TTT4280 Sensorer og instrumentering
Laboppgave 2 Akustikk: Mikrofonarray for å finne retning til en lydkilde

En mikrofon er en sensor som omformer et lydsignal til en elektrisk spenning: utgangssignalet er proporsjonalt med lydtrykket ved mikrofonens membran over et bredt frekvensområde. Mikrofoner er «*ubiquitous*», «allestedsnærværende», da alt utstyr for talekommunikasjon bruker mikrofoner, og telefoner står derfor for et enormt marked for mikrofoner.

Den absolutt vanligste mikrofontypen er omnidireksjonal, men anvendelsesområdet for mikrofoner øker betraktelig når man kobler sammen mer enn én mikrofon i et «array». Med et array kan man f.eks lage en direktiv mikrofon hvilket kan gi mye bedre signalkvalitet ved at lyd fra uønskede retninger undertrykkes. I tillegg kan man detektere hvor en lydbølge kommer fra, og dette er temaet for denne laboppgaven.

Det finnes ulike måter å detektere hvor en lydbølge kommer fra. En rett fram måte er å bruke krysskorrelasjon mellom par av mikrofonsignaler for å finne ut tidsforsinkelsen. Dette kan da brukes for å finne retning til en lydkilde. Det kan også finnes begrensede muligheter for å finne posisjon, og ikke kun retning for en lydkilde.

I denne oppgaven skal dere lage et array med tre små omnidireksjonale mikrofoner som skal monteres slik at dere kan detektere innfallsretningen i et horisontalt plan.

Se teorikapittelet for gjennomgang av hvordan man bestemmer effektiv tidsforsinkelse ved hjelp av krysskorrelasjon, og hvordan man finner retning ut fra effektiv tidsforsinkelse.

Innhold

1 Teori **2**
- 1.1 Krysskorrelasjon . . . 2
- 1.2 Deteksjon av innfallsretning . . . 4
- 1.3 Hva om lydkilden ikke er langt vekk (dvs ikke en planbølge)? . . . 8

2 Forberedelsoppgaver til laben **10**
- 2.1 Forberedelsesoppgave 1 . . . 10
- 2.2 Forberedelsesoppgave 2 . . . 10

3 Laboppgave **11**
- 3.1 Krav for godkjent lab . . . 11

4 Appendix - løsning av lign. (9) via matriseinvertering **12**

4.1 Numerisk eksempel 12

1 Teori

1.1 Krysskorrelasjon

Hvis vi har to diskrete tidssignaler, $x(n)$ og $y(n)$, så er krysskorrelasjonsfunksjonen, r_{xy}, definert som*

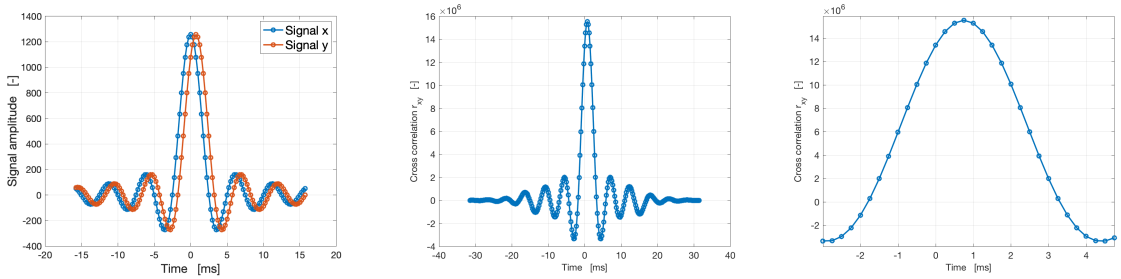
$$r_{xy}(m) = \sum_{n=-\infty}^{\infty} x(n)y(n+m), \quad m = 0, \pm 1, \pm 2, \pm 3, \ldots \tag{1}$$

Denne funksjonen gir altså et tall, r_{xy}, for hver verdi på tidsforskyvingen m (uttrykkt i antall sampler). Hvis krysskorrelasjonsfunksjonen har en høy verdi så betyr det at de to signalene er meget like; en lav verdi betyr at signalet har et stokastisk forhold - for noen n er produktet $x(n+m)y(n)$ positivt; for andre n er det negativt.

Retningen har betydning: hvis x er meget lik y, men x er forskyvet m sampler til venstre, så vil $r_{xy}(m)$ gi en høy verdi. Og, hvis x er meget lik y, men x er forskyvet m sampler til *høyre*, så vil $r_{xy}(-m)$ å gi en høy verdi.

Her har vi da grunnideen for å finne en tidsforskyving mellom to signaler: beregn $r_{xy}(m)$, og finn for hvilken verdi på m som $|r_{xy}(m)|$ har et maksimum! Tidsforsinkelsen gis da av $\Delta t = m/f_s$, hvor f_s er samplingsfrekvensen.

I figur 1a ser vi et eksempel på to identiske signaler, hvor x er forskyvet til venstre, med eksakt tre sampler. Samplingsfrekvensen er 4000 Hz, hvilket tilsvarer at det er 0,25 ms mellom hvert sampel. I figur 1b ser vi krysskorrelasjonsfunksjonen, og i figur 1c en forstørret versjon. I den siste figuren, 1c, kan vi se en topp ved tidsforskyvingen 0,75 ms. Symmetrien omkring toppen gjør at vi innser at maks av r_{xy} skal være på nøyaktig 0,75 ms.



(a) (b) (c)

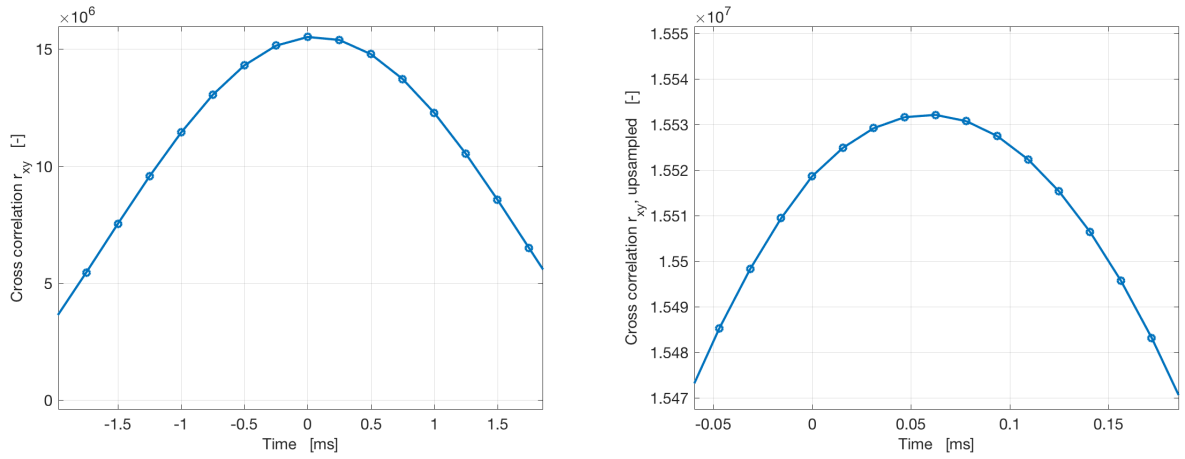
Figur 1: (a) To signaler som er identisk, men forskyvet med eksakt tre sampler. (b) Krysskorrelasjonsfunksjonen mellom de to signalene i (a). (c) En forstørret versjon av diagrammet i (b).

Nå ser vi på et eksempel til, hvor vi har laget en forsinkelse som er på 0,25 sampel (dvs 0,25·0,25 ms = 62,5 μs)! Vi viser kun den forstørrede versjonen i figur 2a, og innser at når vi søker maks, så

*I noen bøker er denne funksjonen definert motsatt, slik at «$n+m$» står i argumentet for x. I Python er funksjonen `numpy.correlate` definert på denne alternative måten.

kommer vi ikke å kunne avlese toppverdien bedre enn hele sampelverdier. Da får vi altså en feil i deteksjonen av effektiv tidsforsinkelse som er maks. ±0,5 sampel = ±0,5·0,25ms = 125 μs.

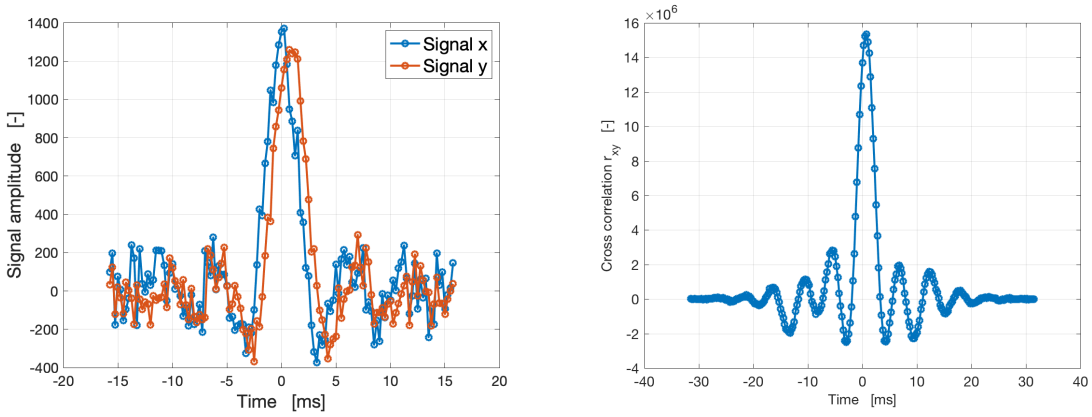
Det er dog mulig å oppsample signalet, og figur 2b viser at med 16 gangers oppsampling, til f_s = 64 kHz, så kan vi avlese toppen med en feil på ±0,5 sampel = 7,8 μs.



(a) (b)

Figur 2: (a) Krysskorrelasjonsfunksjonen mellom to signaler som er identisk, foruten en forskyving på 0,25 sampler, f_s = 4000 Hz. (b) En oppsamplet versjon av funksjonen i (a), med Pythons `numpy.interp`-funksjon og 16 gangers oppsampling, dvs f_s=64000 Hz.

Et siste eksempel: nå går vi tilbake til eksemplet i figur 1, men legger mye (gaussisk) støy på begge signalene, se figur 3 (det er uavhengige støysignaler som har blitt lagt på de to signalene). Hvis man zoomer inn rundt toppen i figur 3b, finner man at toppen fortsatt er på 3 sampler. Med litt mer støy, så hadde toppen kunnet vippet over til 2 eller 4 sampler. Det kan hjelpe å båndpassfiltrere signalet (før man gjør krysskorrelasjonen), men kun om nyttesignalet har et mer smalbåndet spektrum enn støysignalet!

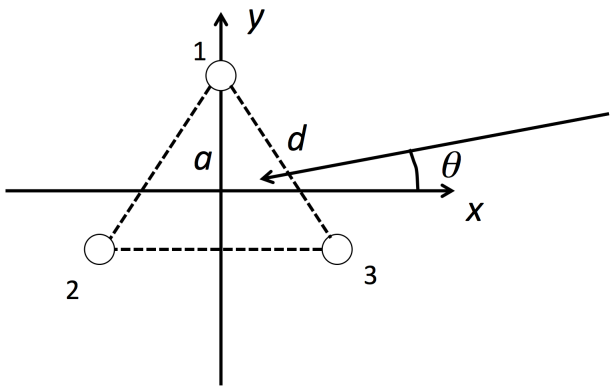


(a) (b)

Figur 3: (a) To signaler som er identisk, men forskyvet med eksakt tre sampler, og med gaussisk støy addert. (b) Krysskorrelasjonsfunksjonen mellom de to signalene i (a).

1.2 Deteksjon av innfallsretning

Vi tenker oss tre mikrofoner plassert som i figur 4.



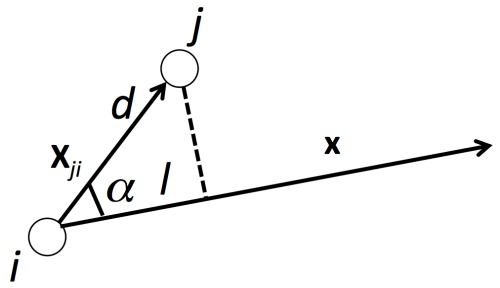
Figur 4: Et array med tre mikrofoner/sensorer plassert i et likesidet triangel, med sidelengde d.

En innfallende plan lydbølge med vinkelen θ mot x-aksen vil gi forskjellig ankomsttid til de tre sensorene. Vi er spesielt interessert i *tidsdifferansen* mellom ankomsttidene til hvert av de tre parene av sensorer. Derfor ser vi på et eneste par, med sensor i og j, som i figur 5.

Der innfører vi to vektorer:

- $\mathbf{x}$ = den ukjente vektoren for innfallende bølge. NB: propageringsretningen for bølgen = $-\mathbf{x}$.
- $\mathbf{x}_{ji}$ = vektoren fra sensor i til sensor j ($\mathbf{x}_{ji} = \mathbf{x}_{\text{sensor},j} - \mathbf{x}_{\text{sensor},i}$); denne vektoren er kjent, og har lengden d.

Vi antar at vi har gjort en måling og funnet ut forsinkelsen fra sensor i til sensor j, τ_{ji} (med krysskorrelasjon). Den forsinkelsen tilsvarer en gangveisforskjell, l: $\tau_{ij} = -l/c$ hvor lengden l vises i



Figur 5: Et par av sensorer, og en innfallende bølge som propagerer i retning -$\mathbf{x}$ (som er ukjent). Vektoren $\mathbf{x}_{ji}$ går fra sensor i til sensor j.

figur 5. Det blir et minustegn fordi vi har en bølge som utbreder seg i retningen $-\mathbf{x}$, og da må τ_{ji} (tiden fra sensor i til sensor j) være negativ.

Lengden l er altså kjent, og henger sammen med den ukjente vinkelen α som

$$l = d\cos\alpha = |\mathbf{x}_{ji}|\cos\alpha \tag{2}$$

dvs vi kan uttrykke den ukjente vinkelen som

$$\cos\alpha = \frac{l}{d} = \frac{l}{|\mathbf{x}_{ji}|} = \frac{-c\tau_{ji}}{|\mathbf{x}_{ji}|} \tag{3}$$

Med to sensorer kan vi tydeligvis finne vinkelen α - men den vinkelen er tvetydig fordi det er to innfallsvinkler som oppfyller ligning (3)*. I figur 5 kan vi «speile» vektoren $\mathbf{x}$ rundt vektoren $\mathbf{x}_{ji}$ (slik at $\mathbf{x}$ peker nesten rett opp), og vinkelen mellom vektorene $\mathbf{x}$ og $\mathbf{x}_{ji}$ vil fortsatt være α. For å finne hvilken av de to mulige løsningene som er riktig så trenger vi en sensor til. Da viser det seg å være praktisk å finne vektoren $\mathbf{x}$ heller enn vinkelen α (fordi $\mathbf{x}$ vil være felles for alle tre mikrofonparene, men vinkelen α vil være forskjellig), og vi kan finne forholdet mellom dem fra prikkproduktet for de to vektorene:

$$\mathbf{x}_{ji}\cdot\mathbf{x} = |\mathbf{x}_{ji}||\mathbf{x}|\cos\alpha \tag{4}$$

Setter vi nå inn (3) i (4) så får vi

$$\mathbf{x}_{ji}\cdot\mathbf{x} = |\mathbf{x}_{ji}||\mathbf{x}|\frac{(-c\tau_{ji})}{|\mathbf{x}_{ji}|} \tag{5}$$

$$\Rightarrow c\tau_{ji} = -\frac{\mathbf{x}_{ji}\cdot\mathbf{x}}{|\mathbf{x}|} \tag{6}$$

Nå kan vi velge å definere $\mathbf{x}$ slik at $|\mathbf{x}| = 1$: $\mathbf{x} = [\cos\theta, \sin\theta]$ (se figur 4) og da er

$$\mathbf{x}_{ji}\cdot\mathbf{x} + c\tau_{ji} = 0 \tag{7}$$

Denne likningen kunne vi skrevet ut som

$$x_{ji}\cos\theta + y_{ji}\sin\theta + c\tau_{ji} = 0 \tag{8}$$

*Hvis vi har kun to sensorer så vil vinkelen α i 3D tilsvare ikke bare to vinkler men en *kon* av innfallsvinkler.

og her har vi jo én ukjent og burde kunne løse likningen - men den gir, som kommentert over, ikke en entydig løsning. To innfallsretninger, speilet rundt vektoren $\mathbf{x}_{ji}$ gir samme tidsforsinkelse τ_{ji}!

Derfor setter vi opp tre sensorer som i figur 4, og innfører tre vektorer fra sensor til sensor: $\mathbf{x}_{21}$, $\mathbf{x}_{31}$, $\mathbf{x}_{32}$ og tre tilsvarende forsinkelser: τ_{21}, τ_{31}, τ_{32}. Hvis vi nå setter opp tre likninger som i (7), får vi i stedet et overbestemt ligningssystem (tre likninger og én ukjent, θ):

$$\mathbf{x}_{ji} \cdot \mathbf{x} + c\tau_{ji} = 0, \quad ji = 21, 31, 32 \tag{9}$$

Overbestemte systemer er faktisk bra fordi da kan vi håndtere at:

- våre målinger av τ_{ji} ikke er perfekte (støy, andre målefeil, mer enn en bølge,...)
- innfallende bølge kanskje har en innfallsretning som ikke er i samme plan som sensorene!

Men hvordan skal vi løse ligningssettet (9)? Jo, med minste kvadraters metode, «*Least mean squares*». På samme måte som når vi utleder linjetilpassing til et sett med datapunkter så innfører vi et feilmål, ε, som forteller hvor dårlig vårt estimat av x, y passer til måleverdiene τ_{21}, τ_{31}, τ_{32}. Det målet definerer vi slik (idealt skal ε være 0):

$$\varepsilon^2 = \sum_{ji} (\mathbf{x}_{ji} \cdot \mathbf{x} + c\tau_{ji})^2 = \sum_{ji} (x_{ji}x + y_{ji}y + c\tau_{ji})^2 \tag{10}$$

Målet er å finne de to verdiene x og y slik at ε^2 minimeres! Her er da den søkte vinkelen θ skjult i x og y slik at på slutten, når vi har funnet x og y så får vi θ som:

$$\tan\theta = \frac{y}{x}, \quad \theta = \operatorname{atan}\frac{y}{x} \tag{11}$$

For å minimere ε^2 så skal vi da derivere ε^2 med hensyn på x, og med hensyn på y, og sette de deriverte til 0:

$$\begin{cases} \frac{\partial \varepsilon^2}{\partial x} = \frac{\partial}{\partial x}\sum_{ji}(x_{ji}x + y_{ji}y + c\tau_{ji})^2 = \sum_{ji} 2(x_{ji}x + y_{ji}y + c\tau_{ji})x_{ji} = 0 \\ \frac{\partial \varepsilon^2}{\partial y} = \frac{\partial}{\partial y}\sum_{ji}(x_{ji}x + y_{ji}y + c\tau_{ji})^2 = \sum_{ji} 2(x_{ji}x + y_{ji}y + c\tau_{ji})y_{ji} = 0 \end{cases} \implies \tag{12}$$

$$\begin{cases} x\sum_{ji} 2(x_{ji}^2) + y\sum_{ji} 2(x_{ji}y_{ji}) + \sum_{ji} 2(c\tau_{ji}x_{ji}) = 0 \\ x\sum_{ji} 2(x_{ji}y_{ji}) + y\sum_{ji} 2(y_{ji}^2) + \sum_{ji} 2(c\tau_{ji}y_{ji}) = 0 \end{cases} \implies \begin{cases} xA + yB + C = 0 \\ xB + yD + E = 0 \end{cases} \tag{13}$$

Dette kan man løse:

$$x = \frac{CD - BE}{AD - B^2}, \quad y = \frac{-BC + AE}{AD - B^2} \tag{14}$$

Nå kan vi sette inn verdier på A, B, C, D og E fordi vi kan se i figur 4 hvor sensorene plasseres:

$$\mathbf{x}_{\text{sensor},1} = [0, 1]a \tag{15}$$

$$\mathbf{x}_{\text{sensor},2} = [-\sqrt{3}/2, -0.5]a \tag{16}$$

$$\mathbf{x}_{\text{sensor},3} = [\sqrt{3}/2, -0.5]a \tag{17}$$

slik at vektorene $\mathbf{x}_{ji}$ blir

$$\mathbf{x}_{21} = \mathbf{x}_{\text{sensor},2} - \mathbf{x}_{\text{sensor},1} = \left[-\frac{\sqrt{3}}{2}, -\frac{3}{2}\right] a \tag{18}$$

$$\mathbf{x}_{31} = \mathbf{x}_{\text{sensor},3} - \mathbf{x}_{\text{sensor},1} = \left[\frac{\sqrt{3}}{2}, -\frac{3}{2}\right] a \tag{19}$$

$$\mathbf{x}_{32} = \mathbf{x}_{\text{sensor},3} - \mathbf{x}_{\text{sensor},2} = \left[\sqrt{3}, 0\right] a \tag{20}$$

Da får vi

$$A = 2\sum (x_{ji})^2 = 9a^2 \tag{21}$$

$$B = 2\sum (x_{ji} y_{ji}) = 0 \tag{22}$$

$$D = 2\sum (y_{ji})^2 = 9a^2 \tag{23}$$

Symmetrien gjorde tydeligvis at $B = 0$, så vi får enklere uttrykk enn i (14):

$$x = \frac{C}{A}, \quad y = -\frac{E}{D} \tag{24}$$

og når vi setter inn uttykkene for A, C, D og E:

$$x = c\frac{\sum x_{ji}\tau_{ji}}{9a^2/2} = \frac{2c}{9a^2}\sum x_{ji}\tau_{ji}, \quad y = -c\frac{\sum y_{ji}\tau_{ji}}{9a^2/2} = -\frac{2c}{9a^2}\sum y_{ji}\tau_{ji} \tag{25}$$

Her kan vi sette inn verdier for x_{ji} og y_{ji} for vår spesifike arraygeometri, fra lign. (18)-(20):

$$x_{21} = -\frac{\sqrt{3}}{2}a, \quad x_{31} = \frac{\sqrt{3}}{2}a, \quad x_{32} = \sqrt{3}a$$

$$y_{21} = -\frac{3}{2}a, \quad y_{31} = -\frac{3}{2}a, \quad y_{32} = 0$$

og da blir

$$x = \frac{2c}{9a}\left(-\frac{\sqrt{3}}{2}\tau_{21} + \frac{\sqrt{3}}{2}\tau_{31} + \sqrt{3}\tau_{32}\right) = \frac{c}{3\sqrt{3}a}\left(-\tau_{21} + \tau_{31} + 2\tau_{32}\right) \tag{26}$$

$$y = -\frac{2c}{9a}\left(-\frac{3}{2}\tau_{21} - \frac{3}{2}\tau_{31}\right) = \frac{c}{3a}\left(\tau_{21} + \tau_{31}\right) \tag{27}$$

Så setter vi inn disse uttrykkene for x og y i (11):

$$\theta = \text{atan}\frac{\frac{c}{3a}\left(\tau_{21} + \tau_{31}\right)}{\frac{c}{3\sqrt{3}a}\left(-\tau_{21} + \tau_{31} + 2\tau_{32}\right)} = -\text{atan}\left(\sqrt{3}\frac{\tau_{31} + \tau_{21}}{\tau_{31} - \tau_{21} + 2\tau_{32}}\right) \tag{28}$$

Vi må være litt forsiktig med atan-funksjonen fordi $\text{atan}(1/1) = \text{atan}(-1/-1)$, hvilket leder til at atan kun gir oss vinkler i intervallet $\theta \in [-90°, 90°]$. Hvis vi ønsker å detektere vinkler fra alle retninger må vi gå tilbake til uttrykkene for x og y i lign. (26) og (27): hvis $x < 0$, skal vi legge til π til θ-estimatet fra lign. (28). Vi kan notere at fortegnet på x gis av fortegnet på $(-\tau_{21} + \tau_{31} + 2\tau_{32})$.

I en praktisk implementering vil vi bruke samplede signaler, og tidsforsinkelsene, τ_{ji}, vil beregnes fra topper i korrelasjonsfunksjoner ved hele antall sampler, n_{ji} slik at $\tau_{ji} = n_{ji}/f_S$. Da kan vi like gjerne bruke forsinkelser uttrykt direkte i sampler:

$$\theta = \text{atan}\left(\sqrt{3}\frac{n_{31} + n_{21}}{n_{31} - n_{21} + 2n_{32}}\right) \tag{29}$$

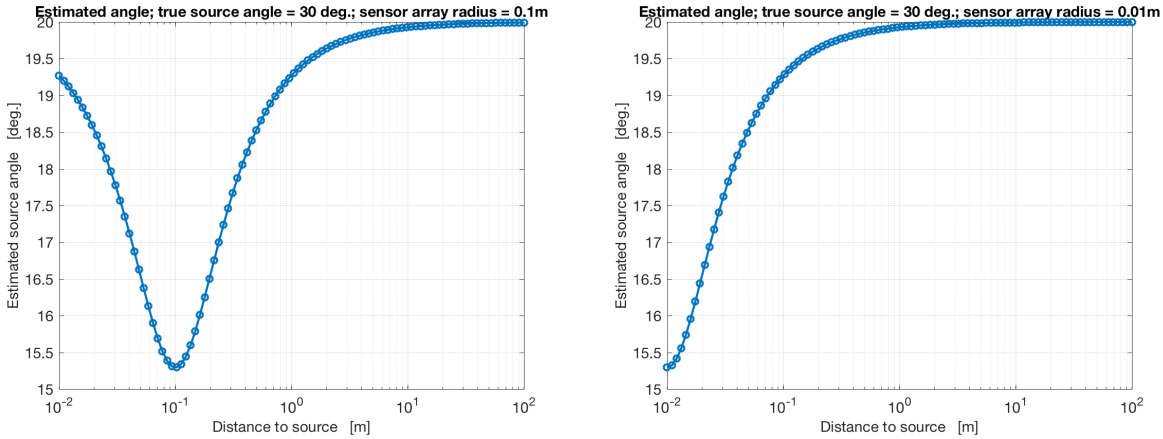
Igjen må vi legge til π til θ-estimatet hvis $x < 0$, dvs hvis $(-n_{21} + n_{31} + 2n_{32}) < 0$. Vi kan også innse at heltallene n_{ji} kun kan ha et begrenset antall forskjellige verdier: $[\pm n_{max}, \pm(n_{max} - 1), ...0]$. Det betyr at kun et begrenset antall vinkler kan estimeres av lign. (29)! Vi kan få en høyere verdi på n_{max} ved å øke størrelsen på arrayet, d, og/eller ved å øke samplingsfrekvensen, f_S.

1.3 Hva om lydkilden ikke er langt vekk (dvs ikke en planbølge)?

La oss nå se på hvordan uttrykket i lign. (28) fungerer hvis kilden ikke er langt vekk. Vi kan gjøre en enkel evaluering ved å plassere en kilde i forskjellig avstand, r_{source}, men altid med vinkel θ, dvs

$$\mathbf{x}_{\text{source}} = r_{\text{source}}[\cos\theta, \sin\theta] \tag{30}$$

Da er det rett frem å beregne τ_1, τ_2 og τ_3 som funksjon av r_{source}. Vi har gjort det for en kilderetning $\theta = 20$ grader, og to forskjellige sensorarraystørrelser, og de vises i figur 6.

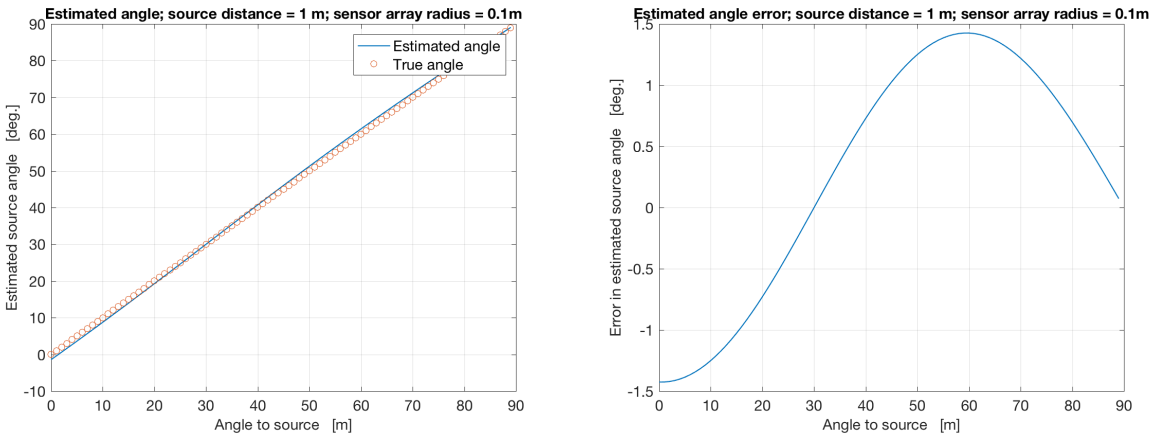


Figur 6: Estimert vinkel til en lydkilde i virkelig retning 20 grader (tittelen på grafen er feil: 30 grader skal altså være 20 grader).

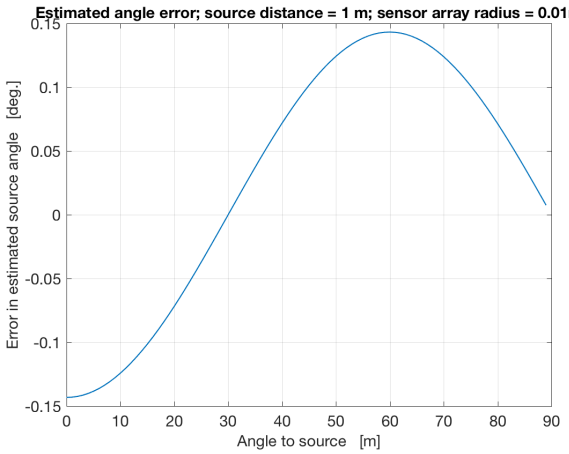
Så, hvis kilden er minst 1 meter fra sentrum av sensorarrayet så er feilen i kildevinkelestimatet mindre enn 0.8 grader for 10cm-arrayet, og mindre enn 0.1 grader for 1cm-arrayet.

Vi kan også se på feilen på en annen måte; hvor stor er feilen for forskjellige retninger til den virkelige kilden? Vi velger å studere avstanden fra kilden til 1 m, og med arrayet som har $a = 10$ cm. Resultatene vises i figur 7.

Feilen i vinkelestimatet er tydelig avhengig av hvilken vinkel det er til kilden. For vinkelen 30 grader så viser det seg at feilen er 0, uavhengig hva avstanden er til kilden! Så kan vi igjen se hva som skjer når vi har et ti ganger så lite array, med $a = 1$cm, se figur 8.



Figur 7: Estimert vinkel til en lydkilde for forskjellige innfallsretninger (venstre), og tilsvarende feil i estimatet (høyre).



Figur 8: Feil i estimert vinkel til en lydkilde for forskjellige innfallsretninger for et lite array.

Feilen i vinkelestimat ser ut å være nesten presis 10 ganger mindre hvis arrayet er 10 ganger mindre.

Det ser ut som om det er mye bedre med et lite array! Er det noen ulempe med et mindre array? Ja, en gitt tidsdifferanse tilsvarer mye mindre fasedifferanse med et mindre array! Og, sensorene har ikke perfekt fasemålingsnøyaktighet. I tillegg, vi må jo plassere sensorene perfekt symmetrisk, og det blir mer vanskelig jo mindre arrayet er.

Til sist, som ble påpekt etter lign. (29), jo mindre arrayet er, jo mindre er n_{max}, og jo færre antall vinkler kan estimeres.

2 Forberedelsoppgaver til laben

Gjøres før laboppgaven.

2.1 Forberedelsesoppgave 1

Beskriv med Python-kode hvordan krysskorrelasjon kan brukes for å finne effektiv forsinkelse mellom to lydsignaler som er tatt opp med samplingsfrekvensen f_s.

Hint: du skal finne for hvilken forsinkelse som `numpy.abs(krysskorrelasjonsfunksjonen)` har maksimum.

2.2 Forberedelsesoppgave 2

(a) Hvis du har et triangelformet array som i figur 4, med sidelengde d, og samplingfrekvensen er f_s; hva er maksimalt antall sampler med forsinkelse som kan oppstå?

(b) Hvis du har et array med kun to mikrofoner, kan du detektere flere unike innfallsvinkler om maksimal forsinkelse mellom dem er 2 sampel eller 4 sampel? *Hint*: Et lite vinkelområde vil gi samme forsinkelse, n_{21}, grunnet avrunding til helt antall sampel!

3 Laboppgave

1. Lag et mikrofonarray med geometri som i figur 4. Velg d slik at alle mikrofonene får plass på koblingsbrettet.
2. Gjør et opptak av et lydsignal med de tre mikrofonene, og vis at de tre signalene ser ut som forventet, dvs amplituden skal variere rundt 0*, og avhengig av hvilket signal du brukte skal alle tre se ut som sinustoner, impulslyder, støysignaler, eller et slikt signal som du brukte.
3. Implementer en krysskorrelasjonsberegning for de tre mikrofonsignalene. Finn tidsforsinkelsen fra toppen av krysskorrelasjonen, uttrykkt i antall sampler og vis de tre krysskorrelasjonssignalene. Du må også vise et eksempel på et autokorrelasjonssignal† - sjekk at du får en topp for eksakt 0 sampels «lag».‡

   Er toppen for disse krysskorrelasjonssignalene innenfor et intervall som kan forventes, dvs innenfor $\pm n_{max}$, hvor n_{max} gis av avstand mellom mikrofonene, samplingfrekvensen og lydhastigheten?
4. Beregn innfallsvinkelen (i planet som de tre mikrofonene er plassert) fra de tre tidsforsinkelsene. Vis at du kan detektere innfallsvinkel fra -180 til $+180$ grader.
5. Gjør en systematisk variasjon av estimert innfallsvinkel for flere innfallsvinkler. Gjør flere, minimum 5, gjentatte målinger for samme innfallsvinkel for å få et mål på usikkerheten. Standardavviket og variansen er mål som godkjennes. Gjør dette for minimum 3 forskjellige vinkler.

Tips: Best resultat fra krysskorrelasjon oppnås ved å bruke et signal med mange frekvenskomponenter (feks støysignal, eller et impulslignende signal). Husk også å fjerne DC-komponenten fra rådataene (etter digitaliseringen).

Tips: Et båndpassfilter (i Python) før krysskorreleringen kan forbedre signal-til-støy-forholdet - men da må du vite i hvilket frekvensområde nyttesignalet du bruker ligger i.

3.1 Krav for godkjent lab

For å få godkjent laben må dere, i tillegg til å få godkjent på forberedelsesoppgavene, vise at dere har gjort laboppgaver 3 til 5.

*dvs du *må* fjerne DC-komponenten fra signalene!

†autokorrelasjonsfunksjon = krysskorrelasjon mellom to identiske signaler

‡Ved en implementering av krysskorrelasjon så kan det være lett å få feil «lag» med ± 1 sampel. Testen med autokorrelasjon er meget bra fordi vi vet at den *altid* skal ha en topp ved 0 sampels «lag».

4 Appendix - løsning av lign. (9) via matriseinvertering

Ligning (9) representerer 3 ligninger:

$$\mathbf{x}_{21} \cdot \mathbf{x} + c\tau_{21} = 0 \tag{31}$$

$$\mathbf{x}_{31} \cdot \mathbf{x} + c\tau_{31} = 0 \tag{32}$$

$$\mathbf{x}_{32} \cdot \mathbf{x} + c\tau_{32} = 0 \tag{33}$$

hvor den ukjente er $\mathbf{x} = [x, y]$ (som altså er en vektor av størrelse (1,2)), og vektorene $\mathbf{x}_{ji} = [x_{ji}, y_{ji}]$ (også vektorer av størrelse (1,2)) beskriver hvor sensorene er i forhold til hverandre. Disse tre ligningene kan sammenfattes i en matriseligning

$$\mathbf{X}_{ij}\mathbf{x}^T = -c\boldsymbol{\tau} \tag{34}$$

hvor

$$\boldsymbol{\tau} = [\tau_{21}, \tau_{31}, \tau_{32}]^T, \text{hvilket er en kolonnevektor av størrelse (3, 1)} \tag{35}$$

$$\mathbf{X}_{ji} = [\mathbf{x}_{21}^T, \mathbf{x}_{31}^T, \mathbf{x}_{32}^T]^T, \text{hvilket er en matrise av størrelse (3, 2)} \tag{36}$$

Denne måten å beskrive en matrise kan se tungvint ut; i Python ville vi skrive, etter `import numpy as np`,

```
>>> Xmatrise = np.array([[x21, y21], [x31, y31], [x32, y32]])
```

Som nevnt i hovedteksten så er (34) et overbestemt ligningssystem, hvilket kan løses med minste kvadraters metode. Dette kan gjøres elegant og effektivt ved å bruke pseudo-inversen til matrisen $\mathbf{X}_{ji}$ (Python-funksjonen `numpy.linalg.pinv` implementerer den såkalte Moore-Penrose pseudoinversen til en ikke-kvadratisk matrise):

$$\mathbf{x}^T = -c\mathbf{X}_{ji}^{\dagger}\boldsymbol{\tau} \tag{37}$$

hvor $\mathbf{X}^{\dagger}$ oppfyller

$$\mathbf{X}_{ji}\mathbf{X}_{ji}^{\dagger}\mathbf{X}_{ji} = \mathbf{X}_{ji} \tag{38}$$

4.1 Numerisk eksempel

Vi har et mikrofonarray som i figur 4, med $a = 0.1$m. En plan lydbølge har infallsvinkelen $\theta = 20$ grader. Da får vi følgende tidsforsinkelser

```
>>> t_matrix = 1e-3*np.array([0.386367, -0.087589, -0.473957])
```

og sensorene er plassert i

```
>>> Xmatrix = np.array([[-0.086603, -0.15], [0.086603, -0.15], [0.173205, 0]])
```

Med metoden i hovedteksten får vi ut θ som:

$$\theta = \arctan\left(\sqrt{3}\frac{\tau_1 + \tau_2}{\tau_1 - \tau_2 - 2\tau_3}\right) = \arctan\left(\sqrt{3}\frac{0.386367 - 0.087589}{0.386367 + 0.087589 + 2 \cdot 0.473957}\right) \tag{39}$$

$$= 0.349054\,\text{rad} \approx 20.00\,\text{grader} \tag{40}$$

Med matriseinvertering:

```
>>> xvector = -343.4 * np.linalg.pinv(Xmatrix)@t_matrix
```

så får vi løsningen

```
>>> xvector
array([0.93967499, 0.34200122])
```

og

```
>>> theta = np.arctan(xvector[1]/xvector[0])
>>> print(f"{theta:.4f}")
0.3491
```

Dvs, de to metodene gir samme resultat. Vi kan også notere at i matriseligning (34) så er det enkelt å innføre en tredje dimensjon! Vi trenger kun å la

$$\mathbf{x} = [x, y, z] \tag{41}$$

og så spesifisere differansen mellom z-koordinatene for de tre sensorene i

$$\mathbf{x}_{ji} = [x_{ji}, y_{ji}, z_{ji}] \tag{42}$$

Dermed blir matriseligningen fortsatt lik - men vi oppdager at selv om kilden ikke er i planet, så kommer vi alltid å få løsningen $z = 0$. Da virker det som om en slik tre-sensorarray ikke kan oppdage en «elevasjonsvinkel»? Det kan den, men vi må gå via en sjekk av lengden på vektoren $\mathbf{x}$. Lengden skal være 1, men med innfall-elevasjonsvinkel, så kommer lengden $||\mathbf{x}||$ ikke å være 1. Da kan vi få frem elevasjonsvinkelen, ϕ (`phi`) via

```
>>> phi = np.arcsin(np.linalg.norm(xvector))
```

Vi kan dog konstatere at et «flatt» mikrofonarray ikke kan detektere om bølgen kommer ovenfra eller underfra.