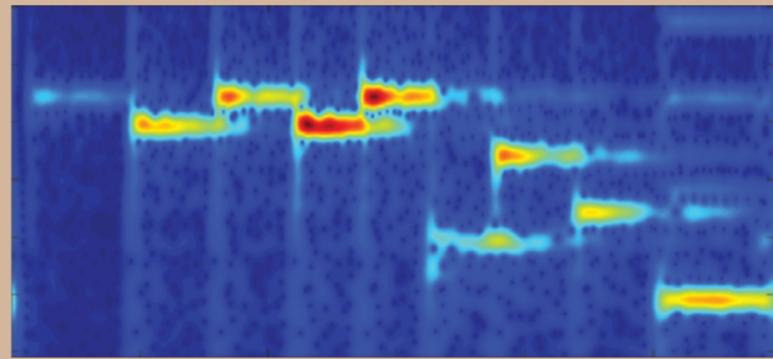
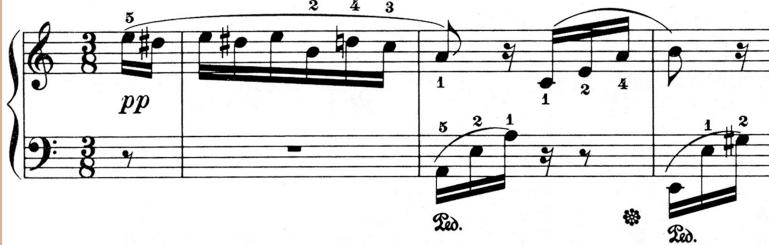


Für Elise

Ludwig van Beethoven
(1770–1817)

Poco moto



14. Wavelettransformasjon

Dette kapitlet tar opp følgende temaer: Tidsoppløst fouriertransanalyse, fourier- vs wavelet-analyse, Morlet wavelets, frekvensoppløsning vs tidsoppløsning, optimalisering, randproblem. Lesetips finnes på websiden med hjelpefiler til boka.

14.1 Tidsoppløst frekvensanalyse

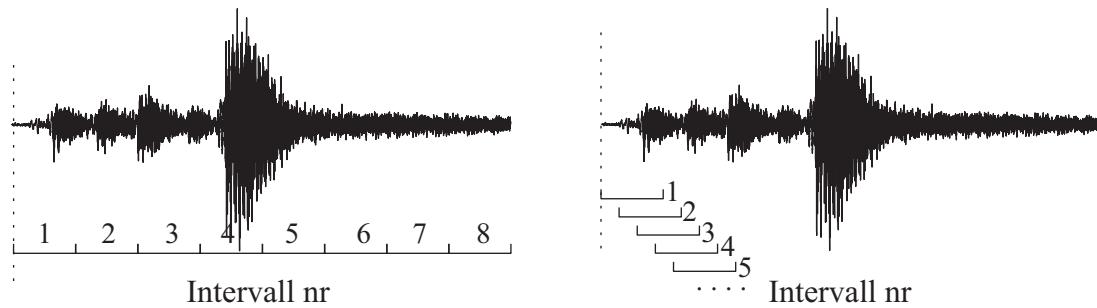
Fouriertransformasjon slik vi har omtalt den i kapittel 5, egner seg meget bra for “stasjonære” signaler (endrer ikke nevneverdig karakter underveis). For signaler som endrer seg over tid, blir tidsinformasjonen fordelt over alle frekvenskomponentene, og det er håpløst å finne ut hvordan frekvensspekteret varierer med tiden. En FFT av et signal som varierer mye fra en tidsperiode til en annen innenfor samme datastrengen er derfor lite verdt.

Det finnes flere ulike metoder for å utforske hvordan frekvensspekteret endrer seg med tiden. I frekvensanalysatorer og ved analyse av signaler som varer ved svært lenge (svært mange datapunkter samlet) brukes en såkalt “kort-tids-fouriertransformasjon” eller stykkevis fouriertransformasjon (se figur 14.1). På engelsk brukes termene “short-time FT” eller “short-term FT” og forkortes STFT. Matematisk kan STFT angis slik:

$$\text{STFT}\{x(t)\}(\tau, \omega) \equiv \mathcal{X}(\tau, \omega) = \int_{-\infty}^{\infty} x(t)w(t - \tau)e^{-i\omega t} dt$$

Her er $w(t - \tau)$ en såkalt vindus-funksjon som har en klokkelignende form. I praksis brukes gjerne såkalt Hanning- eller Gauss-form, som har en betydelig verdi i en begrenset tidsperiode nær referansetiden τ .

Vindusfunksjonen w synker til null utenfor den begrensede tidsperioden slik at vi i praksis bare foretar analyse av et begrenset tidsrom av den totale datamengden. I praksis brukes en fast fourier transform (FFT) for analyse av hvert enkelt tidsvindu. Det utvelgelsen av en lokalisert bit av datamengden ved hjelp av vindusfunksjonen w som gir oss tidsoppløsningen. Ved en smal vindusfunksjon w får vi i prinsippet en høy tidsoppløsning, og ved en bred vindusfunksjon en dårligere tidsoppløsning.



Figur 14.1: For å få tidsinformasjon ved analyse av en lang tidsstrek med data, kan vi stykke opp det totale tidsintervallet og foreta fouriertransformasjon for intervall etter intervall. Intervallene kan velges slik at de ikke overlapper hverandre (venstre del) eller slik at de overlapper hverandre (høyre del).

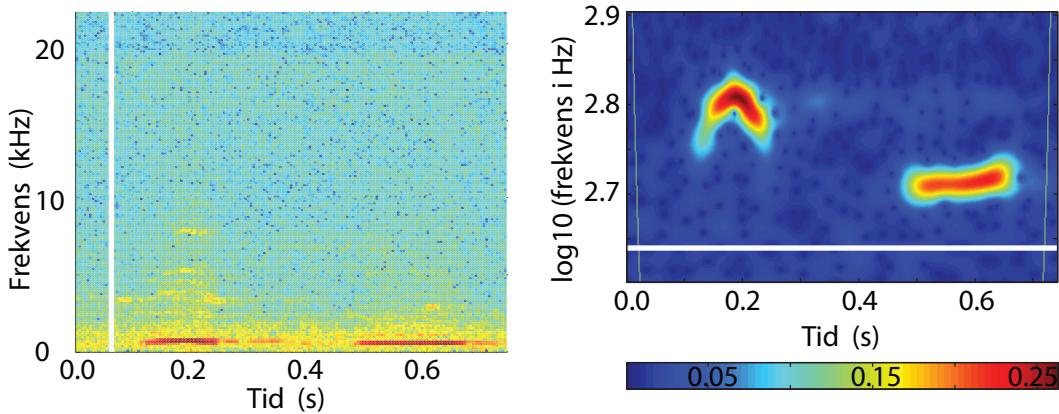
I analysen lar vi τ skli gjennom hele datamaterialet vi ønsker å analysere. Vi kan ofte velge om den neste vindusfunksjonen skal overlappe den forrige eller ikke, og i så fall hvor stort overlappet skal være. Resultatet vises gjerne i et diagram der intensiteten på fourierkomponentene i hvert enkelt tidsvindu plottes som funksjon av tiden. Intensiteten angis gjerne med fargekoding. Resultatet kalles gjerne for et “spektrogram” (se venstre del av figur 14.2).

Fordelen med denne metoden er at vi kan analysere kontinuerlige signaler i ukesvis om vi vil, uten avbrekk. Det er ingen begrensning på datastrekengens lengde siden vi i praksis bare plukker ut en begrenset bit for hver analyse som gjøres.

Ulempen er at vi får en frekvensoppløsning som er omvendt proporsjonal med tiden som analyseres i hvert vindu (dvs hvor lenge tidsvinduet varer). Det betyr at vi får nøyaktig samme frekvensoppløsning (og derved også tidsoppløsning) enten vi analyserer signaler med lav frekvens eller med høy frekvens. Vi må velge bredden på vindusfunksjonen for en eller annen typisk frekvens i signalet. Dersom det imidlertid finnes svært forskjellige frekvenser samtidig i signalet, er det umulig å finne en optimal vinduslengde som passer for alt.

Dette er en viktig detalj for STFT og for alle tidsoppløste frekvensanalyser. På grunn av båndbredde-teoremet vi var innom da vi behandlet fouriertransformasjon, er det umulig å få svært presis informasjon om tid og frekvens samtidig. Bruker vi et vindu w som strekker seg over lang tid, kan vi i prinsippet få temmelig presis frekvensinformasjon. Vi kan imidlertid ikke samtidig få presis informasjon om endringer med tiden. Det er den klassiske analogien til Heisenbergs uskarphetsrelasjon som på ny dukker opp.

Ved et “glidende filter”-metode som i STFT der vinduet skiftes ett datapunkt om gangen før ny analyse, unngår vi hopp i resultatene som skyldes tilfeldigheter i hvordan intervallene velges. Problemet er imidlertid at vi må gjennomføre til dels mange tilsvarende unødvendige beregninger. Vi får med andre ord ganske mye overflødige data (engelsk: “redundancy”) i resultatene. Dette kan unngås ved å foreta hopp i plasseringen av vindusfunksjonen fra en analyse til en annen. Effektiviteten går da opp uten nevneverdig tap av informasjon (dersom vi vet hva vi gjør!), men hva som er passe hopplengde vil avhenge av hvilken frekvens vi analyserer. Ved STFT-funksjonen i Matlab kan vi velge både bredde og grad av overlapp for vindusfunksjonen w , men det er av flere grunner vanskelig å optimalisere valget ved bruk av STFT.



Figur 14.2: Spektrogrammer av lyden fra en gjøk som synger 'ko-ko'. Til venstre er et resultat av Matlabs innebygde STFT-funksjon, her basert på *fft* av 300 punkter om gangen og et såkalt Hammingvinduet er på 400 punkter. Overlapp fra en analyse til den neste er 200 punkter. Den vertikale hvite stripene markerer at dette spektrogrammet bygges opp linje for linje med vertikale linjer. Til høyre er det vist et spektrogram beregnet med wavelettransformasjon med Morlet wavelets, med dataprogrammet gitt senere i dette kapitlet. Parametre brukt gis senere i kapitlet. Horizontal hvit stripe indikerer at et slikt diagram bygges opp linje for linje med horisontale linjer. Se tekst for øvrige detaljer.

I dette kapitlet skal vi ta for oss en annen metode som kan gi tidsoppløst frekvensinformasjon, nemlig såkalt "kontinuerlig waveletanalyse med Morlet wavelets". Vi ender opp med et diagram som viser hvordan frekvensbildet endrer seg med tiden, akkurat som vi ofte gjør ved STFT. Det er imidlertid store forskjeller i hvordan analysen foretas matematisk (se høyre del av figur 14.2).

I STFT bruker vi FFT for å analysere hver bit av tidssignalet (plukket ut ved bruk av vinduet w), og vi skifter så til neste bit av tidssignalet. Vi bygger derfor opp fourierkoeffisient vs frekvens og tiddiagrammet ("STFT-spektrogrammet") stripe for stripe med *vertikale* stiper. Vi får med oss alle frekvenser fra null til halve samplingsfrekvensen enten vi er interessert i hele dette området eller ikke.

Ved wavelet-analyse får vi et "wavelet-spektrogram", og dette diagrammet kan i visse sammenhenger se temmelig likt ut som et STFT-diagram. Ved wavelet-analyse bygger vi også opp diagrammet linje for linje, men nå med *horisontale* linjer (se hvite linjer i figur 14.2). Vi må derfor velge hele tidsintervallet vi ønsker å analysert før analysen starter, noe som i visse sammenhenger er en ulempe. Fordelen er imidlertid at vi selv kan velge hvilket frekvensområde vil studere (se figur 14.2). Vi kan videre selv velge en logaritmisk frekvensakse dersom vi ønsker det for å utnytte fullt ut at den relative frekvensoppløsningen ved denne metoden er den samme for alle frekvenser.

"Kontinuerlig waveletanalyse" ligner på en glidende, kort-tids fouriertransformasjon (STFT), men waveletanalysen med Morlet wavelets gir samme relative frekvensoppløsning for alle frekvensene. Hemmeligheten er å bruke ulik lengde på tidsstrengeen alt etter hvilken frekvens vi analyserer.

Det kan nevnes at det også finnes mange varianter av wavelettransformasjon hvor vi foretar så få transformasjoner som overhodet mulig med bare et ganske begrenset tap av informasjon. En slik transformasjon er mye mer effektiv enn den kontinuerlige, og brukes i teknologiske sammenhenger der det er viktig at ting går fort. Ulempen med en slik wavelettransformasjon er at det transformerte signalet er langt vanskeligere å forstå enn et vanlig fourierspekter. Det er hovedgrunnen til at vi ikke går inn på den metoden her.

Waveletanalyse er et omfattende fagfelt innen matematikk/informatikk, og det gis egne kurs om emnet ved mange universiteter. Vi kommer ikke til å gå i detalj om den strengt matematiske eller datatekniske siden av wavelets. Hensikten med å ta med wavelets i denne boka, er å gjøre oppmerksom på at fouriertransformasjon ofte egner seg *dårlig* for ikke-stasjonære signaler, og samtidig peke på en analysemetode som ofte er langt å foretrekke i slike sammenhenger. Dessuten kan arbeid med wavelets bidra til en dypere forståelse av tidsavgrensede fenomener generelt og de tilsvarende frekvenser. Blant annet er det nære analogier mellom Heisenbergs uskarphetsrelasjon og waveletanalyse.

En del av dere vil nok bruke waveletanalyse i masteroppgaven eller i et evt. PhD-prosjekt (og senere jobber). Av den grunn legger vi vekt på å vise hvor waveletanalyse er nyttig og når metoden ikke har mye å gi. Wavelets brukes bl.a. for å analysere solflekkaktivitet (og endringer i syklusen med tiden), El Niño sørlige oscillasjoner i Stillehavet, isbre-sykler, ruhet, kornstørrelseanalyser, analyse av f.eks. kreftceller vs normale celler og mye mer.

Teknologisk er det en omfattende bruk av wavelets bl.a. i jpeg-komprimering av bildefiler, og i mp3-komprimering av lyd.

14.2 Kort historikk

La oss repetere litt historien bak fouriertransformasjon: Den franske matematikeren Joseph Fourier (1768-1830) "oppdaget" fouriertransformasjon for vel 200 år siden. (Fourier arbeidet først med varmestrømning, og var visstnok den første som oppdaget drivhuseffekten.)

Fouriertransformasjon benyttes i stor grad i analytisk matematikk. I tillegg fikk transformasjonen en enorm utbredelse i dataverdenen etter at J.W.Cooley og J.W.Tukey i 1965 oppdaget den såkalte "Fast Fourier Transform" (FFT) som gjør det mulig å foreta en fouriertransformasjon svært mye raskere enn tidligere. Ved FFT benyttes symmetriene i sinus og cosinusfunksjonene for å redusere antall multiplikasjoner ved utregningen, men for å få den mest effektive transformasjonen, må antall datapunkter være en heltalls potens av 2, dvs $N = 2^n$.

Det hevdet at Cooley-Tukeys Fast Fourier Transform egentlig ble oppdaget av Carl Friedrich Gauss ca 1805, men glemt og delvis gjennoppfunnet flere ganger før 1965. Suksessen til Cooley og Tukeys gjenoppdagning skyldes nok at datamaskinen gjorde sitt inntog omrent på denne tiden.

Waveletanalyse er av langt yngre dato. Riktignok ble wavelets introdusert allerede ca 1909, men metoden ble først tatt i bruk fra ca 1980 av. Det er langt større spillerom for spesielle varianter av waveletanalyse enn ved fouriertransformasjon. Det er både en fordel og ulempe. Vi kan langt på vei skreddersy en waveletanalyse slik at den passer optimalt for de dataene vi vil analysere. Ulempen er at den store variasjonsmuligheten medfører at vi må bruke hodet litt mer ved waveletanalyse enn ved fouriertransformasjon, både når transformasjonen skal gjennomføres, og når vi tolker resultatene. Men resultatene blir ofte desto mer interessante!

14.3 Kort om matematikken bak

14.3.1 Oppfrisking av fouriertransformasjon

Vi har gått gjennom fouriertransformasjon i kapittel 5, men la oss repetere de matematiske uttrykkene også her.

La $x(t)$ være en integrerbar funksjon av tid. Vi kan da beregne en ny funksjon $X(\omega)$, hvor ω er vinkelfrekvens, på følgende måte:

$$X(\omega) = \frac{1}{2} \int_{-\infty}^{\infty} x(t) e^{-i\omega t} dt \quad (14.1)$$

Det morsomme med denne funksjonen er at vi kan ta en tilsvarende ”omvendt” transformasjon:

$$x(t) = \int_{-\infty}^{\infty} X(\omega) e^{i\omega t} d\omega \quad (14.2)$$

og end opp med eksakt den opprinnelige funksjonen igjen. Merk fortegnskiftet i den komplekse eksponensialfunksjonen.

Fra likningene (14.1) og (14.2) ser vi at når $x(t)$ er reell, vil $X(\omega)$ være kompleks. Dette er nødvendig for at $X(\omega)$ både skal kunne angi hvor kraftig svingning vi har ved ulike frekvenser, og i tillegg angi innbyrdes fase til de ulike frekvenskomponentene. (En symmetri i X medfører at x etter den omvendte transformasjonen blir reell, slik den var opprinnelig.)

Det bør forøvrig nevnes at x og X generelt sett ikke behøver å være funksjoner av tid og frekvens. Det finnes mange ulike typer funksjoner og sammenhenger hvor fouriertransformasjon anvendes. I vårt sammenheng begrenser vi oss imidlertid (nesten utelukkende) til tid og frekvens.

Uttrykkene ovenfor anvendes ved analytiske beregninger. Når vi bruker datamaskin, kjenner vi ikke fullstendig til hvordan $x(t)$ varierer i tid. Vi kjenner bare verdien av x i diskrete tidspunkt t_n . I disse tidspunktene har x verdiene x_n hvor n er en indeks som varierer fra 1 til N , dersom x er beskrevet i N tidspunkt. Vi antar at det er valgt ekvidistante tidspunkt slik at det er en fast tid mellom to nærliggende tidspunkt. Total tid x er beskrevet over er da $T = N * \delta t$ hvor δt er tiden mellom to tidspunkt i beskrivelsen (detaljer diskutert i et tidligere kapittel).

Når fouriertransformasjon gjennomføres på diskrete data, brukes en diskret transformasjon. Denne kan angis slik:

$$X_k = \frac{1}{N} \sum_{n=1}^N x_n e^{-i2\pi f_k t_n} \quad (14.3)$$

hvor $k = 0, 1, 2, \dots, N - 1$. Videre er $f_k = 0, f_s/N, 2f_s/N, \dots, f_s(N - 1)/N$ der f_s er samplingsfrekvensen. Endelig er $t_n = 0, T/N, 2T/N, \dots, T(N - 1)/N$ der $N/T = f_s$.

Det er kanskje ikke så lett å se ut av uttrykket, men det vi egentlig gjør for å bestemme den fouriertransformerte for en frekvens f_k , er å multiplisere (ledd for ledd) den digitaliserte funksjonen x_n med en cosinusfunksjon med frekvensen f_k og summere alle leddene som

da framkommer. (For imaginær delen av den fouriertransformerte multipliserer vi med en sinusfunksjon med frekvensen f_k .)

Den tilsvarende ”omvendte” transformasjonen er da:

$$x_n = \sum_{k=1}^N X_k e^{i2\pi f_k t_n} \quad (14.4)$$

for $n = 1, 2, 3, \dots, N$.

14.3.2 Formalisme ved wavelettransformasjon

Wavelettransformasjon kan angis tilsvarende på nokså analog måte som en fouriertransformasjon:

La $x(t)$ være en integrerbar funksjon av tid. Vi kan da beregne en ny funksjon $\gamma_K(\omega_a, t)$ som gir informasjon om frekvenser og tid samtidig.

ω_a kan betegnes som ”analyse-vinkelfrekvens”. K er en ”skarphets”-parameter (også kalt ”bølgetallet”, se siden) knyttet til hvorvidt vi ønsker å ha høy presisjon i tidsangivelser (K liten) eller høy presisjon i frekvensangivelser (K stor).

Enkeltverdier for den nye wavelettransformerte funksjonen kan finnes på følgende måte:

$$\gamma_K(\omega_a, t) = \int_{-\infty}^{\infty} x(t + \tau) \Psi_{\omega_a, K}^*(\tau) d\tau \quad (14.5)$$

Her er $\Psi_{\omega_a, K}(\tau)$ selve waveleten. Asterikken sier at det er den kompleks konjugerte av uttrykket for waveleten vi må bruke.

Det spesielle med waveletanalyse er at vi kan velge mellom nærmest uendelig mange forskjellige wavelets alt etter hva vi ønsker å få fram i analysen. I vår sammenheng kommer vi bare til å bruke såkalt Morlet wavelets. Matematisk kan den reelle delen av en Morlet wavelet uttrykkes som en *cosinusfunksjon* (pluss et litt konstant korreksjonsledd) konvolutert med en *gaussisk funksjon* (”gaussisk omhyllingskurve”). Den imaginære delen er en *sinusfunksjon* konvolutert med samme gaussisk funksjon som i stad (se figur 14.3).

En Morlet-wavelet kan beskrives som:

$$\Psi_{\omega_a, K}(\tau) = C \{ \exp(-i\omega_a \tau) - \exp(-K^2) \} \cdot \exp(-\omega_a^2 \tau^2 / (2K)^2) \quad (14.6)$$

hvor C er en ”normeringskonstant”. Når vi beskrver $\Psi_{\omega_a, K}(\tau)$ numerisk, kan vi med fordel bruke følgende uttrykk for C :

$$C = \frac{0.798}{f_s K} \omega_a \quad (14.7)$$

hvor f_s er samplingsfrekvensen.

[♣ ⇒ Noen kommentarer:

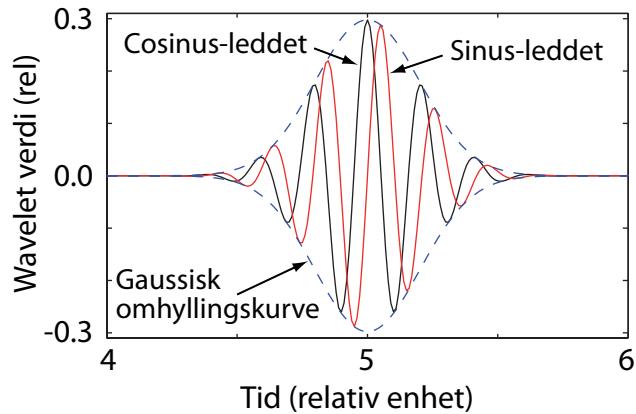
Det har ikke satt seg en ensartet beskrivelse av wavelets ennå. Forskjellige kilder angir formalismen på ulike måter, der blant annet uttrykk så som “scaling-parameter”, “Mother-” og “daughter-wavelets” er sentrale begreper. Vi har valgt å bruke en fremstilling som ligger nær opp til en artikkel av Najmi og Sadowsky (se litteraturlisten) fordi denne ligger temmelig nær opp til øvrig formalisme i denne boka. “Konstanten” C har jeg imidlertid valgt ut fra prøving og feiling ut fra ønsket om at en wavelettransformasjon av et rent sinussignal skal gi amplituden til sinussignalet uansett valg av parametrene ω_a , f_s og K (avvik fra det perfekte er som oftest mindre enn ca 1 %). Det aktuelle uttrykket for en Morlet wavelet kommer vi ikke til å bruke i praksis, bortsett fra et illustrativt eksempel. Ved effektiv wavelettransformasjon kommer vi til å ta utgangspunkt i den fouriertransformerte av waveleten, og beskriver den direkte. Detaljer gis i teksten som følger. ← ♠]

I waveletanalysen sjekker vi om signalet vi studerer inneholder ulike frevenser ved ulike tider. Vinkelfrekvensen vi analyserer er ω_a . Parameteren τ angir tiden der en spesifikk wavelet har et maksimum, og svarer til senter for det lille tidsintervallet vi undersøker.

Parameteren K er en reell konstant og kan kalles “bredden” på waveleten. Noen kaller den “bølgetallet”, fordi den angir omtrentlig antall bølger som det er plass til under den gaussiske omhyllingskurven for waveleten (omhyllingskurven er gitt i siste ledd i ligning (14.6)). Det anbefales at K er 6.0 eller større.

På grunn av det midtre leddet i ligning (14.6) ser vi at waveleten Ψ er kompleks.

Figur 14.3 viser et eksempel på en Morlet wavelet. Vi ser at den bærer navnet med rette: “wavelet” kan nemlig oversettes med “liten bølge”. Vær sikker på at du gjennomskuer hvordan waveleten dannes, nemlig som en gaussisk konvoluttering av en kompleks harmonisk funksjon sentrert rundt tiden τ .



Figur 14.3: Eksempel på en Morlet wavelet for $K = 6$. Både den reelle delen (cosinus-ledd) og den imaginære delen (sinus-ledd) er gitt.

Merk at uttrykket i ligning (14.6) er en generell beskrivelse. Når denne skal implementeres i et datamaskinprogram, og analysere et konkret signal, må vi kjenne til samplingsfrekvensen som ble brukt. Denne kommer inn i normeringskonstanten C . Dersom det konkrete signalet er beskrevet i N ekvidistante punkter i tid, er den totale tiden samlingen har foregått lik $T = N/f_s$. Vi velger da å beskrive enhver Morlet-wavelet vi bruker i analysen ved hjelp av en array med samme samplingsfrekvens og samme lengde på arrayen som det konkrete signalet

vi skal analysere.

Dersom vi sammenligner ligning (14.1) med ligning (14.5), ser vi at uttrykkene ligner mye på hverandre. Vi integrerer opp produktet av en funksjon x og en bølge. Begge er derved knyttet til et ”indreprodukt” innen matematikken, men som sagt, vi skal bare touche matematikken med en harelabb her.

Det er imidlertid flere forskjeller enn vi først skulle tro. En vesentlig forskjell ligger i at wavelettransformasjonen fører til en tredimensjonal beskrivelse (verdien av γ som funksjon av både ω_a og t), mens en beskrivelse basert på fouriertransformasjon bare er todimensjonal (verdien av X som funksjon av frekvens).

Også for en wavelettransformasjon er det mulig å foreta en ”omvendt” transformasjon. Det er essensielt når wavelets brukes i jpeg bildekompresjon og mp3 musikkfil-komprimering. Vi tar imidlertid ikke med detaljer angående denne formalismen i vår sammenheng. (Interesserte henvises til siste referanse i litteraturlisten i slutten av kapitlet.)

14.3.3 ”Diskret kontinuerlig” wavelettransformasjon

Først litt om bruk av ordene ”diskret” og ”kontinuerlig”. Et digitalisert signal vil vi kalle diskret, fordi vi bare har et endelig antall måleresultater (ekvidistante i tid). Vi vil likevel betegne den spesielle wavelettransformasjonen som beskrives i dette kapitlet, som ”kontinuerlig”, i betydning at det ”glidende filteret” (jmf høyre del av fig 14.1) bare forskyves ett punkt fram i det digitaliserte signalet hver gang en ny beregning gjennomføres. Et alternativ ville være å forskyve waveleten med f.eks. halve waveletbredden.

Wavelettransformasjon brukes nesten utelukkende på diskrete signaler, siden beregningene er så omfattende at de nesten er umulige å gjennomføre analytisk (unntatt i svært enkle modell-beskrivelser).

For digitaliserte signaler (diskrete signaler) kan selve Morlet waveleten skrives som:

$$\Psi_{\omega_a, K, t_k}(t_n) = C \{ \exp(-i\omega_a(t_n - t_k)) - \exp(-K^2) \} \cdot \exp(-\omega_a^2(t_n - t_k)^2 / (2K)^2) \quad (14.8)$$

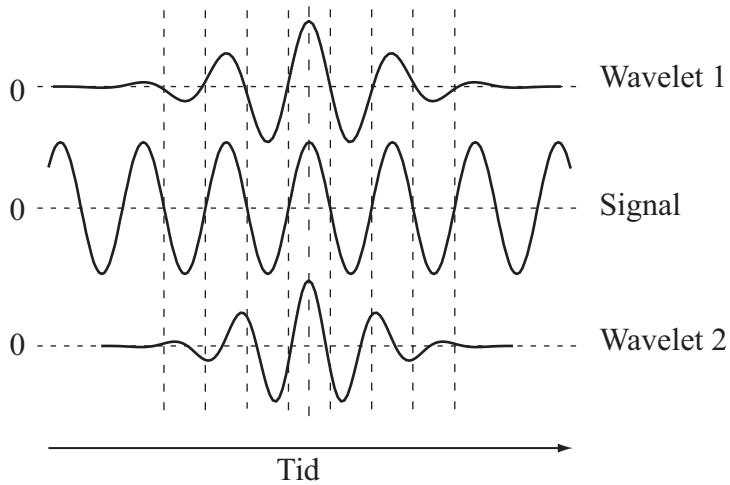
Her er det antatt at signalet vi skal analysere er beskrevet i ekvidistante punkter ved hjelp av tallrekken x_n for $n = 1 \dots N$. Tiden t_k angir *midtpunktet til waveleten* (!).

Selve wavelettransformasjonen for én frekvens og ett analysetidspunkt vil da være:

$$\gamma_K(\omega_a, t_k) = \sum_{n=1}^N x_n \Psi_{\omega_a, K, t_k}^*(t_n) \quad (14.9)$$

La oss bildeliggjøre prosessen litt for at vi skal bedre forstå hva den går ut på. I figur 14.4 viser vi et utsnitt av en tidsstreng sammen med to ulike valg av wavelets. Wavelettransformasjon består i å multiplisere signalet med waveleten, tidspunkt etter tidspunkt, og danne en sum av produktene. Resultatet er den wavelettransformerte av signalet for akurat den frekvensen waveleten representerer og for akkurat det tidspunktet i signalet hvor waveleten har sin maksimale verdi.

For wavelet 1 ser vi at signalet skifter fortegn omrent på samme sted som waveleten skifter fortegn. Det vil si at produktet i ethvert punkt blir positivt, og summen av produkter blir derfor ganske stor (svarer til at $\int \cos^2(\omega t) dt$ er positiv). For wavelet 2 skifter signalet fortegn på andre steder enn for waveleten. Noen av produktene blir derved positive og noen



Figur 14.4: Et sinusformet signal (i midten) sammen med en wavelet med samme periodetid (øverst) og en wavelet med noe kortere periodetid (nederst).

negative. Summen av produkter blir betydelig lavere enn for det første tilfellet (svarer til at $\int \cos(\omega_1 t) \cos(\omega_2 t) dt$ ofte er nær null når $\omega_1 \neq \omega_2$).

Vi har derved forsøkt å anskueliggjøre at wavelettransformasjonen av en regulær sinusbølge vil ha et maksimum når “bølgelengden” til waveleten svarer til “bølgelengden” til signalet i det tidsintervallet hvor vi foretar analysen.

For å analysere signalet x_n for andre bølgelengder, trenger vi å endre waveleten, og det gjøres blant annet ved hjelp av parameteren ω_a .

14.3.4 En mye mer effektiv algoritme

Wavelettransformasjon definert i ligning (14.5) er det vi i matematikken kaller en konvoluttering av tidssignalet $x(t)$ med waveleten $\Psi_{\omega_a, K}^*$. Det såkalte konvolutterings-teoremet er da av interesse for oss, fordi teoremet sier at den fouriertransformerte til en konvoluttering er lik prikkproduktet (punktvis produkt av to funksjoner) av den fouriertransformerte av hver av de to funksjonene som inngår.

La oss betegne den fouriertransformerte av x og den fouriertransformerte av waveleten Ψ med henholdsvis $\mathcal{F}(x)$ og $\mathcal{F}(\Psi)$.¹ La oss også betegne den fouriertransformerte av konvolusjonen $x * \Psi$ for $\mathcal{F}(x * \Psi)$. Da sier konvolutteringsteoremet at

$$\mathcal{F}(x * \Psi) = \mathcal{F}(x) \cdot \mathcal{F}(\Psi) \quad (14.10)$$

der høyresiden er punktvis multiplikasjon av de to fouriertransformerte. Men da kan vi foreta en invers fouriertransformasjon \mathcal{F}^{-1} av høyre og venstre side av denne ligningen, og får:

$$\mathcal{F}^{-1}(\mathcal{F}(x * \Psi)) = x * \Psi = \mathcal{F}^{-1}(\mathcal{F}(x) \cdot \mathcal{F}(\Psi)) \quad (14.11)$$

Den fouriertransformerte av selve signalet, $\mathcal{F}(x)$ kan lett beregnes, og kan brukes uforandret i resten av waveletanalysen. Fouriertransformasjonen av selve waveleten Ψ må i prinsippet

¹Vi skriver Ψ i stedet for Ψ^* i uttrykkene nedenfor for å gjøre uttrykkene lettere å lese. Må bruke korrekte størrelser ved implementering av metoden.

gjennomføres på ny hver gang vi endrer analysefrekvens eller bølgetall. Vi har imidlertid et analytisk uttrykk for den fouriertransformerte av en Morlet wavelet (se nedenfor), noe som gjør beregningen betydelig raskere.

Når vi så tar en invers fouriertransformasjon av produktet $\mathcal{F}(x) \cdot \mathcal{F}(\Psi)$ får vi hele konvolusjonen $x * \Psi$ i en eneste jafs. Det vil si at vi får hele tidsvariasjonen i det konvoluterte signalet for den valgte analysefrekvensen (og K -verdi) på én gang.

For å få en full waveletanalyse må vi så gjennomføre prosedyren på en rekke analysefrekvenser (som vi i prinsippet velger selv). Dersom vi velger f.eks. å gjøre analysen ved 1000 frekvenser, betyr det at beregningene tar om lag 1000 ganger lengre tid enn en enkel fouriertransformasjon. Så selv om metoden basert på konvolutteringsteoremet er svært effektiv sammenlignet med brute force- metoden, tar beregning av kontinuerlig wavelettransfomasjon med Morlet wavelets lang regnetid.

La oss nå se på den fouriertransformerte til en Morlet wavelet definert i ligning (14.6). Morlet waveleten er kompleks, og det morsomme er at dersom vi gjennomfører en fft på denne komplekse funksjonen, får vi et rent reelt resultat. Videre er det faktisk heller ingen speiling i frekvensspekteret!

Den fouriertransformerte til en Morlet wavelet (ligning (14.8)) kan gis på følgende måte:

$$\mathcal{F}(\Psi) \equiv \hat{\Psi}_{\omega_a, K}(\omega) = 2\{\exp(-[K(\omega - \omega_a)/\omega_a]^2) - \exp(-K^2)\exp(-[K\omega/\omega_a]^2)\} \quad (14.12)$$

Vi ser at dette er en klokkeformet (gaussisk) funksjon (bortsett fra et ganske ubetydelig korreksjonsledd for de fleste valg av K). Toppunktet i klokkefunksjonen finnes ved analysefrekvensen.

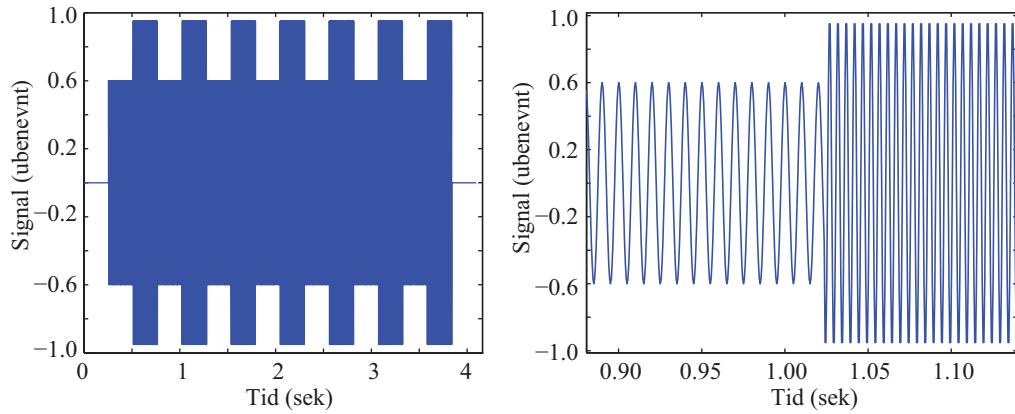
14.4 Eksempel

Vi skal nå vise i praksis et eksempel på wavelettransfomasjon, og starter med å generere et signal som funksjon av tid.

Signalet vi genererer skifter mellom 100 og 200 Hz med faste intervaller (se figur 14.5). De ytterste bitene av signalet settes lik null. Merk at når vi genererer et signal med variabel frekvens i løpet av den tiden signalet eksisterer, må vi sørge for å unngå brudd i signalet akkurat i det tidspunktet frekvensen endrer seg. Vi får en god beskrivelse dersom vi tar utgangspunkt i *fasen* til signalet til enhver tid, og oppgraderer fasen for hvert nytt trinn i tidsbeskrivelsen. Denne detaljen i signalet er demonstrert i detaljutsnittet til høyre i figur 14.5.

Vi implementerer så wavelettransfomasjonen direkte ut fra likningene (14.9) og (14.8), eller vi kan bruke den mer effektive metoden beskrevet ved likningene (14.11) og (14.12). Analysefrekvensen for waveleten vi har valgt er $\omega_a = 2\pi 100$ (som svarer til 100 Hz i selve signalet).

Dersom vi bruker likningene (14.9) og (14.8) vil vi for hvert nytt punkt i wavelettransfomasjonen som beregnes (langs tidsaksen), forskyve toppunktet til waveleten fra å ligge helt i ytre venstre kant til helt i ytre høyre kant. Resultatet er vist i figur 14.6. Vi ser at vi får en verdi på ca. 0.6 for de tidspunktene at det opprinnelige signalet hadde en frekvens lik analysefrekvensen.

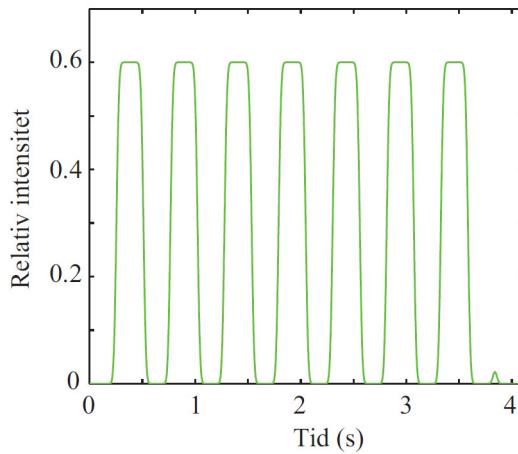


Figur 14.5: Det genererte tidssignalet vi har brukt i eksemplet vårt. Venstre del viser hele signalet, mens en detalj av dette er vist i høyre del. Amplituden på 100 Hz signalet er 0.6 mens amplituden på 200 Hz signalet er 1.0.

Figur 14.7 illustrerer den mer effektive metoden, beskrevet ved likningene (14.11) og (14.12). Den fouriertransformerte av selve signalet multipliseres punkt for punkt med den fouriertransformerte til waveleten (waveleten som svarer til analysefrekvens 100 Hz og gitt K-verdi). Den fouriertransformerte til waveleten er en klokkefunksjon (ca gaussisk form) med posisjon 100 Hz i vårt tilfelle og har ingen “foldet” komponent. Resultatet av den punktvise multiplikasjonen er at bare en av de fire “toppene” i signalets frekvensspekter overlever. Det foretas så en omvendt fouriertransformasjon av dette signalet, og man tar absoluttverdiene. Plotter man resultatet får vi akkurat samme kurve som vist i figur 14.6.

Waveletdiagrammet viser ikke noe tegn til 200 Hz signalet, men det er fordi vi kun har analysert signalet for 100 Hz. For å få et mer fullstendig waveletdiagram, måtte man gjenta prosedyren for et helt sett med frekvenser. Da blir waveletdiagrammet 3-dimensjonalt med tid langs x-aksen, frekvens langs y-aksen og intensiteten som funksjon av tid og frekvens angitt ved hjelp av farger.

En detalj er verd å merke seg allerede her: Kurven i figur 14.6 har avrundete hjørner. Dette skyldes at waveleten har en endelig utstrekning i tid, og derfor vil “oppdage” en 100 Hz sekvens allerede før waveletens toppunkt er innenfor 100 Hz-området. Likeså vil waveleten “oppdage” områder hvor det ikke er 100 Hz selv når toppunktet til waveleten såvidt ligger innenfor 100 Hz-området. Vi kommer tilbake til denne effekten i stor detalj siden.



Figur 14.6: Den wavelettransformerte av tidssignalet for en analysefrekvens på 100 Hz. Parameteren K var 12, hvilket betyr at waveleten var grovt sett om lag $12 \times (1/100)$ s = 0.12 s lang. Bredden på waveleten fører til avrunding av skarpe hjørner i diagrammet.

Det kan nå være passende å gjenta punktene vi gjennom for å få et waveletdiagram (med den mest effektive metoden):

- Beregn den fouriertransformerte av tidssignalet vi skal analysere.
- Beregn direkte den fouriertransformerte til en Morlet wavelet med analysefrekvensen vi er interessert i.
- Multipliser disse med hverandre, punkt for punkt. (Merk at det må være overensstemmelse mellom frekvensene f_k i den fouriertransformerte av signalet og den fouriertransformerte av waveleten.)
- Foreta en invers fouriertransformasjon.
- Absoluttverdien av denne vil da gi informasjon om hvilke tidspunkt det opprinnelige signalet inneholdt frekvenser lik analysefrekvensen.
- Ved å endre Morlet waveleten slik at den svarer til neste analysefrekvens, får vi bygget opp stadig nye horisontale linjer i waveletdiagrammet inntil vi har dekket så mange analysefrekvenser vi ønsker.

Siden Fast Fourier Transform er så effektiv som den er, blir metoden vi nettopp har skissert tilstrekkelig rask til å være nyttig.

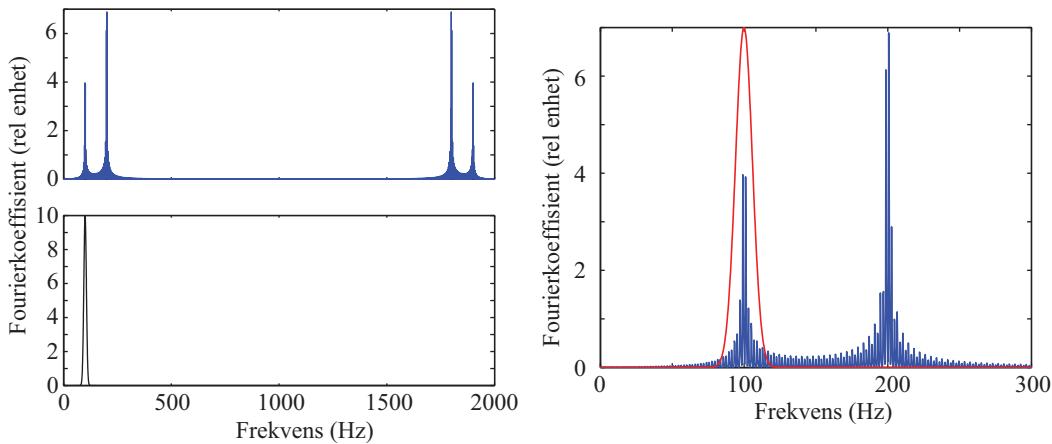
I en programkode litt senere i kapitlet er det vist ett eksempel på hvordan wavelettransfomasjon med den effektive algoritmen kan implementeres.

14.5 Viktige detaljer

14.5.1 Faseinformasjon og skalering av utslaget

I vanlig fouriertransfomasjon foretar vi i prinsippet to transformasjoner samtidig, nemlig en av typen

$$X(\omega) = \int_{-\infty}^{\infty} x(t) \cos(\omega t) dt \quad (14.13)$$



Figur 14.7: Den fouriertransformerte av tidssignalet vårt øverst til venstre, samme med den fouriertransformerte til waveleten nederst til venstre. Til høyre er det vist en detalj for å få fram at den gaussiske kurven på en måte fungerer som et filter og vil bare plukke ut deler av frekvensspekteret til signalet. Se teksten for detaljer.

og en av typen

$$X(\omega) = -i \int_{-\infty}^{\infty} x(t) \sin(\omega t) dt \quad (14.14)$$

Grunnen er at vi har både sinus og cosinusledd er at vi må kunne fange opp f.eks. et sinussignal i x uansett hvilken fase det har.

I vanlig fouriertransformasjon har vi *ett* startpunkt for analysen. Det betyr at det er lett å finne hvilken relativ fase de ulike frekvenskomponentene har.

I kontinuerlig waveletanalyse har vi ulike startpunkt og lengder på analysevinduet underveis i beregningene. Det gjør det langt vanskeligere å holde orden på faser. Dette er nok en av grunnene til at vi nesten utelukkende bare tar utgangspunkt i absoluttverdien av wavelettransformasjonen i en eller annen variant når waveletanalysen skal angis. (Dersom vi imidlertid skulle foreta en invers wavelettransformasjon etterpå, måtte vi selvfølgelig tatt være på faseinformasjonen.)

Det er flere måter vi kan angi styrken i et waveletdiagram. Ofte brukes *kvadratet* av absoluttverdiene, hvilket gir *energien* i signalet.

Erfaringsmessig liker jeg ikke å bruke kvadratet av absoluttverdien, fordi forskjellen mellom de sterke og svake partiene da ofte blir så stor at vi mister informasjon om de svake partiene. Da er det ofte bedre å heller bruke absoluttverdien direkte ("amplitudenivå").

Jeg foretrekker imidlertid ofte å bruke *kvadratroten* av absoluttverdien. Da får jeg fram de svake partiene enda bedre enn om absoluttverdien ble plottet.

Vi står fritt i å velge hvordan resultatet fra wavelettransformasjonen blir plottet, men må ta følgen av vårt valg når vi skal hente kvantitative verdier ut av diagrammene.

14.5.2 Frekvensoppløsning vs tidsoppløsning

Vi skjønte ut fra figur 14.4 at når bølgelengden i signalet x er eksakt lik bølgelengden inne i waveleten, vil wavelettransformasjonen gi maksimal verdi. Endrer vi *litt* på bølgelengden i

waveleten ved å endre på analysefrekvensen, vil transformasjonen gi en lavere verdi, men likevel ikke null verdi. En waveletanalyse vil med andre ord ikke bare gi utslag for en frekvens som svarer til signalets, men også for nærliggende frekvenser.

Tema for dette underkapitlet er å vite hvor langt ut denne "smitteeffekten" går.

La oss da ta utgangspunkt i at en wavelettransformasjon innebærer en "digital filtrering" av et signal, slik vi anskueliggjorde i figur 14.7. Hvor skarp filteringen er bestemmes av bredden på den gaussiske klokkefunksjonen som brukes i filtreringen. *Vi trenger da å finne en sammenheng mellom bredden i frekvensbildet og bredden av waveleten i tidsbildet.*

I figur 14.8 viser til venstre tre ulike valg av waveleter (beregnet ut fra ligning (14.8)), og til høyre er den fouriertransformerte av waveleten (beregnet ut fra ligning (14.12)).

Vi vet fra tidligere at frekvensspekteret fra fouriertransformasjon av et sinussignal konvolutert med en gaussisk omhyllingskurve, selv har en gaussisk omhyllingskurve. Det får vi på ny bekreftet gjennom figur 14.8.

Bredden i tidsutstrekningen til waveleten kan bestemmes ved å ta utgangspunkt i omhyllingskurven (ut fra ligning (14.8)). Dersom vi definerer bredden som tidsforskjellen mellom toppunktet og et punkt hvor amplituden på omhyllingskurven har sunket til $1/e$ av maksimalverdien, er (halv)bredden:

$$\Delta t_{1/e} = 2K/\omega_a$$

Den tilsvarende bredden i den fouriertransformerte av waveleten er meget nær (ut fra ligning (14.12))

$$\Delta f_{1/e} = f_a/K = \omega_a/(2\pi K) \quad (14.15)$$

Det interessante er at

$$\Delta t_{1/e} \Delta f_{1/e} = (2K/\omega_a) \cdot (\omega_a/(2\pi K)) = 1/\pi$$

Dersom vi beregner "standardavviket" for tid og frekvens ut fra mer statistiske mål, slik:

$$\sigma_t^2 = \frac{\int t^2 \Psi^2(t) dt}{\int \Psi^2(t) dt}$$

og

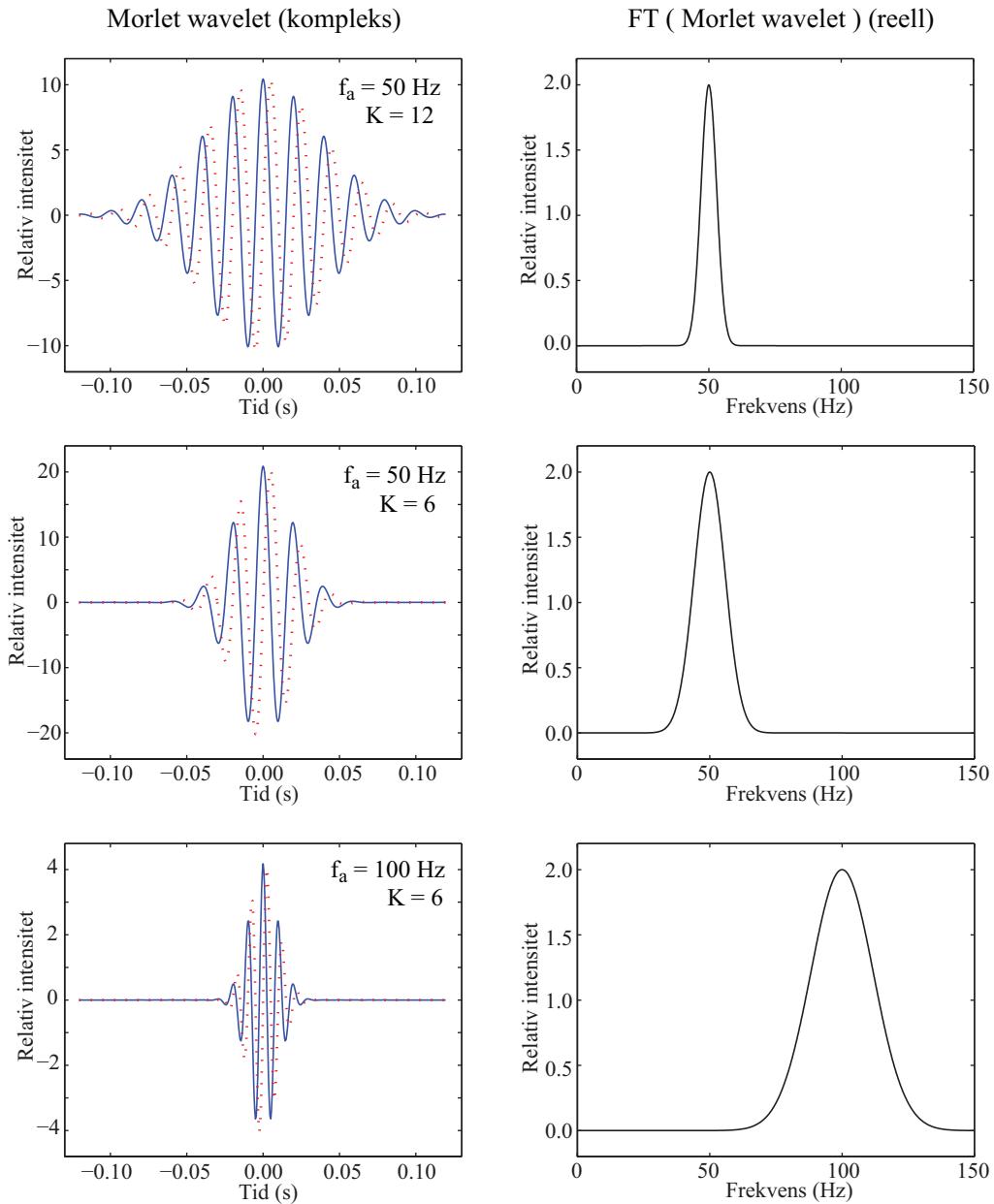
$$\sigma_f^2 = \frac{\int f^2 \hat{\Psi}^2(f) df}{\int \hat{\Psi}^2(f) df}$$

kan det vises at

$$\sigma_t^2 \sigma_f^2 = \frac{1}{2\pi} \quad (14.16)$$

Denne relasjonen er analog til Heisenbergs uskarphetsfunksjon. Eksempler i tråd med denne relasjonen er vist i figur 14.8.

Relasjonen er svært viktig for waveletanalyse. Dersom vi lar en wavelet strekke seg over en lang tid, vil bredden i frekvensdomenet være liten og visa versa. Med andre ord: *Vi kan ikke både få en nøyaktig tidsangivelse av detaljer i et signal samtidig som vi får en nøyaktig frekvensangivelse.*



Figur 14.8: Tre ulike wavelets som indikerer hvordan parametrene analysefrekvensen ω_a og “bølgetallet” K virker inn på waveleten. En wavelet har en avgrenset utstrekning i tid (venstre del). Vi kan angi en bredde på omhyllingskurven f.eks. ut fra at verdien har sunket til $1/e$ av toppverdien.. Fourieromvendes denne waveleten, får vi frekvensresponsene vist til høyre. Legg merke til både posisjon i frekvensspekteret og bredden på de gaussformede kurvene. Bredden på frekvensresponsen kan angis på lignende måte som i tidsbildet. Det interessante er at bredden i tidsdomenet multiplisert med bredden i frekvensdomenet er en konstant, hvilket innebærer at dersom den ene økes, vil den andre avta og visa versa.

En interessant følge av ligning (14.15) er at

$$\Delta f_{1/e}/f_a = 1/K$$

Med andre ord, i en waveletanalyse holder vi oftest K konstant i hele analysen. Da er den relative usikkerheten i frekvensangivelsene konstant i hele diagrammet.

Det er da naturlig å velge en logaritmisk frekvensakse, i betydning at frekvensene vi velger å ta med i analysen forholder seg til hverandre som

$$(f_a)_{k+1} = (f_a)_k \cdot f_{faktor}$$

Vi har valgt logaritmisk akse for de valgte analysefrekvensene i alle eksemplene i dette kapitlet, men det er selvfølgelig mulig å velge analysefrekvensene også ut fra en lineær skala, i alle fall dersom forskjellen mellom minste og største analysefrekvens er liten (f.eks. en faktor to eller mindre).

[♠ ⇒ Sammenligning wavelets vs stykkevis FT

Skulle vi bruke stykkevis FT, ville et fast tidsintervall medføre at vi har svært få (eller ingen hele) periodetider innenfor intervallet for lave frekvenser, men ganske mange periodetider for høye frekvenser. Det betyr at vi ville få en elendig frekvensoppløsning for de laveste frekvensene (målt som relativ frekvens), men en langt bedre frekvensoppløsning for de høyere frekvensene. Det betyr at vi ville ende opp med en analyse som ikke ville være optimal.

Prosedyren som brukes i waveletanalyse gir en optimal tidsoppløsning for *alle* frekvenser. Men vi *kan* likevel velge å vektlegge tidsoppløsning *noe* på bekostning av frekvensoppløsning og omvendt alt etter hva vi ønsker å studere. Det gjør at metoden blir et meget slagkraftig hjelpemiddel i mange sammenhenger.

Kunne vi valgt en stykkevis FT der vi faktisk brukte skaleringsprinsippet med å stykke opp tidsstrenget i mindre biter når vi analyserte høye frekvenser enn ved lave? Det ville kreve en enorm mengde fouriertransformasjoner, men ville da gi omrent samme resultat som en waveletanalyse. Resultatet ville likevel ikke bli like godt. Vi ville nemlig med skalerete FT-intervaller i prinsippet hatt en ren waveletanalyse, men med en wavlet som har en annen form enn Morlet. Det ville være en firkantpuls som da var omhyllingskurven. Frekvensresponsen ville da ha en form

$$\left(\frac{\sin(x)}{x} \right)^2$$

i stedet for en gaussisk omhyllingskurve. Resultatet ville bli en hale utenfor den sentrale toppen som gjør at vi får en dårligere frekvensbestemmelse enn ved Morlet wavelets.

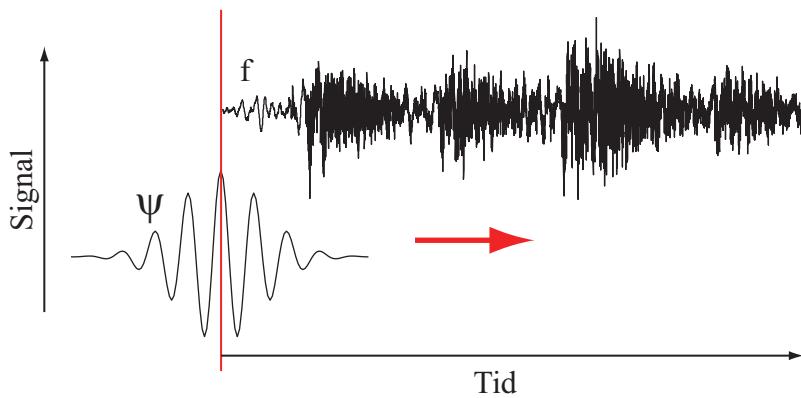
⇐ ♠]

14.5.3 Randproblem

Når vi foretar en wavelettransformasjon, multipliserer vi i prinsippet et signal med en wavelet punkt for punkt og summerer alle produktene. Vi flytter så waveleten og gjør det samme på ny. Dette gjentas om igjen og om igjen fra den situasjonen at waveletens midtpunkt ligger helt i den ene enden av signalet til waveletens midtpunkt ligger i andre enden av signalet.

Vi endrer så analysefrekvensen for waveleten og gjør det samme på ny.

Her oppstår det imidlertid et problem. Så lenge wavleten ikke er fullstendig innenfor dataområdet, vil vi måtte forvente et annet resultat enn om hele waveleten ble brukt i

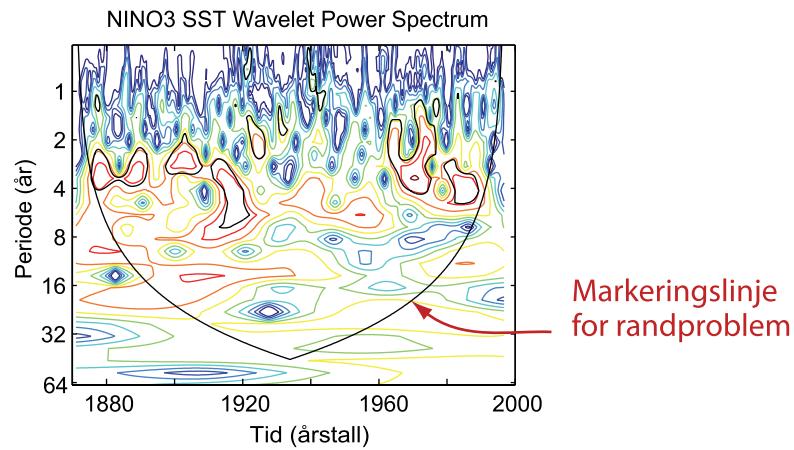


Figur 14.9: Det er ikke mulig å få et korrekt waveletresultat for tider og frekvenser der ikke hele waveleten kommer innenfor dataområdet under beregningene.

beregningene. Dette er anskueliggjort i figur 14.9. For den posisjonen waveleten har i forhold til dataene i denne figuren, vil bare om lag halvparten av waveleten benyttes i praksis. Det betyr at summen av produktene må forventes å bli langt lavere (i størrelsesorden halvparten) av hva summen ville blitt dersom vi hadde fullt overlapp.

Av den grunn kan vi ikke stole noe særlig på datapunktene i en endelig wavelettransformasjon dersom waveleten som er brukt ikke ligger fullstendig innenfor datastrengen vi analyserer. Det er derfor vanlig å markere ytterområdet mhp tid hvor vi har et randproblem.

I figur 14.10 er det vist et eksempel på et waveletdiagram etter analyse av temperaturoscillasjoner i det sydlige Stillehav. Det er brukt kvoter sammen med farger for å markere "energien" i ulike former for svingninger (periodisitet) etter som de har utviklet seg de siste vel hundre år.



Figur 14.10: Eksempel på et waveletdiagram for temperaturoscillasjoner i det sydlige stillehav. Figuren er produsert med data og programvare som var tilgjengelig fra <http://paos.colorado.edu/research/wavelets/> i 20. april 2016.

I dette diagrammet er det tegnet inn en krum V-formet linje som starter langt nede og midt i waveletdiagrammet med symmetriske buede linjer som går opp og ut mot kantene.

Disse linjene markerer området hvor det aller meste av waveletene er fullstendig innenfor datastrengen: Alt over denne buede V-en er gyldige data. Alle resultater under streken må vi ta med en klype salt.

I programeksemplene gitt i resten av dette kapitlet har vi valgt å legge inn en markering som svarer til at bare den ytre delen av waveleten som har verdi mindre enn $1/e$ av maksimum på omhyllingskurven ligger utenfor diagrammet. Vi har bare med et så lite frekvensområde at vi i egne eksempler ikke får fram hele den buete V-en, men bare et smalt horisontalt bånd av den totale V-en. Alle deler av waveletdiagrammet som ligger mellom disse markeringene har kun ubetydelige feil på grunn av randproblematikken. Vi gir detaljer nedenfor om hvordan markeringene settes opp.

14.6 Optimalisering

Waveletanalyse er mer krevende enn vanlig fouriertransformasjon. Vi må velge hva slags wavelet vi vil bruke. Selv om vi holder oss til Morlet wavelets, må vi bestemme oss for hvilket “bølgetall” vi vil bruke.

Vi har tidligere sett at ved å øke “bølgetallet” K , vil waveleten ha en betydelig verdi over et større tidsrom enn ved lavt “bølgetall” (ved samme analysefrekvens). Videre har vi sett at når “bredden” på waveleten i tidsdomenet er stor (det vil si stor K -verdi), vil “bredden” på den fouriertransformerte av waveleten bli liten. Produktet av bredden på waveleten i tidsdomenet og bredden av waveleten i frekvensdomenet er jo konstant.

Følgen er at det finnes ingen Ole Brumm-løsninger: “Ja takk, begge deler” innen waveletanalyse. Ønsker vi å få en nøyaktig angivelse av tidsforløp, vil små K -verdier være å foretrekke. Ønsker vi å få så nøyaktige frekvensangivelser som mulig, bør K -verdien være stor. Vi ønsker i prinsippet så god tidsoppløsning og frekvensoppløsning som mulig, men må alltid velge et kompromiss.

Det optimale resultatet oppnås ofte dersom vi tar utgangspunkt i signalet selv. Signalet har ofte innebygget en uskarphet i tid og/eller en uskarphet i frekvens. Vi kan aldri få en bedre oppløsning i tid ved waveletanalyse enn den oppløsningen signalet selv har. Tilsvarende for analyse av frekvens.

I figur 14.11 er det vist waveletdiagrammer av signalet vi drøftet ovenfor som vekslet mellom 100 og 200 Hz. Fem ulike bølgetall K er brukt. Vi ser at for lave K -verdier er tidsoppløsningen meget presis, men frekvensbestemmelsen er elendig. For høye K -verdier er det motsatt: Frekvensoppløsningen er god, men tidsoppløsningen er elendig.

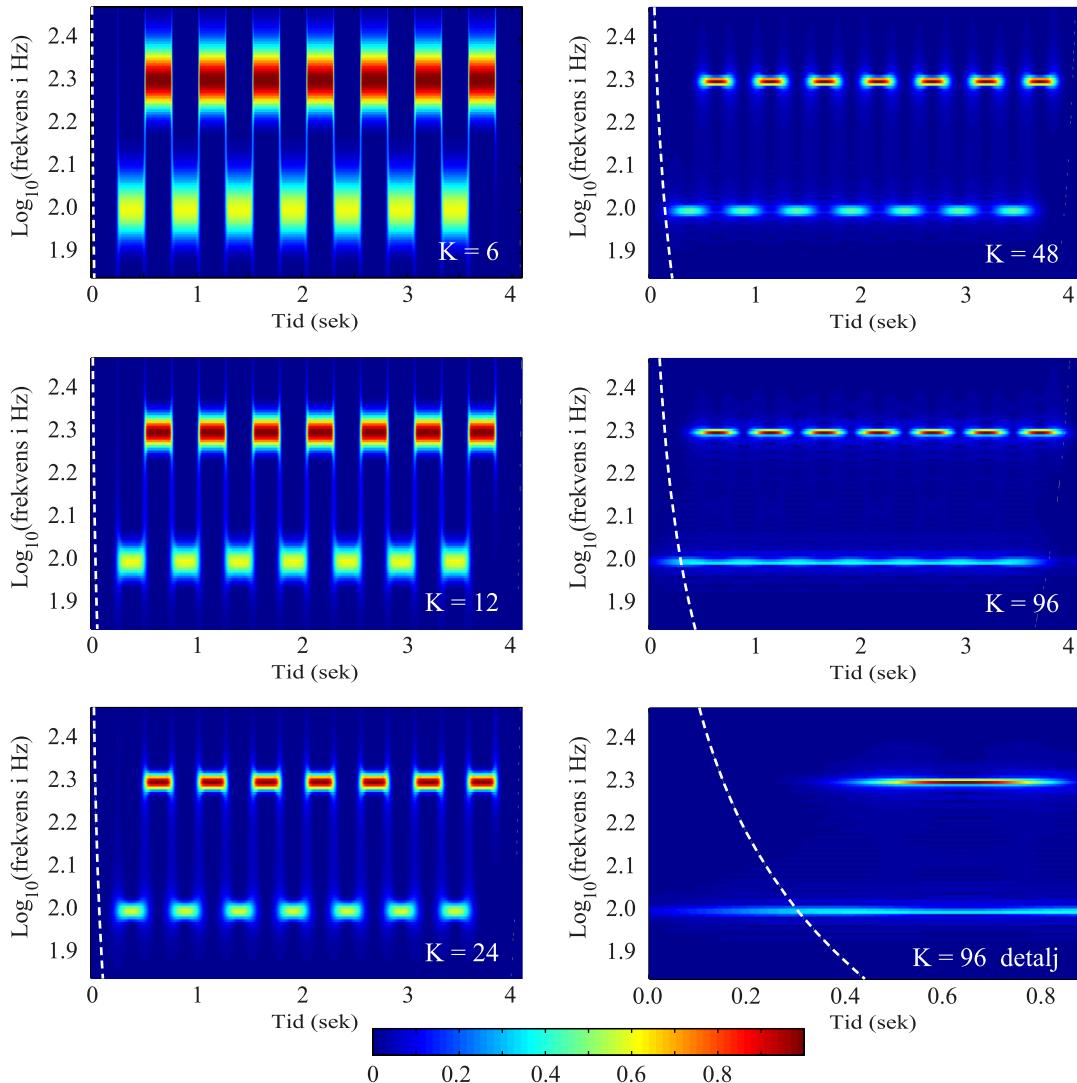
I dette tilfellet er det egentlig ikke mye mer å hente i frekvensoppløsning når vi går fra $K = 48$ til $K = 96$. Dette skyldes at signalet selv har en “uskarphet i frekvens” siden varigheten av hver periode med “konstant frekvens” er begrenset. Når waveletens varighet ved den aktuelle analysefrekvensen omrent svarer til varigheten av hver periode i selve signalet med “konstant frekvens”, får vi ofte beste resultatet. I vårt tilfelle oppnår vi dette med en K -verdi litt større enn 48 (men litt forskjellig for de to analysefrekvensene).

I figur 14.11 har vi gjort markeringen av venstre side av randproblemsområdet ekstra tydelig. Vi ser da klart at randproblemet øker med økende K -verdi. Det kan være interessant å merke seg at randproblemmarkeringen endrer seg med analysefrekvensen. Videre er det nyttig å vite at avstanden fra sidekanten til randproblemmarkeringen også indikerer utsmøring i

klare tidsangivelser i waveletdiagrammet. All tidsinformasjon i analysen blir smurt ut om lag med en tidsforskjell som nettopp svarer til avstanden fra randen til randproblemmerkeringen.

Hva er da “beste valg” av alle analysene gitt i figur 14.11? Vel, det kommer an på hva vi ønsker å få fram av opplysninger. Diagrammet for $K = 6$ demonstrerer at endringen fra 100 til 200 Hz (og omvendt) foregår meget skarpt i tid. Diagrammet for $K = 96$ viser at frekvensen er så ensartet som den kan være innenfor hver av tidsintervallene. Skulle vi velge en slags generell optimalisering, kunne kanskje $K = 48$ eller deromkring være et godt valg.

En vanlig fouriertransformasjon av signalet ville gitt to topper, en for 100 Hz og en for 200 Hz. Hadde vi tatt absoluttverdien av frekvensspekteret ville vi overhodet ikke sett noe spor som kunne vise at signalet vekslet mellom 100 og 200 Hz i tid.



Figur 14.11: Waveletdiagrammer for tidssignalet i figur 14.5 for seks ulike “bølgetall” K . Se teksten for detaljer.

14.6.1 Optimalisering i frekvensoppløsning (programmeringsteknisk)

En annen form for optimalisering ligger i valg av frekvensområde for analysen. I en digital fast fourier analyse får vi automatisk “alle” frekvenser mellom null og samplingsfrekvensen (men bare halvparten er nyttig pga folding). For en kontinuerlig waveletanalyse velger vi oftest å innskrenke frekvensområdet til det området der frekvensinnholdet er av interesse.

I figur 14.11 valgte vi bare å ta med frekvenser mellom 70 og 300 Hz i analysen. Grunnen er at vi visste at signalet bare inneholdt frekvenser nær opp til 100 og 200 Hz. Det kan ofte være en fordel å starte med en vanlig fouriertransformasjon for å sikre oss at vi velger et frekvensområde som egner seg.

Det er imidlertid viktig også å tenke på hvor mange mellomliggende frekvenser vi skal ta med i analysen. I denne sammenheng må vi gå tilbake til “bredden” av wavleten i frekvensdomenet. Denne bredden var som vi tidligere har sett:

$$\Delta f = f_a/K$$

Denne “bredden” var bestemt ved at den gaussformede frekvenskurven var kommet ned til $1/e$ av maksimum verdi. Vi ønsker ikke å ta så store steg i frekvens fra en analysefrekvens til den neste, men kanskje bare en brøkdel av dette.

Praktisk testing viser at et optimalt valg av forskjellen mellom en analysefrekvens og den neste er da om lag

$$f_{a,neste} - f_{a,naa} = f_{a,naa}/8K \quad (14.17)$$

Dersom vi skal spenne over et frekvensintervall $[f_{start}, f_{slutt}]$ kan det da enkelt vises at vi bør bruke M analysefrekvenser i en logaritmisk orden, hvor

$$f_{slutt} = (1 + \frac{1}{8K})^{M-1} \cdot f_{start}$$

Antall analysefrekvenser er da:

$$M = 1 + \log(f_{slutt}/f_{start}) / \log(1 + \frac{1}{8K}) \quad (14.18)$$

14.6.2 Optimalisering i tidsoppløsning (programmeringsteknisk)

Et kontinuerlig waveletdiagram kan iblant bestå av svært mange punkter. Dersom vi f.eks. tar utgangspunkt i lyd som er digitalisert med en samplingsfrekvens på 44.1 kHz, og vi studerer lyd med frekvens i området 100 - 10 000 Hz, kan vi i praksis plukke ut bare hvert fjerde punkt i tidsdimensjonen uten problemer. Når vi at på til vet at wavleten har en bredde på om lag K ganger periodetiden for analysefrekvensen, innser vi at vi kan fjerne enda flere punkter i tidsdimensjonen uten at det vil oppdages i et waveletdiagram.

Det kan iblant være av interesse å optimalisere et waveletdiagram mhp tidsangivelsen. Ikke minst er dette viktig for å få plottefiler som er så små at de lett lar seg innlemme i rapporter og liknende.

I praksis viser det seg at vi kan nøye oss med å gjengi hvert P -te punkt i tidsdimensjonen i et waveletdiagram (uten at nevneverdig informasjon går tapt) når P er gitt ved:

$$P = \text{Heltallsverdien} - \text{av} \left(\frac{K}{24} \frac{f_s}{f_{a,\max}} \right) \quad (14.19)$$

14.7 Eksempler på wavelettransformasjon

14.7.1 Gjøkens “ko ko”

Figur 14.12 viser et eksempel på en optimalisert waveletanalyse. Signalet er en lydfil i CD-kvalitet som gir lyden av en gjøk som synger sitt “ko ko”. Signalet er gitt i tre varianter, nemlig som et rent tidssignal, som frekvensspekter etter en vanlig FT, og endelig analysert ved å bruke wavelettransformasjon.

I den totale arbeidsgangen er første trinn å velge ut et passe utsnitt fra lydfilen. Det gjøres ved at vi velger startpunkt og totalt antall punkter som skal hentes ut fra den tilgjengelige datafilen. Dernest foretas en fourieranalyse. Fra fourierspekteret ser vi at lyden stort sett bare inneholder frekvenser mellom 450 og 750 Hz. Av den grunn er waveletanalysen begrenset til dette frekvensintervallet.

Til slutt må vi forsøke med ulike K -verdier og velge et “beste” kompromiss mellom god tidsangivelse og frekvensangivelse samtidig. Vi må da bestemme oss for hvorvidt vi vil prioritere tidsoppløsning (ved å ha en liten K), men da på bekostning av en nokså bred frekvensrespons, eller godta en litt dårligere tidsoppløsning (ved å velge en større K) for å få en noe bedre frekvensoppløsning. Hva som er optimalt kommer an på selve signalet vi analyserer og vil avhenge av hva den enkelte vektlegger mest. I vårt eksempel ble $K = 40.0$ benyttet.

Legg merke til de nydelige detaljene som kommer fram i waveletanalysen. Har du f.eks. vært klar over at lyden i det første “ko”-et faktisk endrer seg betydelig den korte stunden lyden varer? Detaljer i waveletanalyse av fuglesang gjør det mulig for ornitologer å gjenkjenne fugler individuelt. Detaljene er finere enn hva en menneskehørsel klarer å oppfatte.

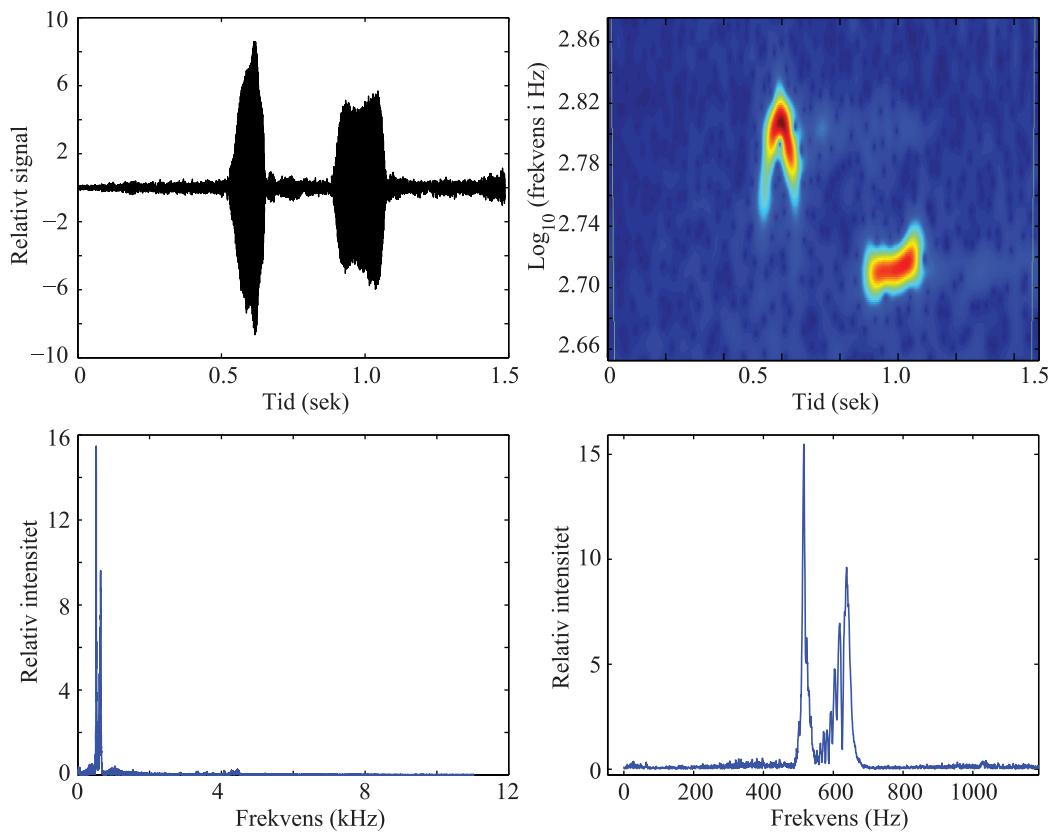
Det bør være innlysende at wavelettransformasjon gir langt mer interessante data enn en vanlig fouriertransformasjon for en slik type lyd.

14.7.2 Bokfinkens sang

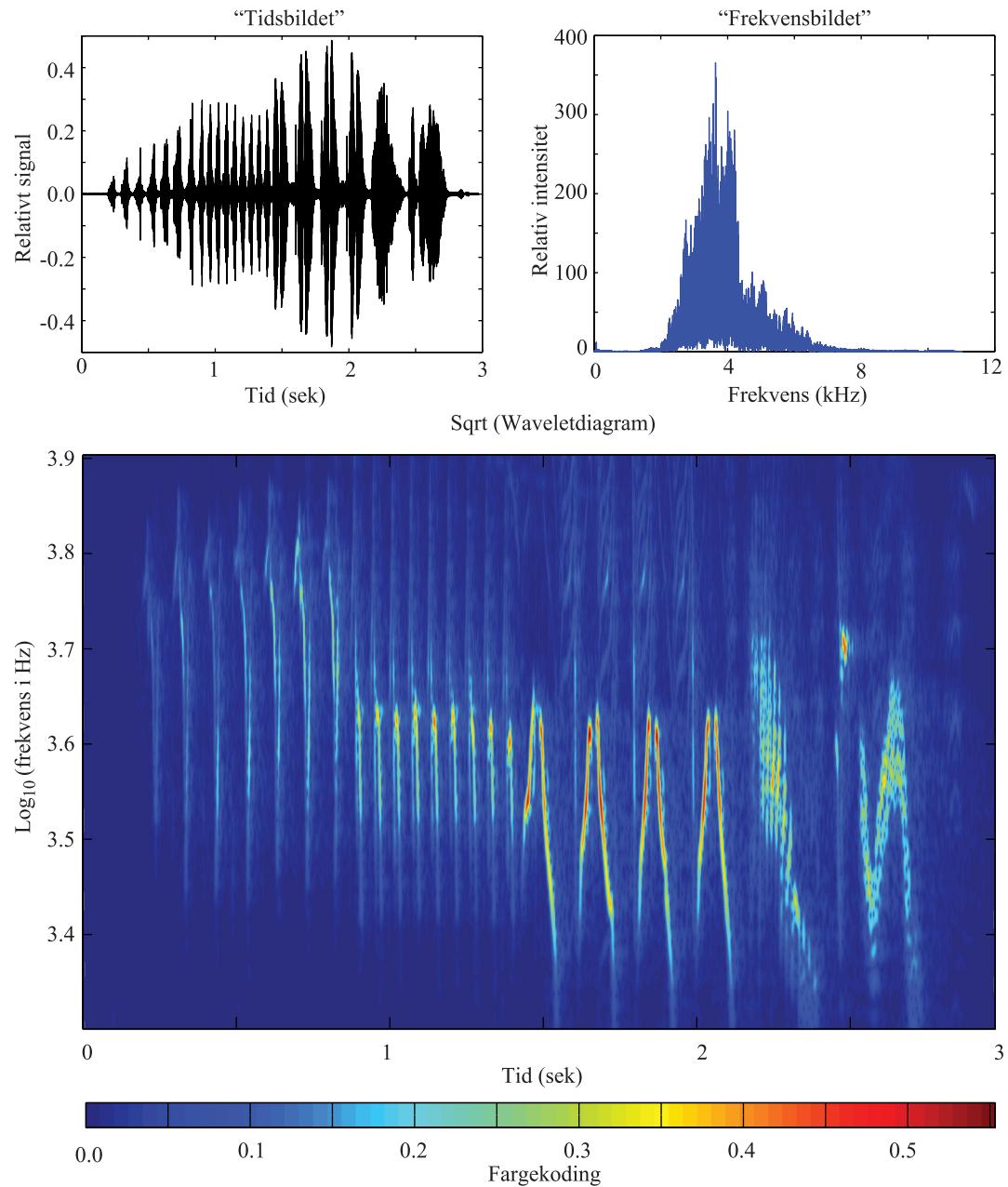
Vi tar med to ytterligere eksempler på waveletanalyse. Det første (figur 14.13) er liknende den vi hadde for gjøkens ko ko. Vi har valgt kvitringen til en bokfink, som dominerer fuglesangen i april. Bokfinkens sang blir karakterisert på flere ulike måter. Selv liker jeg best karakteristikken “tit tit tit tit tit.... el-ske-de-viv”. Det morsomme med waveletanalysen er at lydbildet er langt mer komplisert enn det vi mennesker oppfatter. Det er en meget rask variasjon i frekvens innen hver “tit” som vi ikke oppfatter. K -verdi brukt i analysen var 48.0.

14.7.3 Trompetlyd, harmoniske i logaritmisk skala

Det siste eksemplet er en waveletanalyse av en trompetlyd (figur 14.14). Vi har valgt et utsnitt i tid der trompeten holder samme tone, og intensiteten på lyden opplever vi som temmelig konstant. Tidsbildet av lyden viser mer variasjon i intensitet enn det vi oppfatter. Frekvensbildet (frekvensspekteret) er imidlertid helt slik vi hadde forventet det. Det består av en rekke skarpe linjer som viser grunntonen og de harmoniske.



Figur 14.12: Gjøkens "ko ko" analysert i tidsbildet, i frekvensbildet (med et utsnitt) og i kombinasjonen: Frekvens og tid i form av et waveletdiagram.



Figur 14.13: *Bokfinkens "tit tit tit tit tit.... el-ske-de-viv"* analysert i tidsbildet, i frekvensbildet og i en wavelettransformasjon.

Frekvensaksen er lineær, og derved er avstanden mellom to nærliggende harmoniske fast, nærmere bestemt lik frekvensen til grunntonen.

En waveletanalyse av et slikt signal klarer ikke å matche skarpheten i frekvensspekteret, så dersom vi først og fremst er interessert i frekvensen til grunntonen og de harmoniske *for en vedvarende tone*, er fourieranalyse metoden som bør velges.

Dersom vi er interessert i variasjoner i lyden over tid, egner imidlertid ikke fourieranalysen seg. Da kommer waveletanalysen inn. Vi har tatt med to ulike varianter av analyse, basert på bølgetallene $K = 24$ og $K = 96$. I det første tilfellet er frekvensoppløsningen nokså dårlig, men tidsoppløsningen passer i forhold til signalet. I det siste tilfellet er frekvensoppløsningen god, men tidsoppløsningen dårlig.

Får vi noe ekstra ut av waveletanalysen framfor fourieranalysen? Ja, faktisk. Vi ser at styrken på grunntonen og de harmoniske varierer litt i tid. Vi ser også at det er en viss veksling mellom intensiteten til grunntonen og den første harmoniske: Når den ene er kraftig er den andre svak og visa versa. Dette gir liv til lydbildet, og viser et eksempel på at det er vanskelig å erstatte virkelig lyd med syntetisk lyd.

Legg forøvrig merke til at avstanden mellom de harmoniske ikke er konstant i et normalt waveletdiagram, siden vi der normalt har en logaritmisk frekvensakse.

Frekvensene f_n til de harmoniske var som vi husker $f_n = nf_1$ hvor f_1 er frekvensen til grunntonen. Vi har da

$$\log(f_n) = \log(nf_1) = \log(f_0) + \log(n)$$

for $n = 1, 2, 3, \dots$. Da følger:

$$\log(f_1) = \log(f_1)$$

$$\log(f_2) = \log(f_1) + \log(2) = \log(f_1) + 0.301$$

$$\log(f_3) = \log(f_1) + \log(3) = \log(f_1) + 0.477 = \log(f_2) + 0.176$$

$$\log(f_4) = \log(f_1) + \log(4) = \log(f_1) + 0.602 = \log(f_3) + 0.125$$

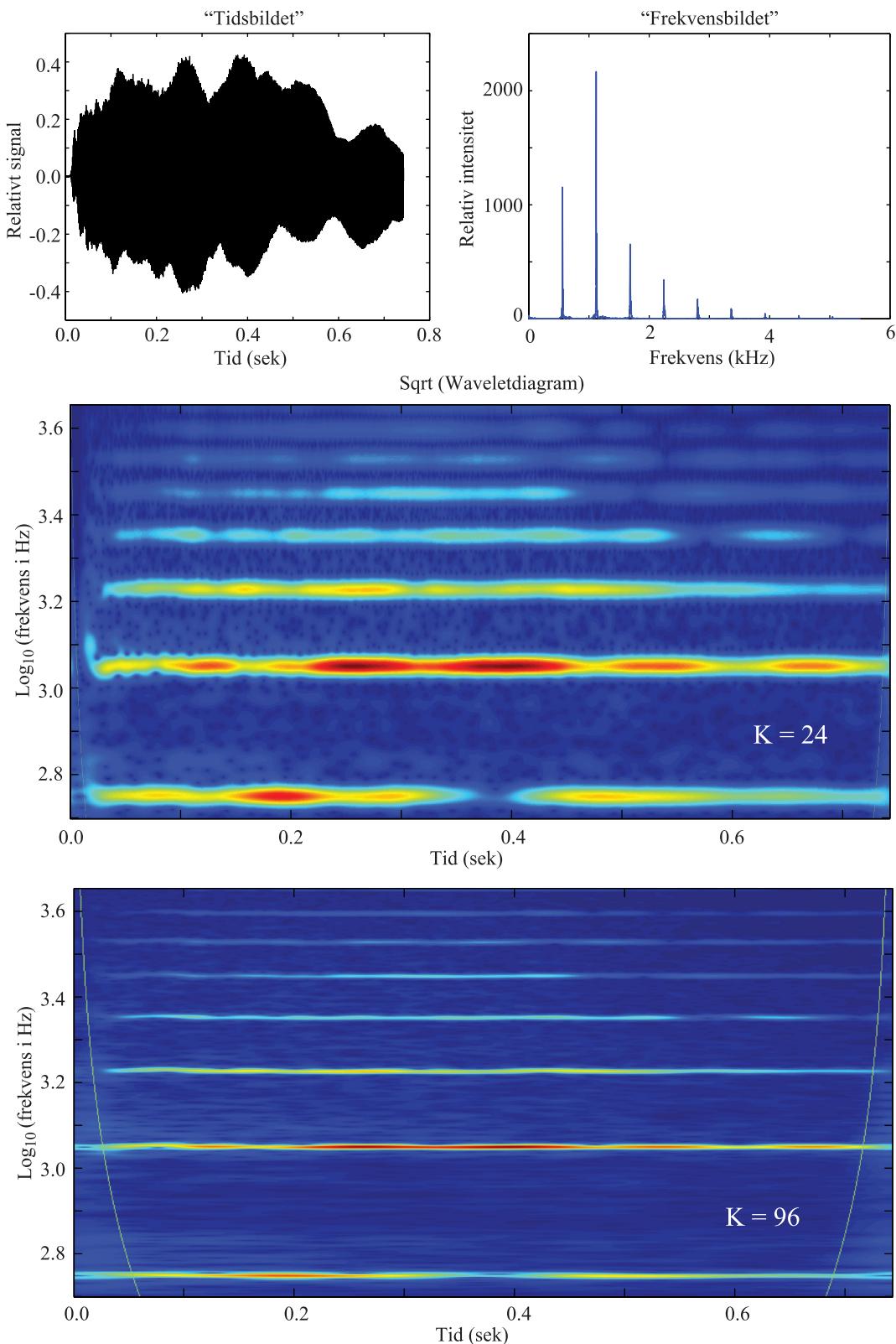
$$\log(f_5) = \log(f_1) + \log(5) = \log(f_1) + 0.699 = \log(f_4) + 0.097$$

Avstanden mellom de harmoniske vil med en logaritmisk frekvensakse være:

$$0.301, 0.176, 0.125, 0.097, \dots$$

uansett hvilken frekvens grunntonen har (se figur 14.14), noe som er ganske annerledes enn i et vanlig fourierspekter.

Det er iblant svært nyttig å bruke disse avstandene både for å gjenkjenne harmoniske og for å være sikker på at man har valgt et frekvensområde for analysen som inkluderer grunntonen (der det er viktig).



Figur 14.14: En ren trompetlyd analysert i tidsbildet, i frekvensbildet og i en wavelettransformasjon. To ulike K -verdier er brukt i waveletanalysen.

14.8 Matlabkode for wavelettransformasjon

Vi har tatt med et Matlab program som kan brukes for å analysere biter av lydfiler. Programmet kan kopieres direkte fra pdf-filen slik at du slipper å skrive den inn på nytt.

Koden er fordelt på fire ulike funksjoner: Et “hovedprogram”, en funksjon som leser inn lydfiler i wav-format og plotter tidsbildet av signalet, en funksjon som foretar en fouriertransformasjon og plotter resultatet, og en funksjon som foretar en waveletanalyse av signalet (gitt et sett parametre som vi selv må bestemme). “Hovedprogrammet” anvender de tre andre funksjonene etter tur.

[♣ ⇒ Noen kommentarer:

Programmet er selvsnekret og bærer preg av at jeg lærte programmering i en helt annen tid enn nå. Skriv gjerne din egen programversjon mer i tråd med moderne trender og krav. Imidlertid, dersom du har knapt med tid, kan det være lurt å anvende vårt Matlabprogram direkte for å få gjort det du må gjøre, og heller på lengre sikt forbedre programmet på egen hånd (enten i Matlab eller Python). ← ♣]

HOVEDPROGRAMMET

```
function WLavWAV

% "Hovedprogram" som leser en wav-fil, plotter tidbildet
% og frekvensbildet, og dernest den wavelettransformerte.

% Leser en wavfil
c = 'gjok.wav';
nstart = 31000;
N = 1024*32;
[fs,h] = leswavfil(c,nstart,N);

% Fouriertransformerer
[FTsignal] = fftogplot(h,N,fs);

% Wavelet-analyse
fmin = 400.0;    % Bestemmes ut fra FT og skjønn
fmax = 800.0;    % ds
K = 32;           % Må optimaliseres for hvert signal
[msg] = wltransf(FTsignal,fmin,fmax,K,N,fs);
```

LESER WAV-FIL OG PLOTTER RESULTATET

```
function [fs,h] = leswavfil(c,nstart,N)

% Denne funksjonen leser en wav-fil ned filnavn c.
% Vi starter lesingen nstart punkter etter filstarten,
% og det leses N datapunkter fra filen.
% Lyden spilles, og signalet plottes.
% Funksjonen returnerer samplingsfrekvensen og
% den ene kanalen av stereosignalet i wav-filen.
% Denne versjonen er fra 20. april 2016

nslutt = nstart+N-1;
[y, fs] = audioread(c, [nstart nslutt]); % Les array y(N,2) fra fil
% 'fs' er vanligvis 44100 (samplingsfrekvens ved CD kvalitet)
h = zeros(N,1);    % Plukker ut bare én kanal fra stereosignalet lest
h = y(:,1);
sound(h,fs);        % Spiller av utsnittet som er brukt
T = N/fs;           % Total tid lydutsnittet tar (i sek)
t = linspace(0,T*(N-1)/N,N);
plot(t,h,'-k');
title('Wav-filens signal');
xlabel('Tid (sek)');
ylabel('Signal (rel enhet');
```

BEREGNER FFT OG PLOTTER RESULTATET

```
function [FTsignal] = fftogplot(h,N,fs)

% Funksjon som foretar en fft av et signal h med N punkter.
% Samplingsfrekvensen er fs. Den fouriertransformerte
% plottes (absoluttverdien), mens det komplekse
% fouriertransformerte signalet returneres til den kallende funksjon.
% Versjon 20.4.2016

% Beregner først FFT av tidsstrenge h
FTsignal = fft(h);

% Plotter frekvensspekteret (absoluttverdier only)
f = linspace(0,fs*(N-1)/N, N);
nmax = floor(N/2);    % Plotter bare opp til halve samplingsfrekv.
figure;
plot(f(1:nmax),abs(FTsignal(1:nmax)));
xlabel('Frekvens (Hz)');
ylabel('Relativ intensitet');
title('Frekvensspektrum av signal');
```

BEREGNER DEN WAVELETTRANSFORMERTE OG PLOTTER RESULTATET

```

function [msg] = wltransf(FTsignal,fmin,fmax,K,N,fs)

% Funksjonen foretar ALT som er involvert ved beregning
% inklusiv optimalisering av antall frekvenser som skal beregnes og antall
% punkter i tidsbeskrivelsen som skal benyttes i endelig plot.
% Som input har vi FT av signalet vi skal analysere, og
% min og max frekvens som skal inngå i wl-analysen.
% Vi kan velge mellom to ulike skaleringer av "intensitet".
% Til slutt plottes waveletdiagrammet.
% Versjon 20.4.2016

% Beregner # analysefrekvenser, skriver til skjerm, klargjør frekvensene
M = floor(log(fmax/fmin) / log(1+(1/(8*K)))) + 1;
AntallFrekvenserIAlyse = M
ftrinn = (fmax/fmin)^(1/(M-1));
f_analyse = fmin;
T = N/fs; % Total tid lydutsnittet tar (i sek)
t = linspace(0,T*(N-1)/N,N);
f = linspace(0,fs*(N-1)/N, N);

% Allokkerer plass til waveletdiagrammet og array for lagring av frekvenser
WLdiagram = zeros(M,N);
fbrukt = zeros(1,M);

% Løkke over alle frekvenser som inngår i analysen
for jj = 1:M
    faktor = (K/f_analyse)*(K/f_analyse);
    FTwl = exp(-faktor*(f-f_analyse)).*(f-f_analyse));
    FTwl = FTwl - exp(-K*K)*exp(-faktor*(f.*f)); % Lite korreksjonsledd
    FTwl = 2.0*FTwl; % Faktor (ulike valg!)
    % Beregner så en hel linje i waveletdiagrammet i én jafs!
    %WLdiagram(jj,:) = abs(ifft(FTwl.*transpose(FTsignal))); % Ett alternativ
    WLdiagram(jj,:) = sqrt(abs(ifft(FTwl.*transpose(FTsignal)))); % Ett annet
    % Bruker den siste varianten for å få svake partier bedre synlig
    fbrukt(jj) = f_analyse; % Lagrer frekvensene som faktisk er brukt
    f_analyse = f_analyse*ftrinn; % Beregner neste frekvens
end;

% Reduserer filstørrelse ved å fjerne mye av overflødig informasjon i tid.
% Dette gjøres kun for at filstørrelsen på plottene skal bli håndterbar.
P = floor((K*fs)/(24 * fmax)); % Tallet 24 kan endres ved behov
TarBareMedHvertXITid = P % Skriver til skjerm (monitorering)
NP = floor(N/P);
AntallPktITid = NP % Skriver til skjerm (monitorering)
for jj = 1:M
    for ii = 1:NP
        WLdiagram2(jj,ii) = WLdiagram(jj,ii*P);
        tP(ii) = t(ii*P);
    end;
end;

```

(Fortsetter på neste side.)

```
% Foreta en markering i plottet for å vise områder med randproblemer
maxverdi = max(WLdiagram2);
mxv = max(maxverdi);
for jj = 1:M
    m = floor(K*fs/(P*pi*fbrukt(jj)));
    WLdiagram2(jj,m) = mxv/2;
    WLdiagram2(jj,NP-m) = mxv/2;
end;

% Plotter waveletdiagrammet
figure;
imagesc(tP,log10(fbrukt),WLdiagram2);
set(gca,'YDir','normal');
xlabel('Tid (sek)');
ylabel('Log10(frekvens i Hz)');
%title('Wavelet Power Spektrum'); % Velg denne når det er aktuelt,
title('Sqrt(Wavelet Power Spektrum)'); % men denne når sqrt blir brukt
colorbar('location','southoutside');

msg = 'Done!';
```

14.9 Wavelet-ressurser på nett

1. A-H Najmi og J Sadowsky: "The continuous wavelet transform and variable resolution time-frequency analyses." Johns Hopkins APL technical digest, vol 18 (1997) 134-140. Tilgjengelig på <http://www.jhuapl.edu/techdigest/TD/td1801/najmi.pdf> den 20. april 2016.
2. <http://www.polyvalens.com>, "A really friendly guide to wavelets", med mere (C. Valens). Tilgjengelig 20. april 2016.
3. <http://tftb.nongnu.org/>, "Time-frequency toolbox". Tilgjengelig 20. april 2016.
4. <http://dsp.rice.edu/software/rice-wavelet-toolbox>, "Rice Wavelet Toolbox." Tilgjengelig 20. april 2016.
5. <http://www.cosy.sbg.ac.at/~uhl/wav.html>, Mengde wavelet-lenker. Tilgjengelig 20. april 2016.
6. Et 72 siders hefte av Liu Chaun-Lin: "A tutorial of the wavelet transform" (datert 23. februar 2010) er tilgjengelig på <http://disp.ee.ntu.edu.tw/tutorial/WaveletTutorial.pdf> tilgjengelig 20. april 2016. Heftet tar også for seg wavelets brukt i bildebehandling.

14.10 Læringsmål

Etter å ha jobbet deg gjennom dette kapitlet bør du kunne:

- Gjøre rede for likheter og forskjeller mellom fouriertransformasjon og wavelettransfomasjon.
- Gjøre rede for hvilke signaler fouriertransformasjon er å foretrekke og hvilke signaler der wavelettransfomasjon foretrekkes. Begrunn hvorfor.
- Forklare hva vi kan lese ut av et gitt waveletdiagram.
- Forklare hvordan vi kan justere en wavelettransfomasjon for å fremheve detaljer i tid, eller detaljer i frekvens.
- Forklare kvalitativt analogier mellom wavelettransfomasjon og Heisenbergs uskarphetsrelasjon.
- Bruke et waveletanalyseprogram og optimalisere analysen.

14.11 Oppgaver

Forståelses- / diskusjonsspørsmål

1. Hva er viktigste forskjell mellom fouriertransformasjon og waveletanalyse?
2. I hvilke situasjoner gir fouriertransformasjon et temmelig ubruklig resultat?
3. Hvilke ulemper har wavelettransfomasjon sammenlignet med fouriertransfomasjon?
4. Når ble fouriertransfomasjon tatt i bruk i stor stil (FFT), og når ble wavelettransfomasjon tatt i bruk i betydelig grad?
5. Wavelettransfomasjon har et "randproblem". Hva menes med det? Hvor stor er randverdisonen?
6. Kan du skissere hvordan wavelettransfomasjon kan tenkes brukt for å generere noter direkte fra et lydopptak? Hvilke problemer ser du for deg kan forekomme?

Regneoppgaver

7. a) Bereng en Morlet wavelet (i tidsdomenet) for analysefrekvensen 250 og 750 Hz når samplingsfrekvensen er 5000 Hz og K -parameteren er 16. Plot resultatet med korrekte angivelser av tid på x-aksen (tilsvarer den figur lignende figur 13.2).
- b) Beregn den fouriertransformerte av hver av de to waveletene. Bruk både en FFT direkte på Morlet-waveleten beskrevet i tidsbildet, og ved å beregne den fouriertransformerte direkte ved hjelp av ligning (14.12). Plot resultatene med korrekte angivelser av frekvens på x-aksen.
- c) Kontroller at toppunktet kommer på det stedet du skulle forvente. Ser du speiling?
- d) Gjenta punkt a-c også når K -parameteren er 50.
8. I denne oppgaven er det underliggende temaet analogien til Heisenbergs uskarphetsfunksjon.
 - a) Generer en numerisk tallrekke som representerer signalet

$$f(t) = c_1 \sin(2\pi f_1 t) + c_2 \cos(2\pi f_2 t)$$

Bruk 10 kHz samplingsfrekvens og $N = 8192$ punkter, $f_1 = 1000$ Hz, $f_2 = 1600$ Hz, $c_1 = 1.0$, $c_2 = 1.7$. Signalet skal være hele tiden vi betrakter signalet. Plot et passe utsnitt av signalet i "tidsbildet" (utslag som funksjon av tid) slik at detaljer kommer fram. Pass på å få korrekte tall samt tekst langs aksenene, gjerne også en overskrift.

- b) Beregn den fouriertransformerte av signalet. Plott et passe utsnitt av signalet i "frekvensbildet" (velg gjerne absoluttverdier av fourierkoeffisientene som funksjon av frekvens), med tall og tekst langs aksene som ovenfor.
- c) Beregn den wavelettransformerte av signalet (kan godt ta utgangspunkt i programmene gitt i dette kapitlet, eller du kan skrive programmet mer eller mindre fra scratch selv). Bruk Morlet wavelets, og la analysefrekvensen gå f.eks. fra 800 til 2000 Hz (logaritmisk fordelt som vanlig innen wavelettransformasjon). Plot etter tur resultatet for bølgetallet K lik 24 og 200. Kommenter resultatet.
- d) La så signalet være et harmonisk signal som før, men nå konvolutert med en gaussisk funksjon slik at vi får to "bølgepakker":

$$f(t) = c_1 \sin(2\pi f_1 t) \exp(-[(t - t_1)/\sigma_1]^2) + c_2 \cos(2\pi f_2 t) \exp(-[(t - t_2)/\sigma_2]^2)$$

hvor $t_1 = 0.15$ s, $t_2 = 0.5$ s, $\sigma_1 = 0.01$ s og $\sigma_2 = 0.10$ s. Beregn den fouriertransformerte av signalet, og også den wavelettransformerte av signalet. Plot signalet i tidsbildet, frekvensbildet (passe utsnitt) og den wavelettransformerte av signalet for $K = 24$ og 100 (test gjerne flere verdier!) og øvrige parametre som i deloppgave c). Kommenter resultatene!

9. Analyser sangen til en svartrost ved hjelp av wavelettransformasjon. En lydfil er tilgjengelig fra kurswebsidene. Bruk programsnuttene i dette kapitlet (eller egen versjon), og analysér en tidsstrek på vel 1.4 s. Parametre ved analysen: Filnavn: 'Svartrost2.wav', Nstart = 17000, datastrengens lengde 64 k, frekvensområde 1500-8000 Hz, bølgetallet K lik 12 og 96 (og gjerne noen mellom disse verdiene også). Signalet består av fem ulike lydgrupper. Vi er først og fremst interessert i den fjerde av disse! Plot signalet i tidsbildet, frekvensbildet og signalet analysert ved wavelettransformasjon for denne fjerde lydgruppen. Pass på å ta med også noen utsnitt av de opprinnelige

plottene for å få fram detaljer. Dette gjelder i særdeleshet tidsbildet av den opprinnelige lyden! Forhåpentligvis vil du da gjenkjenne et signal vi har støtt på minst to ganger i tidligere kapitler. Du bør gjenkjenne hvordan vi kan lage et slikt signal matematisk. Nøyte analyse av den fjerde biten av lydsignalet i (1) tidsbildet og (2) etter wavelettransformasjon gjør det mulig å se hvordan en nær analogi til Heisenbergs uskarphetsrelasjon spiller inn på en praktfull måte! For å få fullt utbytte bør du hente ut tidsforskjeller og frekvensforskjeller i diagrammene og sammenholde disse med waveletens utbredelse i tids- og frekvensbildet for de to valgte K-verdiene.

Dersom du er student og har tilbud om hjelp fra lærere, anbefaler vi sterkt at du diskuterer de aktuelle detaljene med læreren inntil at du gjennomskuer samspillet vi ønsker å få fram. Det er mye verdifull kunnskap å hente fra dette problemet, kunnskap som også kan være verdifull i mange andre deler av fysikken!

10. Foreta en waveletanalyse av lydfilen "bokfink" på kurssets websider. Forsøk en K -faktor som er dobbelt så stor som i eksemplet i figur 14.13. Forsøk også en analyse med halvparten av K -verdien som ble brukt i den nevnte figuren. Beskriv forskjellene du ser.
11. Foreta waveletanalysen av bokfinklyden på ny for $K = 48$. Velg etter tur å bruke "power spectrum" (absoluttverdien etter omvendt fouriertransformasjon kvadrert), waveletanalyse "på amplitudenivå" (absoluttverdien etter omvendt fouriertransformasjon direkte) og waveletanalyse med kvadratroten av waveletanalysen (kvadratroten av absoluttverdien etter omvendt fouriertransformasjon). Gjør dine vurderinger av hvilken av disse metodene du liker best for akkurat dette signalet. Kan det hende at du for et annet signal ville foretrukket en av de andre anskuelsesformene (kvadrat, rett fram, eller kvadratrot-fremstillingene)?
12. Bruk kunnskapen fra kapittel 2 og 3 til å beregne tidsforløpet for en fjærpendel etter at den er satt i sving av en harmonisk kraft med frekvens lik resonansfrekvensen. Følg svingningene også en tid etter at den harmoniske kraften er fjernet. Foreta så en waveletanalyse av svingeforløpet. Forsøk å optimalisere analysen mhp K -verdi. Finner du en tilsynelatende sammenheng mellom Q-verdien for pendelsvingningen og K -verdien som gir optimalt waveletdiagram?
13. Velg selv en lydfil som du kan transformere til en .wav-fil, og velg et utsnitt som du kan ha lyst til å analysere. Optimaliser analysen, og fortell hvilken informasjon du får ut av diagrammet.
14. Finn data på nettet som viser en tidssekvens du synes kan være interessant å studere. Det kan være værdata, solflekker, strømforbruk eller hva du måtte finne på. Analysér datasettet både ved tradisjonel fouriertransformasjon og med waveletanalyse. Hvilken metode synes du egner seg best for de dataene du valgte? (Bør ha data med en eller annen form for løs periodositet med minst 20-30 perioder innenfor dataene du har tilgjengelig.)