaljert bilde av klippets frekvensmønstre over tid.

3.3 Matematisk Eksperimentering med Vindusparametere

Basert på teorien om kontinuerlig- og diskret-tid STFT, utførte vi flere eksperimenter for å undersøke hvordan valg av vindusfunksjon og størrelsen på $w(\tau - t)$, påvirker nøyaktigheten i noteidentifikasjonen. Eksperimentene startet med korte tidsvinduer for å oppnå høy tidsoppløsning, noe som ga detaljert informasjon om raske frekvensendringer. Deretter ble lengre tidsvinduer testet for å oppnå bedre presisjon i frekvensoppløsningen, som er nødvendig for å identifisere nærliggende frekvenser med høy nøyaktighet.

Vi evaluerte flere vindustyper, inkludert Hanning- og Gaussiske vinduer, for å optimalisere balansen mellom tids- og frekvensoppløsning. Hanning-vinduet reduserte spektral lekkasje og ga en jevn overgang mellom vinduer, noe som forbedret presisjonen i tids- og frekvensdomenet. Gaussisk vindu ble også testet for sin evne til å gi smale spektrale topper, som øker frekvenspresisjonen, men på bekostning av tidsoppløsningen. Justeringen av vindusparametere ble basert på formel 9 fra seksjon 2.3.2, som beskriver diskret-tid STFT. Gjennom disse eksperimentene kunne vi justere hvordan spektrogrammet representerte frekvensene i lydklippet og oppnå optimal presisjon for identifikasjon av noter.

Etter evalueringen av ulike vindustyper, valgte vi å bruke Hanning-vinduet for den endelige analysen. Dette valget ble tatt fordi Hanning-vinduet ga den beste balansen mellom tids- og frekvensoppløsning, samtidig som det reduserte spektral lekkasje og ga en jevn overgang mellom segmentene. Dette er avgjørende for musikkanalyse, der presisjon i både tids- og frekvensdomenet er viktig for å identifisere noter og akkorder. Gaussisk vindu hadde noen fordeler når det gjaldt smale spektrale topper og økt frekvenspresisjon, men det viste seg å gi dårligere tidsoppløsning, noe som gjorde det vanskeligere å identifisere raske endringer i musikken. Derfor valgte vi å bruke Hanning-vinduet for å sikre en helhetlig og presis representasjon i spektrogrammet.

3.4 Analyse av Polyfoniske Signaler

Et sentralt mål i prosjektet var å evaluere STFTs evne til å analysere polyfoniske signaler, der flere noter spilles samtidig. Dette utgjør en matematisk utfordring fordi det krever at STFT effektivt skiller mellom flere overlappende frekvenser i spektrogrammet, særlig når frekvensene er nær hverandre.

Når flere noter spilles samtidig, dannes en kompleks sum av sinuskurver med ulike frekvenser. For å analysere slike sammensatte signaler, anvender STFT Fourier-transformasjonen innenfor hvert

tidsvindu, noe som gir en funksjon av både tid og frekvens: $X[m, k]$. Denne prosessen er detaljert beskrevet i teoridelen (se seksjon 2.3.2).

Denne metoden gjorde det mulig å identifisere de dominerende frekvenskomponentene som tilhører ulike noter, selv når de ble spilt samtidig. Gjennom denne tilnærmingen kunne vi observere hvordan hver frekvens opptrådte og utviklet seg over tid, og evaluere STFTs nøyaktighet i adskillelsen av individuelle noter i polyfoniske signaler.

3.5 Tonemanipulasjon

For å øve oss på å modulere signalet og analysenene til prøven vil en kjent frekvens bli lagt til det opprinnelige lydsignalet for å undersøke hvordan frekvensspesifikke komponenter påvirker signalets amplitudespekter og spektrogram. Denne frekvensen forventes å fremstå som en markant amplitude i amplitudespekteret, og et notch-filter vil bli implementert med et bånd som dekker små avvik rundt frekvensen. Valget av dette båndet er ment å sikre at hele frekvenskomponenten blir fjernet effektivt, samtidig som tilgrensende frekvenser forblir uendret. Filtreringsprosessen vil dermed kunne verifisere at det ønskede frekvensområdet fjernes presist og uten å påvirke andre komponenter i signalet.

3.6 Noteanalyse

I oppgaven ønsket vi å analysere hvilke noter som blir spilt i et musikkstykke ved å identifisere deres respektive frekvenser. Hver musikalsk note har en spesifikk frekvens, og ved å sammenligne disse frekvensene med en tabell over kjente notefrekvenser, kan vi bestemme hvilke noter som er til stede i et lydsignal. Ved å bruke STFT kan vi ikke bare identifisere hvilke frekvenser som opptrer, men også kartlegge når notene blir spilt og hvor lenge de varer.

Gjennom denne tilnærmingen kunne vi se hvordan enkelttoner og akkorder utvikler seg over tid i musikkstykket. STFT gir oss en tid-frekvens-representasjon som muliggjør en detaljert analyse av noteoverganger og rytmiske mønstre. Dette er spesielt nyttig for å forstå strukturen i komplekse musikalske passasjer og for å analysere polyfoniske signaler der flere noter spilles samtidig. Resultatet er et spektrogram som visuelt viser både hvilke noter som spilles, tidspunktet for når de spilles, og varigheten av hver note.

Oktav	C	C# / Db	D	D# / Eb	E	F	F# / Gb	G	G# / Ab	A	A# / Bb	B
C2	65.41	69.30	73.42	77.78	82.41	87.31	92.50	98.00	103.83	110.00	116.54	123.47
C3	130.81	138.59	146.83	155.56	164.81	174.61	185.00	196.00	207.65	220.00	233.08	246.94
C4	261.63	277.18	293.66	311.13	329.63	349.23	369.99	392.00	415.30	440.00	466.16	493.88
C5	523.25	554.37	587.33	622.25	659.26	698.46	739.99	783.99	830.61	880.00	932.33	987.77
C6	1046.50	1108.73	1174.66	1244.51	1318.51	1396.91	1479.98	1567.98	1661.22	1760.00	1864.66	1975.53
C7	2093.00	2217.46	2349.32	2489.02	2637.02	2793.83	2959.96	3135.96	3322.44	3520.00	3729.31	3951.07
C8	4186.01	4434.92	4698.63	4978.03	5274.04	5587.65	5919.91	6271.93	6644.88	7040.00	7458.62	7902.13

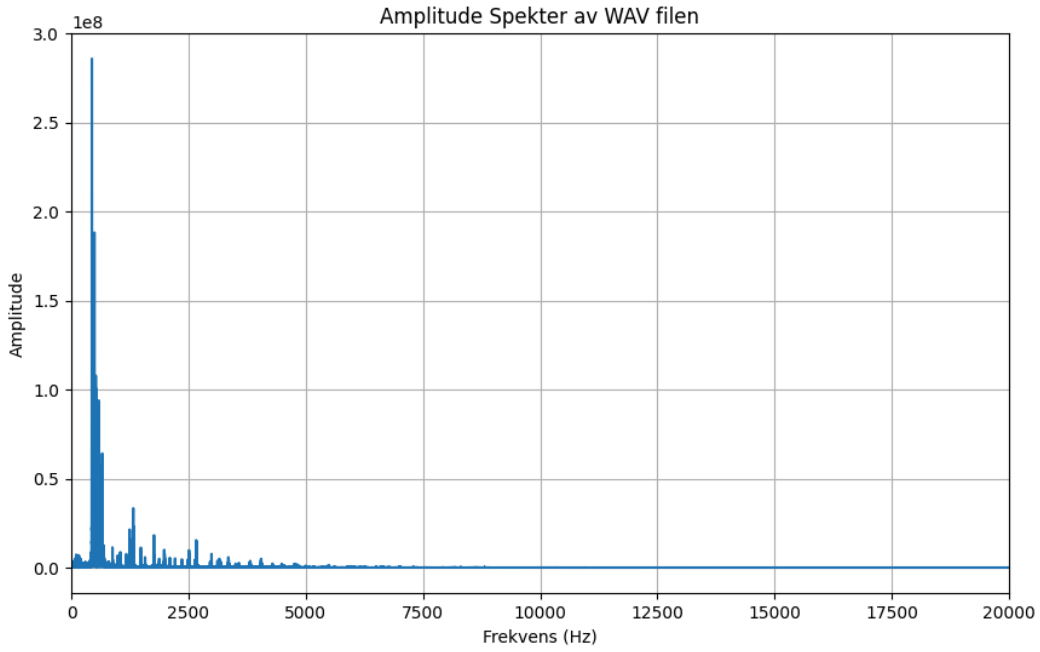
Figur 3: Tabell over noter og respektiv frekvens i Hz

For å identifisere hvilken note som spilles, kan man analysere amplitudespekteret og se hvilke frekvenser som har betydelig amplitude. Ved å identifisere disse frekvensene og sammenligne dem med et notekart, kan man bestemme hvilke noter som er til stede i lydsignalet. Dette gir en presis metode for å matche frekvensene fra analysen med de tilsvarende musikalske notene.

4 Resultater

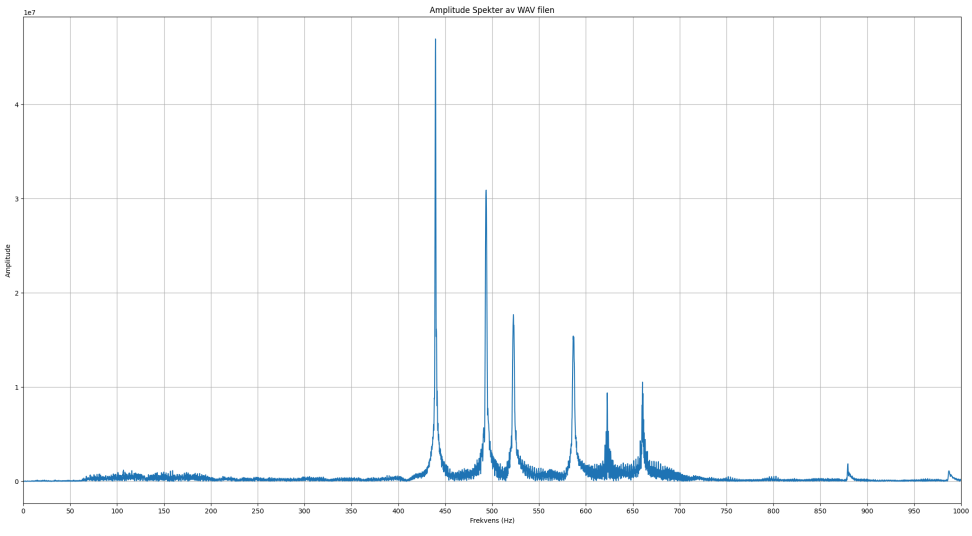
4.1 Initial Amplitudespektrumanalyse

Vi startet analysen ved å beregne Fourier-transformasjonen (FFT) av lydsignalet for å undersøke frekvensinnholdet. Amplitudespekteret ble begrenset til et område fra 0 til 20 kHz, da dette er innenfor det hørbare området for mennesker.



Figur 4: Amplitude spekter av lydfil, fra 0-20kHz

Etter en nærmere inspeksjon ble området videre avgrenset til 0–1 kHz for en mer detaljert visualisering og tolkning. Dette området gir en mer tydelig fremstilling av de lavere frekvensene som er av størst interesse for analyse av musikalske noter.

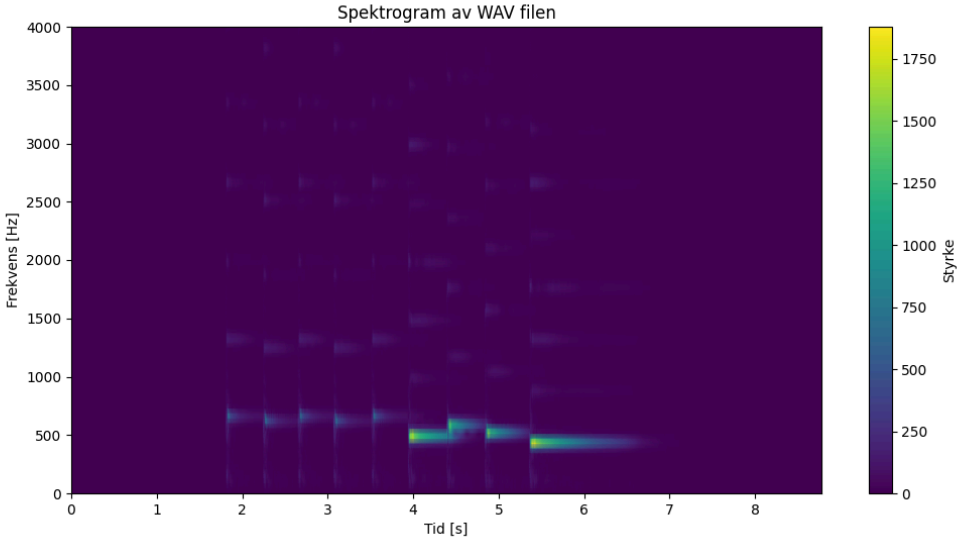


Figur 5: Amplitudespekter avgrenset til 1000 Hz

Fra amplitudespekteret observerte vi at følgende noter ble spilt: A4(440Hz), B4(493.88Hz), C5(523.25), D5(587.33), D#5(622.25), E5(659.26Hz). Disse funnene samsvarer med de faktiske tonene som blir spilt på pianoet i klippet. Dette bekrefter at analysen vår er nøyaktig, og at FFT og STFT effektivt kan brukes til å identifisere frekvensinnholdet i musikk og kartlegge de spesifikke notene som spilles.

4.2 Generering av Spektrogram

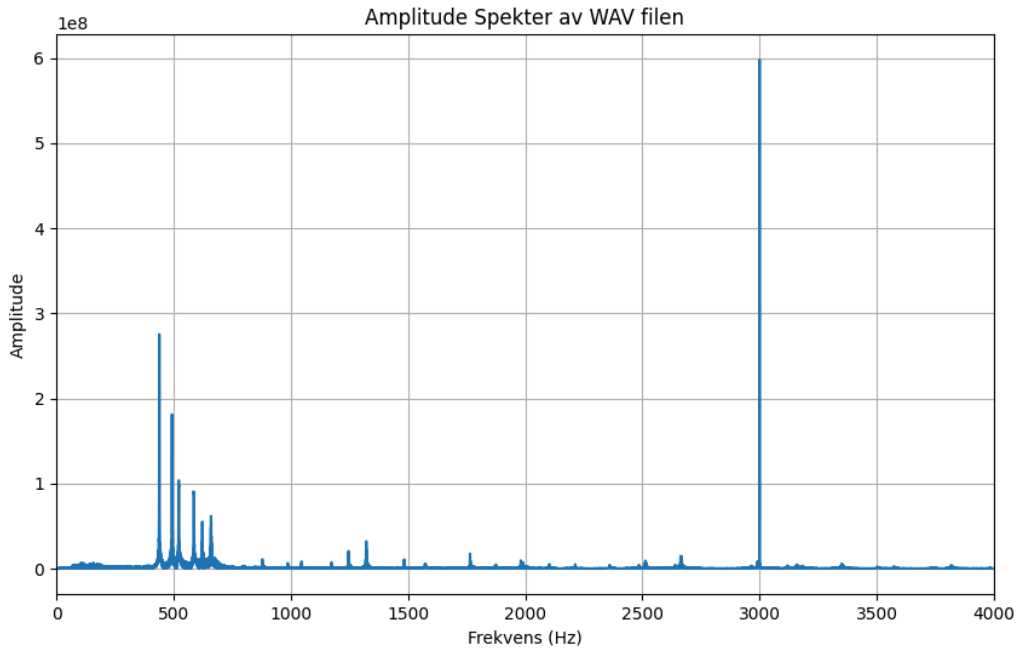
For å analysere signalet over tid, brukte vi Short-Time Fourier Transform (STFT) for å generere et spektrogram. Spektrogrammet ble avgrenset til et intervall på 0–4 kHz. Dette gjorde det mulig å observere hvordan frekvensinnholdet i signalet varierte over tid, noe som er viktig for gjenkjenning av noter og identifisering av spesifikke lydmønstre.



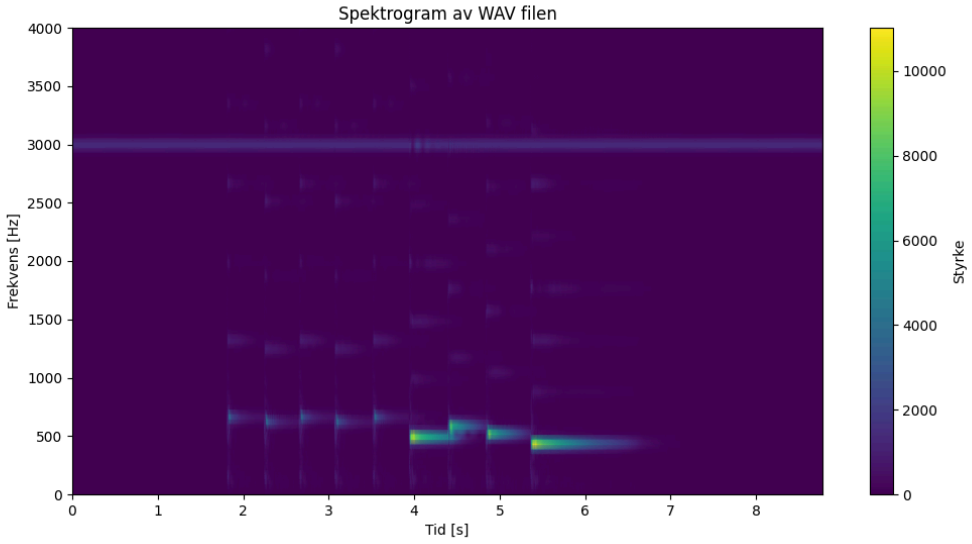
Figur 6: Spektogram av lydfilen, avgrenset til 4 kHz

4.3 Introduksjon av en 3000 Hz Tone

For å teste signalmanipulasjon og verifisere korrektheten av filtreringsprosessen, ble det lagt til en ren sinus-tone på 3000 Hz til det eksisterende lydsignalet. Denne prosessen ble utført for å undersøke hvordan endringer i frekvensspekteret påvirker amplitudespekteret og spektrogrammet.



Figur 7: Amplitude spekter med 3kHz tone



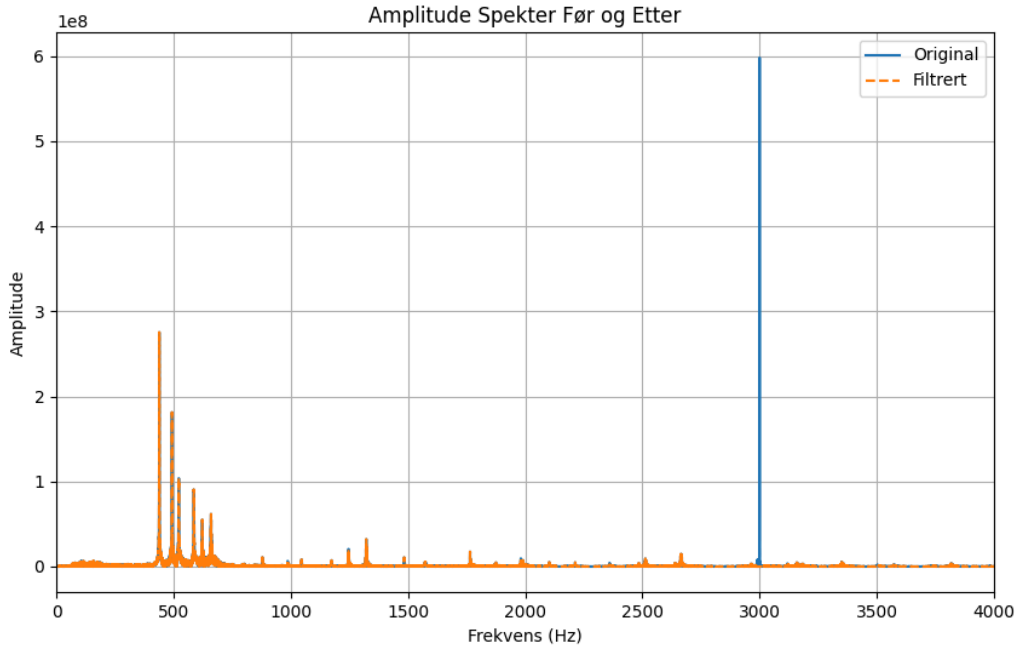
Figur 8:

4.4 Filtrering av 3000 Hz Tone

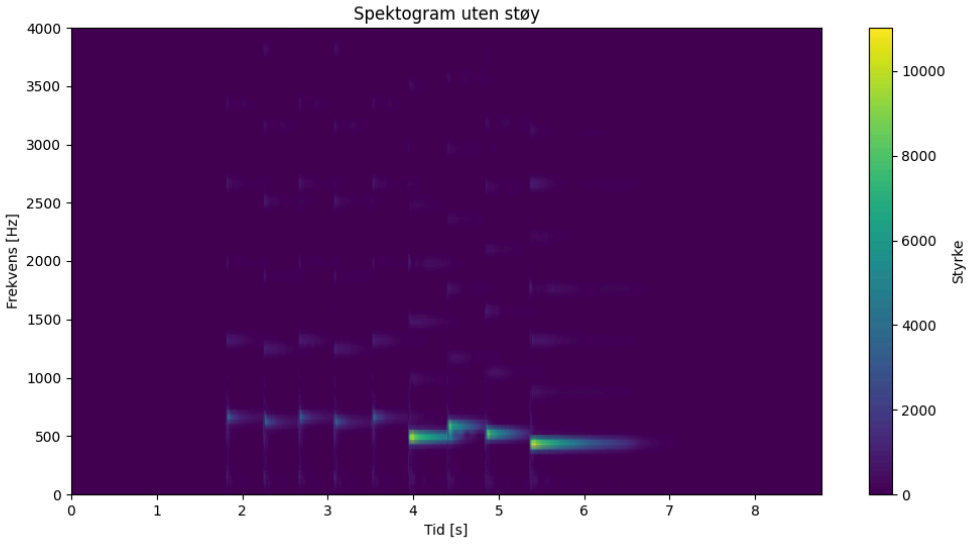
Etter å ha lagt til tonen, identifiserte vi den i amplitudespekteret som en betydelig amplitude ved 3000 Hz. Vi implementerte deretter et filter for å fjerne frekvensområdet 2985–3015 Hz for å sikre at hele 3000 Hz-signalet ble eliminert. Dette båndet ble valgt for å inkludere små variasjoner rundt frekvensen.

4.5 Rekonstruksjon av Signal

For å få det filtrerte signalet tilbake til tidsdomenet, ble inverse FFT (IFFT) brukt. Denne prosessen ga oss et rekonstruert lydsignal uten 3000 Hz-tonen, som vi deretter kunne analysere og lytte til for å verifisere suksessen til filtreringen.



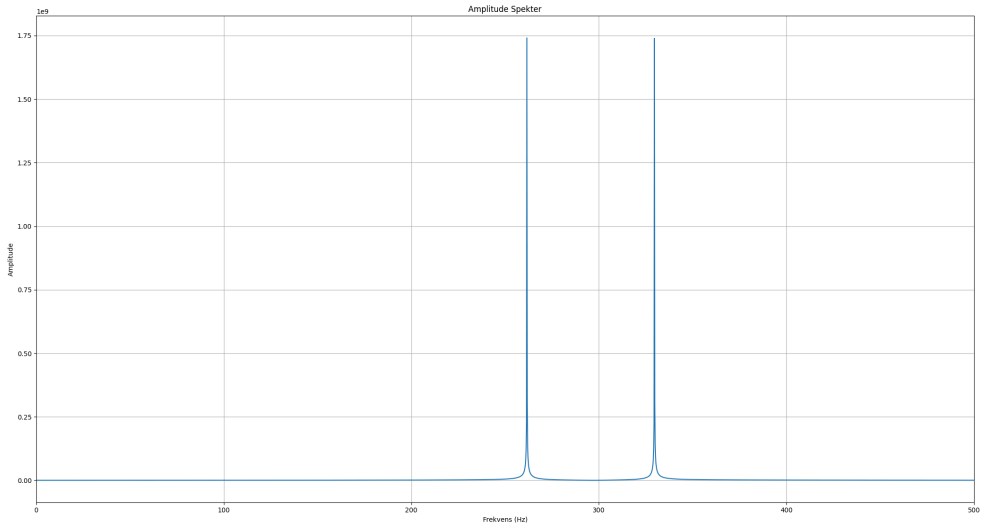
Figur 9: Før og etter filtrering av signal



Figur 10: Spektogram av filtrert lydfil

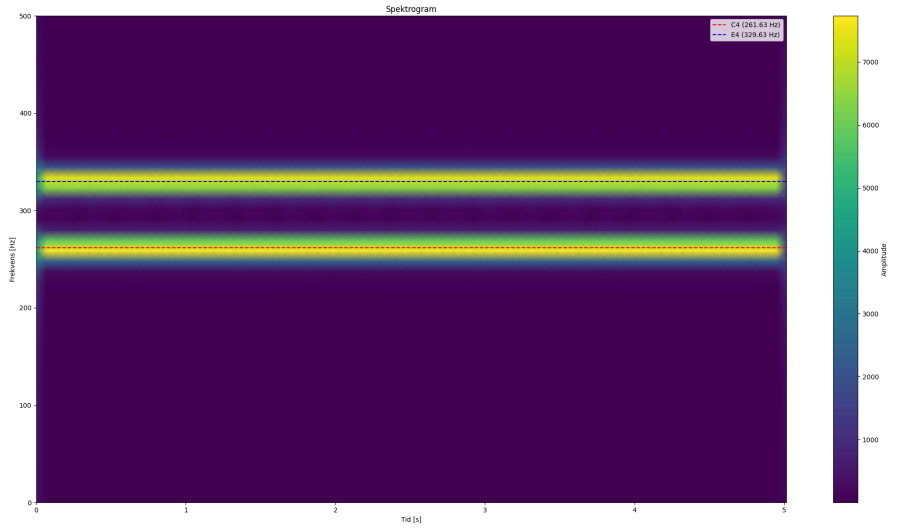
4.6 Analyse av To Noter i Signal

Vi genererte et lydsignal bestående av to rene sinusnoter: C4 (261.63 Hz) og E4 (329.63 Hz). For å analysere frekvensinnholdet i signalet, benyttet vi Fourier-transformasjonen (FFT) til å lage et amplitudespekter. Resultatene fra FFT viste tydelige frekvenstopper ved de forventede notefrekvensene, noe som bekreftet at signalet inneholdt de to notene vi hadde generert.



Figur 11: Amplitude spekter med to like sterke signal

Vi brukte også STFT for å lage et spektrogram som viste hvordan de to notene oppførte seg over tid. Spektrogrammet gjorde det mulig å observere at notene var jevnt til stede gjennom hele varigheten av signalet, og de viste stabile frekvenser som samsvarte med C4 og E4. Denne analysen ga en tydelig visualisering av de individuelle komponentene i signalet og hvordan de ble representert i frekvensdomenet over tid.



Figur 12: Spektogram av to signaler over tid

5 Diskusjon

5.1 Valg av metode for Frekvensanalyse

Hovedproblemstillingen i dette prosjektet var: *Hvordan kan STFT og FFT brukes til å identifisere og modifisere frekvenser i lydsignaler fra et kjent musikkstykke, og hvilke faktorer påvirker nøyaktigheten av noteidentifikasjonen?* For å svare på dette valgte vi å bruke både Fast Fourier Transform (FFT) og Short-Time Fourier Transform (STFT). FFT ble brukt for å generere amplitudespekteret, som ga oss en rask og oversiktlig måte å identifisere hvilke noter som var til stede i musikkstykket. Dette gjorde det enklere å få en helhetlig forståelse av hvilke frekvenser som var til stede i lydsignalet.

STFT ble brukt som en komplementær metode til FFT for å gi informasjon om når og hvor lenge de ulike frekvensene (notene) ble spilt i klippet. Siden STFT inkluderer en tidsdimensjon i analysen, gjorde dette det mulig å observere hvordan frekvensinnholdet i musikken varierte over tid. Dette var spesielt nyttig for vårt formål med Für Elise, hvor noter og akkorder endrer seg dynamisk.

STFT ble valgt fremfor alternativer som wavelet-transformasjon og Empirical Mode Decomposition (EMD). Wavelet-transformasjon, som gir variabel tidsoppløsning basert på frekvens, kunne vært nyttig for signaler med raske frekvensendringer, som slagverk, mens EMD er mer egnet for naturlige og komplekse lyder. STFT var imidlertid det beste valget for dette prosjektet, da musikkstykket har relativt stabile frekvenser som passer godt med en fast vindusstørrelse. Denne tilnærmingen ga oss dermed en presis og effektiv metode for å både identifisere og visualisere noter i musikkstykket, ved å kombinere styrkene til både FFT og STFT.

5.2 Valg av Vindusfunksjon og Påvirkning på Resultater

Den første underproblemstillingen vår var: *Hvordan påvirker valg av vindustype og -størrelse nøyaktigheten i noteidentifikasjonen ved bruk av STFT?* Gjennom eksperimenter med ulike vindusfunksjoner fant vi at Hanning-vinduet ga en optimal balanse mellom tids- og frekvensoppløsning. Fordi Hanning-vinduet går til null ved endene, reduseres sidelobestøy, og det gir jevnere overganger mellom segmentene, noe som forbedrer nøyaktigheten i 5gb. Denne funksjonen bidro til å redusere sidelobestøy, som kan forårsake forstyrrelser i analysen, og gav oss et sammenhengende bilde av frekvensinnholdet over tid.

Selv om Hanning-vinduet fungerer godt for vår analyse, kunne vi også vurdert andre vinduer, som Blackman-vinduet, som gir høyere frekvenspresisjon, eller Gaussiske vinduer, som gir fleksibilitet i tids- og frekvensoppløsning. For fremtidige analyser, særlig av mer komplekse musikkstykker med tettliggende noter, kan det være nyttig å utforske slike vinduer for å øke nøyaktigheten i noteidentifikasjonen.

5.3 Signalmanipulering og Filtreringsresultater

Den andre underproblemstillingen var: *Hvordan kan en uønsket tone fjernes fra et lydsignal ved hjelp av DFT?* For å teste dette la vi inn en ekstra 3000 Hz tone i lydsignalet. Deretter brukte vi et notch-filter for å fjerne denne spesifikke frekvensen. Dette viste hvordan STFT og DFT kan brukes til å isolere og eliminere uønskede frekvenser uten å påvirke resten av signalet, noe som kan være svært nyttig i lydredigering og for å forbedre kvaliteten i opptak.

Filtreringen fungerte som forventet: 3000 Hz-tonen ble effektivt fjernet, mens andre nærliggende frekvenser forble upåvirket. Dette eksperimentet viste tydelig at det er mulig å fjerne uønskede lyder presist ved bruk av frekvensfiltrering, noe som er relevant i mange praktiske situasjoner, som fjerning av støy fra opptak eller redigering av musikkfiler.

5.4 Presisjon og Begrensninger i Polyfonisk Analyse

Den siste underproblemstillingen vi tok opp var: *Hvordan presterer STFT når det gjelder å identifisere noter fra et ekte musikkstykke som inneholder komplekse lyder og overlappende frekvenser?* Vi fokuserte på starten av Für Elise for å unngå kompleksiteten i partier med mange overlappende noter og varierende styrker. STFT ga gode resultater for enkle noter, men utfordringer oppstod ved analyser av akkorder der frekvensene overlappet. Den faste vindusstørrelsen i STFT gjør det vanskelig å skille tettliggende frekvenser, noe som reduserer presisjonen ved komplekse signaler.

Vi klarte imidlertid å analysere et enklere polyfonisk signal med to rene noter (C4 og E4), som viste at STFT kan brukes til å identifisere overlappende frekvenser i enklere tilfeller. Dette ga oss en bedre forståelse av metodens styrker og begrensninger. For fremtidige analyser kan teknikker som wavelet-transformasjon vurderes for bedre tilpasning til komplekse musikkstykker.

5.5 Implikasjoner og Forslag til Videre Forskning

Gjennom dette prosjektet har vi vist at STFT er en effektiv metode for noteidentifikasjon i musikk, men at den har begrensninger ved analyse av polyfoniske signaler og tettliggende frekvenser. For fremtidige studier kan det være nyttig å utforske alternative metoder, som wavelet-transformasjon, som gir variabel oppløsning og kan håndtere komplekse musikkstykker med raske frekvensskift. Videre kan eksperimenter med ulike vindusfunksjoner og finjustering av STFT-parametere bidra til å øke nøyaktigheten og gjøre metoden mer robust for et bredere spekter av musikalske analyser.

6 Konklusjon

Prosjektet har demonstrert hvordan Short-Time Fourier Transform (STFT) kan brukes til å identifisere og modifisere frekvenser i lydsignaler, med fokus på noteidentifikasjon i musikkstykket Für Elise. Gjennom STFT-analysen har vi vist at det er mulig å oppnå en god balanse mellom tids- og frekvensoppløsning ved nøye valg av vindusparametere, som med Hanning-vinduet som ga jevne overganger og redusert støy. Dette muliggjorde en presis identifikasjon av noter i et dynamisk musikalsk signal. Eksperimentet vi gjennomførte med filtrering av en 3000 Hz tone viste oss også hvordan frekvensfiltrering kan isolere og eliminere uønskede frekvenser uten å påvirke resten av signalet i noen større grad.

Gjennom dette arbeidet har vi fått erfart at STFT er et kraftig verktøy samt også noen av bruksområdene, men vi har også erfart at STFT har enkelte begrensninger når det gjelder komplekse polyfoniske signaler. Mens STFT identifiserte dominerende frekvenser, var det utfordrende å skille tettliggende frekvenser, noe som kan være et hinder ved analyse av musikk med mange overlappende noter.

Gjennom denne oppgaven får vi innsikt i hvordan STFT kan benyttes i musikkteknologi for å bland annet identifisere noter, men også innenfor lydredigering for støyfjerning samt kvalitetssikring av lydfiler. Metoden gir et grunnlag for praktiske anvendelser innenfor digital musikkbehandling, lydanalyse og utvikling av verktøy for automatisert noteidentifikasjon i musikkprogramvare.

For videre forskning er det et potensial til å undersøke alternative metoder som kan tilpasse seg raskt skiftende frekvenser, slik som wavelet-transformasjon. Videre eksperimentering med ulike vindusfunksjoner og parametere kan samt bidra til å finjustere STFT for ulike musikktyper og for å oppnå enda høyere presisjon i polyfoniske analyser. Dette vil bidra til å utvide bruksområdene til frekvensanalyse i musikk og lyd, og styrke forståelsen av digitale metoder i signalbehandling.

Referanser

[1] “Fourier transform.” [Online]. Available: https://en.wikipedia.org/wiki/Fourier_transform

[2] “Discrete Fourier transform.” [Online]. Available: https://en.wikipedia.org/wiki/Discrete_Fourier_transform

[3] “Fast Fourier transform.” [Online]. Available: https://en.wikipedia.org/wiki/Fast_Fourier_transform

[4] “Wavelet transform.” [Online]. Available: https://en.wikipedia.org/wiki/Wavelet_transform

[5] “Multidimensional empirical mode decomposition.” [Online]. Available: https://en.wikipedia.org/wiki/Multidimensional_empirical_mode_decomposition

[6] “Window function.” [Online]. Available: https://en.wikipedia.org/wiki/Window_function