TMA4320 – våren 2019, Prosjekt 1
Blind source separation

Brynjulf Owren

1 Bakgrunn.

Tanken bak dette prosjektet er å gi en liten introduksjon til et veldig relevant område innenfor beregningsvitenskap i dag som har forgreninger til neurale nett og maskinlæring. Det handler om å prosessere og hente ut informasjon fra store datamengder. I dette prosjektet har vi lite tid til rådighet, derfor vil vi begrense oss til et litt mindre problem. Men idéene er noen av de samme som brukes innefor for eksempel områdene Data Science, Big Data osv.

Vi tar utgangspunkt i det såkalte cocktailselskap-problemet. Flere personer snakker samtidig i et åpent rom, kanskje det spilles musikk eller det kan være andre lydkilder. Så står det noen mikrofoner spredt rundt som registrerer og lagrer lydsignalene. Det kan være vanskelig å tyde hva som blir sagt fordi mikrofonene registrerer kun et surr av stemmer, dvs en miks fra flere lydkilder. Spørsmålet blir: Er det mulig å separere lydkildene fra hverandre? Merk at problemstillingen er ganske generell, i stedet for et cocktailparty kan det handle om signaler av ulik type sammen med støy som skal renskes opp. Det behøver ikke være lydsignaler, det kan også dreie seg om helt andre typer data som for eksempel bilder.

Matematisk sett kan vi tenke oss at (lyd)signal nr j er beskrevet av en matematisk funksjon $\mathbf{s}_j(t)$, $t \in [0, T]$ der t er tid og T er lengden av lydklippet. Når mikrofonene registrerer signalet så er det en lineær blanding av d kilder $\mathbf{s}_1(t), \ldots, \mathbf{s}_d(t)$, og man registrerer $\mathbf{x}_i(t)$ ved mikrofon nr i som

$$\mathbf{x}_i(t) = a_{i1}\,\mathbf{s}_1(t) + \cdots + a_{id}\,\mathbf{s}_d(t), \quad i = 1, \ldots, n.$$

Vi vil straks anta for dette prosjektet at $n = d$: antall mikrofoner er det samme som antall lydkilder. Det er ikke en nødvendig antagelse, man *kan* ha $n > d$, men her forenkler vi situasjonen litt.

Problemet er at miksematrisen $\mathbf{A} = ((a_{ij}))$ er ukjent. Så oppsummert har man altså d ukjente kilder $\mathbf{s}_1(t), \ldots, \mathbf{s}_d(t)$ og en ukjent miksematrise $\mathbf{A}$. De gitte data er d observasjoner $\mathbf{x}_1(t), \ldots, \mathbf{x}_d(t)$. I praksis er signalet digitalt, så i stedet for kontinuerlige variable $\mathbf{s}_j(t)$ og $\mathbf{x}_i(t)$ er dette samplede signaler gitt som en lang vektor, en tidsserie, $[\mathbf{s}_j(t_1), \mathbf{s}_j(t_2), \ldots, s_j(t_N)]$ og tilsvarende for $\mathbf{x}_i(t)$. Det vi vil gjøre når vi diskuterer og implementerer vår algoritme er å legge inn samplene

$\mathbf{s}_i(t_j)$ og data $\mathbf{x}_i(t_j)$ som elementene i matriser som kalles henholdsvis $\mathbf{s}$ og $\mathbf{x}$. Disse er begge av formen $d \times N$. Mens d er et lite tall, typisk $d = 3$ i våre eksempler, så er N et stort tall, antall sampler i et slikt signal kan være mange tusen. Så matrisene har noen få linjer og veldig mange kolonner.

2 Kort forklaring på "Uavhengig komponentanalyse"

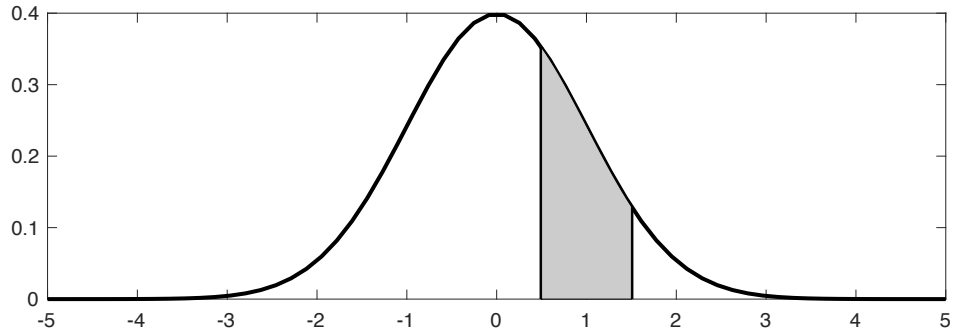
På engelsk kalles dette "Independent Component Analysis" og det betyr rett og slett det som allerede er antydet. Gitt noen signaler som er blitt mikset, utfør en demiksing av disse for å rekonstruere de umiksede signalene. Hva slags prinsipper som ligger til grunn for demiksingen forsøker vi å skissere her.

2.1 Antagelser

En viktig antagelse er at kildene kan ses på som statistisk uavhengige av hverandre. Det betyr at for hvert enkelt tidspunkt t kan $\mathbf{s}_j(t)$ oppfattes om uavhengige statistiske variable. Du kan tenke på dette slik: Selv om en ved tid t vet hva $\mathbf{s}_1(t)$ er så gir ikke dette noe mer informasjon om verdiene $\mathbf{s}_j(t)$ for $j > 1$. Hvis ikke dette er oppfylt, så kan man ikke forvente at analysen vil fungere. I praksis er ofte kildene uavhengige, men det kommer selvsagt an på hva slags signaler som skal demikses.

En annen antagelse er at signalene ikke er Gaussiske (et ekvivalent ord er normalfordelte). Mange av dere lærer nå statistikk, men vi skal etter beste evne forsøke å unngå å bruke teori fra statistikken. Men noe må vi nesten si for at problemstillingen skal gi mening. At en stokastisk variabel er gaussisk betyr at dens sannsynlighetsfordeling er en bestemt funksjon. Med to parametre, μ (forventningsverdi) og σ (standardavvik) har vi

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



Figuren viser et plott av en Gaussfordelingskurve med $\mu = 0$, $\sigma = 1$, også kalt normalfordeling. Arealet av det skraverte feltet viser sannsynligheten for at variabelen tar verdi mellom 0.5 og 1.5, den er ca 0.24.

Det er altså en forutsetning i metoden vi bruker at de opprinnelige signalene $s_j(t)$ ikke er gaussiske (normalfordelte). I statistikken fins et ganske avansert teorem som heter sentralgrenseteoremet. Der ser man på summen eller gjennomsnittet av uavhengige stokastiske variable, $Z = \frac{1}{n}(X_1 + \cdots + X_n)$. Når $n \to \infty$ vil fordelingen til Z nærme seg gaussisk. Dette gjelder også mer generelt for vektede kombinasjoner av $X_1, \ldots, X_n$. Poenget er at i vårt tilfelle kan vi konkludere med at de registrerte signalene $\mathbf{x}_i(t)$ har en fordeling som er "mer gaussisk" enn kildene $\mathbf{s}_i(t)$. Idéen i uavhengig komponentanalyse er å finne den kombinasjonen

$$\mathbf{y}(t) = w_1\mathbf{x}_1(t) + w_2\mathbf{x}_2(t) + \cdots + w_d\mathbf{x}_d(t) \tag{1}$$

som gjør $\mathbf{y}(t)$ mest mulig "ikke-gaussisk". Registreringene $\mathbf{x}_i(t)$ er gitt, så det er w_i vi må maksimere med hensyn på. Det resulterende signalet $\mathbf{y}(t)$ tas som approksimasjon til ett av kildesignalene $\mathbf{s}_j(t)$.

2.2 Hvordan kvantifisere "Ikke-gaussiskhet"?

For en kontinuerlig stokastisk variabel X med sannsynlighetsfordeling $f(x)$, $x \in \mathbb{R}$ så gjelder

$$\int_{-\infty}^{\infty} f(x)\,dx = 1.$$

Forventningsverdien til en slik variabel er

$$\mu = E[X] = \int_{-\infty}^{\infty} x\,f(x)\,dx,$$

mens variansen er definert som

$$\mathrm{Var}[X] = E[X^2] - E[X]^2 = \int_{-\infty}^{\infty} x^2\,f(x)\,dx - \mu^2$$

Merk at alle disse tre ligningene involverer uttrykk av typen $E[X^k]$, disse kalles generelt for momenter. Med $k = 4$ får man noe som er relatert til begrepet kurtose i statistikken, og dette er faktisk et mye brukt mål på ikke-gaussiskhet, og derfor nettopp det vi trenger. Det fins også andre måter å kvantifisere ikke-gaussiskhet på, blant annet noe som kalles negentropy. Men negentropy er vanskelig å beregne, og man ender ofte opp med å approksimere negentropy med varianter av slike momenter som nevnt ovenfor, og vi får gjerne noe som er beslektet med kurtose uansett. Uten å gå mer i detalj kan vi foreløpig konkludere med at uavhengig komponentanalyse handler om å maksimere et mål for ikke-gausiskhet, altså finn $w_1, \ldots, w_n$ som maksimerer f.eks. kurtosen til

$$\sum_{i=1}^{d} w_i\mathbf{x}_i(t)$$

3 Beskrivelse av algoritmen.

I det følgende antar vi at vi har d kilder $\mathbf{s}_1(t), \ldots, \mathbf{s}_d(t)$ og tilsvarende d registrerte signaler som er ulike mikser (lineærkombinasjoner) av disse. Som tidligere nevnt antar vi at hvert signal er representert i en radvektor med N elementer, f.eks.

$$\mathbf{s}_j = [\mathbf{s}_j(t_1), \mathbf{s}_j(t_2), \ldots, \mathbf{s}_j(t_N)]$$

De registrerte signalene $\mathbf{x}_i$ vil også være radvektorer med N elementer. Vi kan nå definere en $d \times N$-matrise $\mathbf{x}$ hvis rader er $\mathbf{x}_1, \ldots, x_\mathbf{d}$.

I det følgende beskriver vi trinnvis hva som må gjøres for å separere signalene. Kort oppsummert er det to preprosesseringsskritt etterfulgt av en maksimering av et mål for ikke-gaussiskhet.

3.1 Preprosessering

1. For signaler har den absolutte verdien gjerne ingen betydning. Det betyr for eksempel at signalene $\mathbf{x}(t)$ og $\mathbf{x}(t) + c$, der c er en konstant oppfattes som samme signal. Det blir for oss enklere formler dersom vi krever at forventningsverdien er 0, dvs at signalet oscillerer mellom positive og negative verdier omkring 0. En enkel måte å ordne dette på er å trekke fra gjennomsnittsverdien til hvert av de registrerte signalene. Om vi antar at t er en diskret variabel, dvs $t = t_i,\ i = 1, \ldots, N$ så modifiserer vi hvert av signalene slik

$$\mathbf{x}_j(t_i) \leftarrow \mathbf{x}_j(t_i) - \frac{1}{N} \sum_{m=1}^{N} \mathbf{x}_j(t_m), \quad i = 1, \ldots, N \tag{2}$$

   Dette gjør vi med hver av de d radene i $\mathbf{x}$ og får ut en såkalt sentrert versjon av dataene. For å unngå å lage mye unødvendig notasjon, så antar vi i det videre at matrisen $\mathbf{x}$ er slik at radene summerer seg til 0, dvs de er blitt sentrert.

2. "Whitening". Selv om kildene $s_j(t)$ antas å være uavhengige og dermed ukorrelerte, så vil ikke de miksede signalene $x_i(t)$ være det. Men det er mulig å transformere dem slik at de blir ukorrelerte med varians 1. Vi skal ikke diskutere i detalj hva "ukorrelert" betyr, det viktige er at vi gjør nok en transformasjon på de registrerte signalene som forenkler den videre prosesseringen. Fra $\mathbf{x}$ kan man utlede den såkalte kovariansmatrisen, som er en symmetrisk $d \times d$-matrise $\mathbf{C} = \mathbf{C}(\mathbf{x})$ med elementer

$$\mathbf{C}_{ij} = \frac{1}{N-1} \sum_{m=1}^{N} \mathbf{x}_i(t_m)\, \mathbf{x}_j(t_m)$$

   eller på mer kompakt matriseform: $\mathbf{C} = \frac{1}{N-1}\, \mathbf{x}\, \mathbf{x}^T$. Vi ønsker å finne den inverse kvadratroten av $\mathbf{C}$, dvs $\mathbf{C}^{-1/2}$. Dette er en symmetrisk matrise slik at $\mathbf{C}^{-1/2} \cdot \mathbf{C}^{-1/2} = \mathbf{C}^{-1}$. Deretter transformeres $\mathbf{x}$ ved å sette

$\tilde{\mathbf{x}} = \mathbf{C}^{-1/2} \cdot \mathbf{x}$. Det vil ha den behagelige konsekvensen at kovariansen etter transformasjon er $\tilde{\mathbf{C}} = \mathbf{I}$ dvs identitetsmatrisen. Det fins en standard måte å beregne slike inverse kvadratrøtter på ved hjelp av egenverdidekomposisjon. Man kan faktorisere C som

$$\mathbf{C} = \mathbf{E} \cdot \mathbf{D} \cdot \mathbf{E}^T$$

der $\mathbf{E}$ har egenvektorene til $\mathbf{C}$ som kolonner og $\mathbf{D}$ er diagonalmatrise med egenverdiene $\lambda_1, \ldots, \lambda_d$ til $\mathbf{C}$ på diagonalen. Disse er reelle og positive i dette tilfellet så man kan finne $\mathbf{D}^{-1/2}$ som diagonal matrisen med verdier $\lambda_1^{-1/2}, \ldots, \lambda_d^{-1/2}$ på diagonalen. Da finner man at

$$\mathbf{C}^{-1/2} = \mathbf{E} \cdot \mathbf{D}^{-1/2} \cdot \mathbf{E}^T$$

og de transformerte dataene blir

$$\tilde{\mathbf{x}} = \mathbf{C}^{-1/2} \cdot \mathbf{x} = \mathbf{E} \cdot \mathbf{D}^{-1/2} \cdot \mathbf{E}^T \cdot \mathbf{x}$$

Nå har dataene blitt transformert fra $\mathbf{x}$ til $\tilde{\mathbf{x}}$. Det er gjort på en slik måte at vi fremdeles har

$$\tilde{\mathbf{x}} = \tilde{\mathbf{A}} \cdot \mathbf{s}$$

Men $\tilde{\mathbf{A}}$ har blitt ortogonal, det vil si $\tilde{\mathbf{A}}^T \tilde{\mathbf{A}} = \mathbf{I}$. Det betyr at også $\tilde{\mathbf{A}}^{-1}$ er ortogonal.

3.2 Maksimering

Det siste og største steget handler altså om maksimering av et mål for ikke-gaussiskhet. Her gjør vi noen snarveier i forklaringen og refererer til [1]. Denne artikkelen formulerer en iterasjonsmetode for å maksimere et slikt mål. Målet for ikke-gaussiskhet er uttrykt ved en generell funksjon $G(u)$ og dens deriverte $G'(u)$. Det fins minst to veletablerte alternative valg av $G(z)$ nemlig

$$G_1(u) = 4u^3, \qquad G_2(u) = u\, e^{\frac{-u^2}{2}}.$$

1 og 2 i subskriptene svarer til henholdsvis *Kurtose* og *Negentropy*. Formålet med iterasjonen er å finne en $d \times d$-matrise $\mathbf{W}$ som approksimerer $\tilde{\mathbf{A}}^{-1}$ slik at vår approksimasjon til kildesignalene blir

$$\mathbf{y} = \mathbf{W} \cdot \tilde{\mathbf{x}}$$

Merk at sammenlignet med for eksempel (1) der vi kun så på én rekonstruksjon om gangen, arbeider vi nå med d sett av $\mathbf{w}$-vekter samtidig, og de blir da radene i matrisen $\mathbf{W}$.

Matrisen $\mathbf{W}$ bør (som $\tilde{\mathbf{A}}^{-1}$) være ortogonal, dvs $\mathbf{W}^T \cdot \mathbf{W} = \mathbf{I}$. Iterasjonen starter med en tilfeldig valgt $\mathbf{W}_0$ hvor lengden av hver rad normaliseres til 1 (slik tilfellet er i ortogonale matriser). Selve iterasjonen består av to skritt

1. Beregn en $\mathbf{W}^+$ fra $\mathbf{W}_{k-1}$ med formål om at $\mathbf{W}^+$ har større ikke-gaussiskhet enn $\mathbf{W}_{k-1}$, men uten å kreve at $\mathbf{W}^+$ er ortogonal. Dette er en slags variant av Newton's metode for optimering. Denne første delen kaller vi "Optimeringssteget".
2. Beregn $\mathbf{W}_k$ som den "nærmeste" ortogonale matrisen til $\mathbf{W}^+$. Dette er "Ortogonaliseringssteget" eller også kalt "Dekorrelasjonssteget".

I disse skrittene antas matrisen med de preprosesserte dataene $\tilde{\mathbf{x}}$ ($d \times N$-matrise) å være gitt som input, anta også at vi har funnet $\mathbf{W}_k$ for en $k \geq 0$.

1. Optimeringssteget
   - Beregn $\mathbf{s}_k = \mathbf{W}_k \cdot \tilde{\mathbf{x}} \in \mathbb{R}^{d \times N}$
   - Anvend funksjonen G elementvis på $\mathbf{s}_k$, vi kaller resultatet $\mathbf{G}$ (som er en $d \times N$-matrise)
   - Anvend funksjonen $G' = \frac{dG}{du}$ elementvis på $\mathbf{s}_k$, vi kaller resultatet $\mathbf{G}'$ (som er en $d \times N$-matrise)
   - Beregn $d \times d$-matrisen
   $$\mathbf{W}^+ = \frac{1}{N}\,\mathbf{G} \cdot \tilde{\mathbf{x}}^T - \operatorname{diag}\left(E[\mathbf{G}']\right) \cdot \mathbf{W}_k.$$
   Med $E[\mathbf{G}'] \in \mathbb{R}^d$ mener vi vektoren vi får ved å ta gjennomsnittsverdien av hver rad i $\mathbf{G}'$. Med $\operatorname{diag}(\mathbf{v})$ der $\mathbf{v} \in \mathbf{R}^d$ mener vi diagonalmatrisen ($d \times d$) med elementene til $\mathbf{v}$ langs hoveddiagonalen.
   - Normaliser radene i $\mathbf{W}^+$ til 1.
2. Ortogonaliseringssteget, projiser $\mathbf{W}^+$ på en ortogonal matrise $\mathbf{W}_{k+1}$
   - Sett $\mathbf{W}_{k+1} = \left(\mathbf{W}^+(\mathbf{W}^+)^T\right)^{-1/2}\mathbf{W}^+$. Den inverse kvadratroten av matrisen $\mathbf{W}^+(\mathbf{W}^+)^T$ beregnes på samme måte som i det andre prosesseringssteget (whitening).

Etter å gjennomført 1. og 2. så må man sjekke et konvergenskriterium. Om dette ikke er oppfylt går man tilbake og utfører 1. og 2. igjen, i motsatt fall termineres iterasjonen,

Konvergeskriterium. For å sjekke om iterasjonen kan stoppes er det rimelig å sjekke avviket mellom $\mathbf{W}_{k-1}$ og $\mathbf{W}_k$. En mulighet er å se på et mål for størrelsen til $\Delta = \mathbf{W}_k - \mathbf{W}_{k-1}$, for eksempel ved å bruke en matrisenorm og kriteriet $\|\Delta\| < \texttt{tol}$. Men vi skal bruke et litt annet mål som tar hensyn til at matrisene $\mathbf{W}_k$ er ortogonale.

$$\delta = \max_{1 \leq i \leq d} \left(1 - \left|\sum_{j=1}^{d} (\mathbf{W}_k)_{ij} (\mathbf{W}_{k-1})_{ij}\right|\right)$$

Kriteriet blir naturligvis $\delta < \texttt{tol}$.

4 Oppgavebeskrivelse

Her kommer en arbeidsbeskrivelse over hva du trenger å gjøre på prosjektet

1. Finn sammen med medstudenter du eventuelt vil være på gruppe med, og registrer gruppa på Blackboard
2. Les gjennom dette notatet, og les det du kan forstå av referanse [1]
3. Identifiser all utdelt kode og data og sett deg inn i hvordan du laster opp, plotter og spiller av lydklipp i Python
4. Skriv og test ut kode ved å fylle ut Jupyter Notebook som er vedlagt. Det anbefales at du tester ut hver enkelt funksjon separat, gjerne med noen enkle data som du lager selv slik at du ser at funksjonene gjør det de skal
5. Sett sammen hele algoritmen og prøv den ut med de oppgitte miksede lydklippene
6. Bonusoppgave: Prøv med egenproduserte lydklippmikser. Gruppa kan enten ta opp lyd selv, eller finne ferdige lydklipp (søk på nettet, bruk musikk eller annet) som mikses syntetisk på d ulike (tilfeldige) måter, for eksempel ved å bruke en randomgenerator for matrisen $\mathbf{A}$

Veiledende liste over Python-funksjoner som skal skrives. Det er forklart hva disse gjør i Jupyter Notebook dokumentet `Project_HO.ipynb` og input- outputargumenter er beskrevet.

```
center_rows
whiten_rows
normalize_rownorms
decorrelate_weights
update_W
measure_of_convergence
fast_ICA
```

Hva skal leveres inn

- Prosjektet leveres som en Jupyter Notebook fil (Én kopi for hele gruppa)
- All kode som skrives og brukes skal ligge i notebook'en og skal lages slik at den er enkel å kjøre for de som vurderer oppgaven
- Alle funksjoner skal komme med en kort beskrivelse av hva de gjør, med forklaring av input/output-variable
- Man skal kommentere hvor godt separasjonsmetoden fungerte, her skal man kun lytt og bruke skjønn i sin vurdering

References

[1] A. Hyvärinen and E. Oja, *Independent component analysis: algorithms and applications*, Neural networks 13 (2000) 411–413.