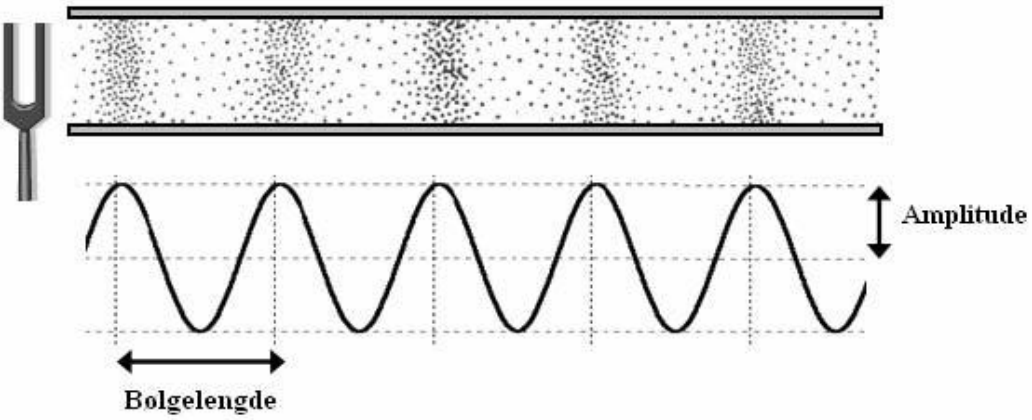
Digitalisering av lyd

Denne øvelsen er basert på materiale som Tore A. Danielsen utviklet som del av sin masteroppgave i fysikkdidaktikk. Arnt Inge Vistnes har også bidratt med ideer og diskusjoner.

Hva er lydbølger?

Lydbølger er fortetninger og fortynninger som forplanter seg i et stoff (for eksempel luft). Lydbølgene er longitudinelle bølger (”langsbølger”), det vil si at luftmolekylene svinger fram og tilbake i fartsretningen til bølgene.

Prikkene på figuren nedenfor representerer luftmolekyler i et rør, og vi ser at tettheten av molekylene varierer med posisjonen. I grafen under er tettheten på et gitt tidspunkt tegnet som en funksjon av posisjon i røret. Vi ser at tettheten har topper der lufttrykket er høyt og bunner der lufttrykket er lavt. En tilsvarende kurve ville vi fått hvis vi stilte oss opp i et bestemt punkt langs røret og målte lufttrykket som funksjon av tiden.



Figur 1: En lydbølge som forplanter seg i et rør.

Lydbølgen på figuren over er en ren sinusbølge. Det betyr at denne bølgen representerer en ”ren” tone, som vi for eksempel kan få fra en stemmegaffel. Dette er et spesialtilfelle. Lydbølgene vi lager når vi for eksempel snakker er mye mer komplekse. *Alle* lydbølger kan imidlertid skrives som en sum av sinusbølger med forskjellig frekvens.

Når vi hører en lyd, er det trommehinnene våre som oppfanger fortetninger og fortynninger i luften. Trommehinnene våre er små membraner som svinger i takt med lydbølgene. Disse svingningene blir omgjort til elektriske impulser som hjernen tolker som lyd. Dette er altså akkurat det motsatte av det som skjer i en høyttaler, der elektriske signaler gjør at høyttalermembranen settes i bevegelse for å lage lyd.

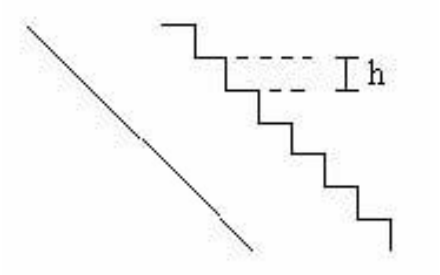
Det er **frekvensen** til lydsignalet som bestemmer tonehøyden vi hører. Frekvensen måles i antall svingninger per sekund, og enheten er hertz, Hz. 1 Hz = 1 svingning per sekund. Mennesker kan ideelt høre lydbølger med frekvenser i området fra ca 20 – 20 000 Hz, men i praksis hører vi sjelden lyder utenfor området 40 – 15 000 Hz. Jo eldre man blir, desto lavere blir den øvre grensen. Mørke/dype toner har lav frekvens, og lyse/høye toner har høy frekvens. Tenk på hvordan lyden endrer seg når du strammer en gitarstreng: en stram streng vibrerer raskere, og gir derfor lysere tone enn en slakk streng.

Hvor ”kraftig” lyden er henger sammen med **amplituden** (maksimalt utsving) til lydbølgene, altså hvor kraftige tetthetsvariasjonene i lufta er (se figur 1).

Hva betyr det at noe er digitalt?

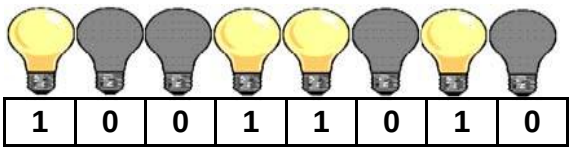
Ordet "digital" møter man ofte. Et digitalt system er et system som bare bruker **diskrete verdier**. I dette ligger det at data i et digitalt system bare kan ha én av et bestemt antall mulige verdier. I et analogt system har man et uendelig antall mulige verdier.

Figur 2 viser en trapp og en bakke med samme høydeforskjell fra bunn til topp. Vi kan se på bakken som et analogt system med kontinuerlige verdier, og trappa som et digitalt system med diskrete verdier. I bakken kan du velge i et kontinuerlig spektrum av høyder fra bunnen til toppen av bakken (uendelig mange forskjellige høyder), for eksempel 2,534·h. I trappa må du imidlertid stå på ett av trappetrinnene, så du kan bare befinne deg i en av høydene 0, h, 2h, …, 7h. Ingen andre høyder er mulige. Vi har altså et diskret utvalg av 8 høyder i trappa.



Figur 2: Diskret ("trapp") og kontinuerlig ("skråbakke") spektrum av høyder.

Datamaskiner, mobiltelefoner og Mp3-spillere er eksempler på digitale systemer. I et moderne digitalt system er all informasjon lagret som en serie ett-tall og nuller.



Figur 3: Totallsystemet. Hver bit (lyspære eller rute) kan bare ha to verdier (hhv av/på og 0/1).

Den minste enheten av informasjon (den minste lagringsplassen) i et digitalt system kalles en "bit". Hver bit kan kun ha to forskjellige verdier – en eller null. Vi kan tenke på hver bit som en lyspære som enten er på eller av (figur 3). Et tallsystem hvor hvert element kan ha to forskjellige verdier kalles 2-tallsystemet. I titallsystemet, som vi bruker til vanlig, kan hvert element (siffer) ha 10 forskjellige verdier, 0 – 9. Les gjerne mer i en matematikkbok eller på Internett om hvordan tall kan uttrykkes i totallsystemet.

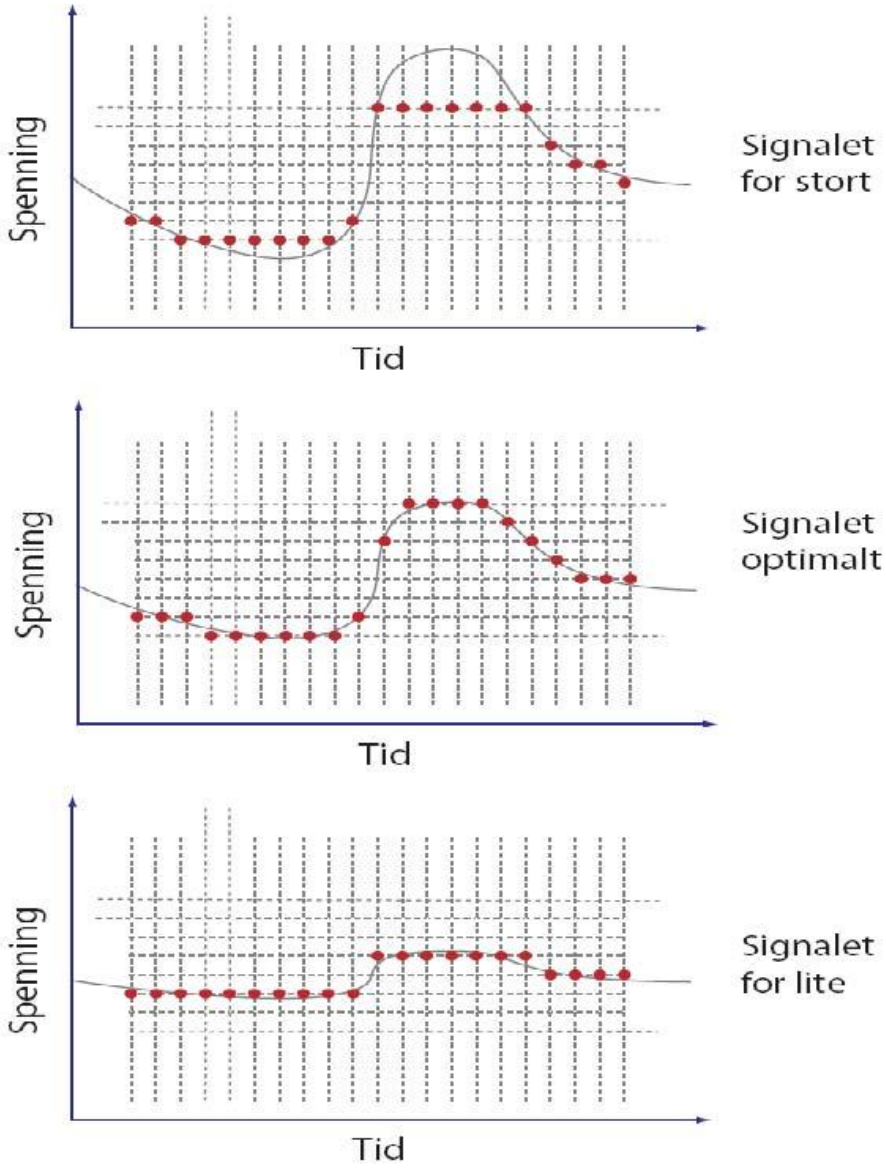
Vi kan si at å digitalisere et noe betyr å måle verdien av det til det nærmeste av de nivåene man har til rådighet (256 i eksempelet over). I moderne elektronikk lagres så denne målingen i totallsystemet, som enere og nuller.

Siden hver bit kan ha 2 forskjellige verdier, kan N bit representere 2^N forskjellige verdier. Sett at de 8 bit'ene (lyspærene) på figur 3 er lagringsplassen som brukes i et digitalt termometer. Da kan termometeret skille mellom $2^8 = 256$ forskjellige temperaturer. Er det et utetermometer vi skal lage, vil vi kanskje ønske å kunne måle temperaturer mellom -20 og $+30^o$ C. Da må de 256 nivåene fordeles over et temperaturintervall på 50^o C, og **oppløsningen** i termometeret blir på (50/256) = $0{,}2^o$ C. Skal vi i stedet lage et febertermometer, vil vi ønske å måle temperaturer mellom 36^o C og 41^o C. Da blir oppløsningen på $0{,}02^o$ C. I begge tilfeller har vi oppnådd tilfredsstillende oppløsning.

Hvordan lyd registreres digitalt – oppløsning, samplingsfrekvens og dynamisk område

Når lydsensoren på dataloggeren skal ta opp et lydsignal, måler (sampler) den lufttrykket på lydsensorens plass (utsvinget til lydsignalet) et gitt antall ganger i sekundet. Antall ganger lufttrykket måles i sekundet kaller vi **samplingsfrekvensen,** f_s**.** Dersom samplingsfrekvensen er 500 Hz, betyr det at lydsensoren tar 500 målinger per sekund. Hvordan lufttrykket er mellom to målinger vet vi imidlertid ingenting om.

For å få lydopptak som ligger nærmest mulig det lydsignalet vi tar opp, trenger vi høy samplingsfrekvens – vi vil ha minst mulig tidsrom mellom målingene.



Figur 4 (gjengitt med tillatelse av Arnt Inge Vistnes): Et signal som er digitalisert med oppløsning på 3 bits ($2^3 = 8$ mulige måleverdier). Her ser vi hvor viktig det er at signalet vi skal digitalisere "passer" til det måleområdet vi har valgt - at vi har et passende "dynamisk område". I den øverste figuren mister vi informasjon fordi måleområdet vårt (avstanden fra laveste til høyeste lovlige målenivå) er for lite. I den nederste figuren mister vi informasjon fordi måleområdet er altfor stort – vi bruker bare noen få av de mulige nivåene. I den midterste figuren ser vi imidlertid at måleområdet er akkurat passelig. Når vi skal digitalisere et lydsignal må vi være oppmerksom på dette og søke å utnytte de tilgjengelige målenivåene optimalt!

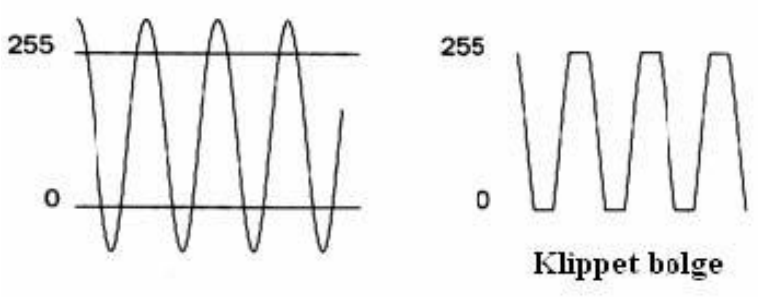
En digital lydsensor kan bare skille mellom et visst antall forskjellige lufttrykk (se figur 4). Hvor mange ulike lufttrykk en lydsensor kan skille mellom, avhenger av hvor mange bits oppløsning den har. Høy oppløsning betyr kort avstand mellom to mulige måleverdier, og lav oppløsning betyr stor avstand mellom to mulige måleverdier. Dersom vi har en lydsensor med 8 bits oppløsning, kan denne skille mellom $2^8 = 256$ forskjellige lufttrykk. Når vi måler lufttrykket med denne lydsensoren, kan vi derfor bare få en av 256 forskjellige verdier.

De forskjellige måleverdiene kan vi utnytte på forskjellige måter, avhengig av hvilket **dynamisk område** vi velger – hvor vi legger hhv øvre og nedre nivå av de (i vårt eksempel) 256 nivåene vi har til rådighet. I termometer-eksemplene over valgte vi dynamiske områder på hhv -20 -- $+30^o$ C og $36 – 41^o$ C.

Nøyaktigheten til hver måling synker altså når det dynamiske området blir større. Som vi ser av figur 4 er det svært viktig at det dynamiske området til sensoren passer til det signalet som skal digitaliseres, slik at vi får med oss mest mulig informasjon i målingene vi gjør.

Klipping

Dersom deler av lydsignalet vi skal digitalisere ligger utenfor det dynamiske området vi har valgt, får vi et fenomen som kalles **klipping** (se figur 5). Den delen av signalet som ligger utenfor måleområdet blir "klippet" bort fordi vi ikke kan måle så store utsving.



Figur 5: Her skal det digitaliserte signalet kodes i 8 bits. Det dynamiske området er for lite, slik at toppene og bunnene i signalet faller utenfor de 256 (0 t.o.m. 255) nivåene vi kan måle. Da sier vi at toppene og bunnene klippes bort.

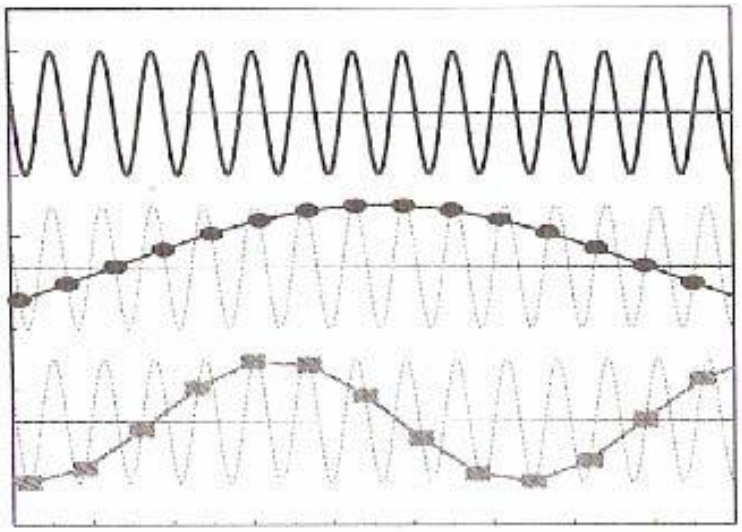
Aliasing

Hvor høye frekvenser vi kan måle med en sensor avhenger av hvor stor samplingsfrekvens vi bruker. Når vi skal digitalisere et signal med frekvens f må vi velge en samplefrekvens, f_s, slik at:

$$f_s > 2f$$

Frekvensen må være så stor fordi vi trenger mer enn to målinger på hver bølgelengde for at digitaliseringen skal gi meningsfull informasjon om lydsignalet[1]. Hvis samplingsfrekvensen er *litt* mindre enn dette, vil det digitaliserte signalet synes å ha en frekvens $f_s - f$. Dette kaller vi et *"alias"* av det opprinnelige lydsignalet (se figur 6).

[1] Den høyeste frekvensen som kan måles med et system med samplingsfrekvens f_s kalles Nyquistfrekvensen



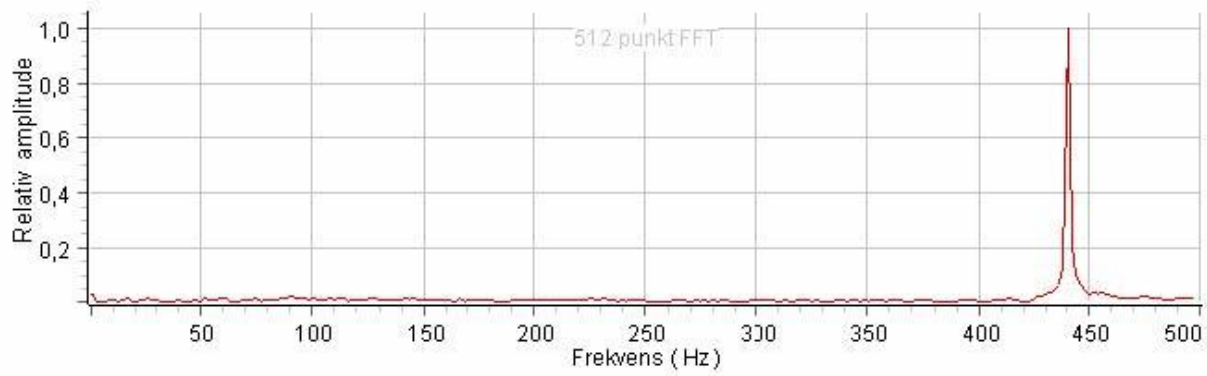
Figur 6: De to nederste grafene viser digitaliseringer av et signal (øverst) der samplingsfrekvensen er mindre enn dobbelt så stor som signalfrekvensen.

Den øverste kurven på figur 6 er det signalet vi skal digitalisere. På begge de to nederste figurene ser vi at samplingsfrekvensen er for lav, slik at målingene våre ser ut til å komme fra et signal med mye lavere frekvens enn det egentlige signalets frekvens – vi får et "alias".

Legg merke til at alle målepunktene selvfølgelig ligger på det opprinnelige signalet. Vi har selv gjettet hvordan signalet ser ut mellom målingene (vi vet jo ikke noe om hvordan signalet oppfører seg mellom de målingene vi har gjort). Dette har vi gjort ved å trekke rette linjer mellom målepunktene. Det samme vil programmet Data Studio gjøre med et signal fra lydsensoren.

FFT

Som nevnt kan alle komplekse signaler skrives som en sum av sinussignaler med ulik frekvens. FFT (eng: Fast Fourier Transformation) er en matematisk operasjon som finner alle frekvenskomponentene i et signal. I DataStudio kan vi bruke denne operasjonen ved å velge funksjonen FFT. Da finner denne funksjonen alle frekvensene i lydsignalet vi har digitalisert, og plotter hvor sterkt hver frekvenskomponent er representert. Det betyr at jo sterkere en frekvens er til stede i lydsignalet – desto høyere blir toppen til denne frekvensen i fourierspektret til lydsignalet. Dersom lydsignalet vårt er en enkel sinusbølge, har det bare én frekvens – altså får vi bare én topp i FFT- vinduet. Hvis vi derimot har et signal med flere frekvenskomponenter, vil vi få flere topper i FFT- vinduet.



Figur 7: Dette er grafen man får ut når man bruker FFT – funksjonen i DataStudio. Vi ser at det bare er én frekvens i lydsignalet som er tatt opp, og at denne frekvensen er ca 440 Hz. X-aksen går opp til 500 Hz fordi samplingsfrekvensen er 1 000 Hz – vi kan måle frekvenser opp til halvparten av samplingsfrekvensen (se "Alias").

Hvorfor har MP3- formatet blitt så populært?

MP3 har de siste årene blitt et svært populært lydformat. Årsaken til dette er at en sang i MP3 – format tar vesentlig mindre plass enn en tilsvarende sang i cd-kvalitet. På en cd er det 16 bits koding av lyden. Det betyr at det på en cd kan registreres 2^{16} = 65 536 ulike verdier (cd-er har en oppløsning på 65 536 bits). Når man spiller inn lyd på cd bruker man samplingsfrekvens på 44 100 Hz for at man skal få med alle frekvenser innenfor det området som er hørbart for mennesker (samplingsfrekvensen må som tidligere nevnt være minst dobbelt så stor som den høyeste frekvensen i lyden vi skal ta opp). I tillegg er det to kanaler – en til hver høyttaler - for å få stereolyd. Mengden digitale data som trengs for å lagre cd – kvalitet blir derfor:

Bitrate: 44 100 målinger/sekund * 16 bits/måling * 2 kanaler = 1 411,2 kbit/s

I MP3-format, derimot, er det mest vanlig med bitrater på 32 - 320 kbit/s, avhengig av hvilken kvalitet man ønsker. 128 kbit/s er for eksempel den bitraten som vanligvis brukes til musikkfiler på internett. Det vil si at dersom man gjør om sanger man har på cd til MP3- filer med kvaliteten 128 kbit/s, så vil sangene bare trenge 1/11 av lagringsplassen de opptok på cd'en! Man kan altså få mange flere sanger inn på samme lagringsplass dersom man gjør om sangene til MP3- format.

Men noe går jo tapt når man forminsker en fil så mye. Når lydfiler formateres i MP3- format, gjøres det imidlertid noen grep for at lydkvaliteten ikke skal minke så mye som reduksjonen i størrelse på lydfilen skulle tilsi. Her er noen eksempler på dette:

- Man fjerner frekvenser som ligger utenfor det hørbare området
- Når to eller flere lyder spilles av samtidig og en lyd er (mye) kraftigere enn de andre hører man bare den kraftigste. Derfor kan man droppe de andre.
- Dersom to lyder er svært like i frekvens (høres svært like ut) kan den ene sløyfes uten at det påvirker lyden dramatisk.
- Det er vanskelig å høre hvor en lyd kommer fra ved høye og lave frekvenser, og derfor kan man kode disse frekvensene i mono uten å tape særlig kvalitet.

Det man står igjen med etter at dette er gjort komprimeres deretter til den størrelsen man skal ha.

Slik gjør du - Veiledning til dataloggerforsøk om digitalisering av lyd

I dette forsøket skal du bruke en datalogger med lydsensor til å digitalisere lyden fra en stemmegaffel, og du skal studere ulike problemer og fenomener knyttet til digitalisering av lyd. Sentrale begreper du skal innom er samplingsfrekvens, oppløsning, dynamisk område, klipping og aliasing, og fouriertransformasjon.

Du trenger:

- En Pasco datalogger med lydsensor
- En stemmegaffel med tonen a_1, frekvens 440 Hz.

Gjør:

- Først kobler du sammen datalogger og pc.
- Deretter kobler du til lydsensoren og starter programmet DataStudio (DS).
- Velg"Lydsensor".
- Du får nå opp et ikon for lydsensoren oppe til venstre i DS- vinduet[2]. Klikk og dra dette ikonet ned til der det står "graf".
- Nå har du fått opp et grafvindu som viser spenning (fra lydsensoren) som funksjon av tiden.
- Gå inn på "oppsett" (oppe til venstre i DS- vinduet). Sett målefrekvensen til 2 000 Hz, og velg følsomhet lav.
- Slå hammeren mot stemmegaffelen, og trykk på start rett etterpå. Trykk stopp etter 3-5 sekunder.
- Zoom inn på et område av grafen du nå har fått opp og se nærmere på den.

På grafen du får er hver måling markert med et punkt, og det er trukket rette linjer mellom disse punktene. Egentlig har dataloggeren bare målt lufttrykket på lydsensorens plass ved hvert av de tidspunktene målingene er gjennomført. Vi vet ikke noe om hvordan lufttrykket endrer seg mellom målingene, men programmet DataStudio "gjetter" at trykket øker eller minker lineært mellom hvert målepunkt. Du kan fjerne linjene mellom datapunktene ved å klikke på den lille pilen ved siden av ikonet helt til høyre på verktøylinjen rett over grafvinduet. På menyen fjerner du markeringen for "connected lines".

Gjør:

- Ta en ny måling med samplefrekvens på 10 000 Hz. Zoom inn og se nærmere på denne grafen også. Hva er forskjellen på de to grafene du har fått til nå? Hvorfor er det sånn? Kan du telle perioder og anslå frekvensen til lydbølgene?
- Dra ikonet for måleserien ned til "FFT" og slipp. Du får opp et fourierspektrum av signalet. Stemmer det med ditt anslag ut fra telling av perioder?

Under avsnittet om hvordan lydsignaler digitaliseres i teoriheftet står det at samplefrekvensen må være minst dobbelt så stor som frekvensen til signalet som digitaliseres. Vi skal nå se hva som skjer dersom vi prøver å bruke lavere samplefrekvens enn dette.

[2] Sensoren oppgir en spenning (i volt) som er proporsjonal med utsvinget til lydsignalet, dvs lufttrykket.

Gjør:

- Under "Setup", velg samplingsfrekvens 500 Hz.
- Ta opp et signal fra stemmegaffelen igjen. Hvordan ser grafen ut? Kan du telle perioder og anslå frekvensen? Kommentar??
- Klikk på ikonet for måleserien, dra det ned på FFT-ikonet og slipp.
- Les av frekvensen til stemmegaffelen i FFT-vinduet. Stemmer den med ditt frekvensanslag ut fra telling av perioder?
- Hva kalles fenomenet vi observerer her?

Utfordring: Klarer du, ut fra det som står i teoriheftet under "Aliasing", å forklare hvorfor du får akkurat denne frekvensen?

Gjør:

- Ta et nytt opptak av lyden fra stemmegaffelen med samplingsfrekvens 10 000 Hz, men denne gangen velger du høy følsomhet.
- Zoom inn på grafen. Hva ser du? Forklar hva som har skjedd!
- Still sensoren på lav følsomhet igjen (i Setup-vinduet) og velg samplingsfrekvens 10 000 Hz. Forsøk å synge en énstrøken a (bruk stemmegaffelen til å finne tonen!) og studér signalet. Hvordan vil du beskrive det?
- Klikk fram FFT-vinduet, syng igjen, og undersøk hvilke frekvenser som er til stede. Hva merker du deg ved toppene i frekvensspektret? (Hint: "overtoner", "høyere harmoniske").

Konkurranser til slutt:

- Hvem klarer å synge den mørkeste tonen (lavest frekvens)?
- Hvem klarer å synge den lyseste tonen (høyest frekvens)?
- Hvem har størst forskjell (i Hz) mellom den lyse og den mørke ton en? Hvem klarer å synge en a med frekvens nærmest 440 Hz?

Konklusjon:

Hva er det viktig å passe på i forhold til samplingsfrekvens og oppløsning når vi skal digitalisere et lydsignal? Hva kan vi bruke fourier-sepktre til?