# Disposition 9 – Midlingsfiltre

Ved arbejde med signaler vil der altid opstå støj på en eller anden måde i det målte signal. Dette kan komme fra selve den fysiske ”størrelse” vi måler på, som har en form for støjkilde, eller det kan være i selve målingen støjen opstår (ved transduceren).

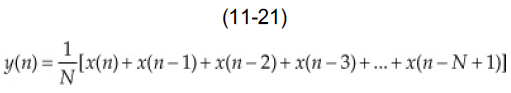
Dette er næsten altid uundgåeligt, men præcisionen af outputtet kan forbedres ved at midle over en række samplinger og tage gennemsnittet af dette. Dette vil give et mere præcist output. F.eks. hvis vi tager 100 samples og dividere summen af dem med 100 vil dette give os en enkelt 100-point gennemsnitsværdi af de 100 samples. Dette vil være et ret godt estimat af den sande værdi. Problemet ved denne metode er at, i realtids-krævende systemer, vil vi så skulle vente endnu 100 samples blot for at få nr. 2 output-værdi. I stedet anvender vi digitale midlingsfiltre til at løse dette problem.

### FIR moving averager

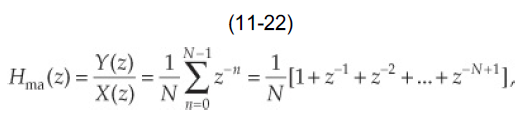
Moving average filtre kan implementeres bade som non-recursive eller recursive. Deres magnitude frekvens respons og phase response er identiske. Men implementeringsmæssigt ligner de ikke hinanden. Dette beskrives efter en kort introduktion af de to forskellige filtres differensligninger.

### Non-recursive

Herunder ses en N-point non-recursive moving averager’s output I tidsdomænet (differensligning):

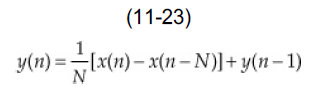


Og i Z-domænet:

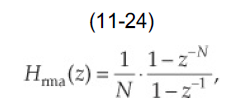


#### Recursive

Herunder ses differensligningen for N-point recursive moving averager:

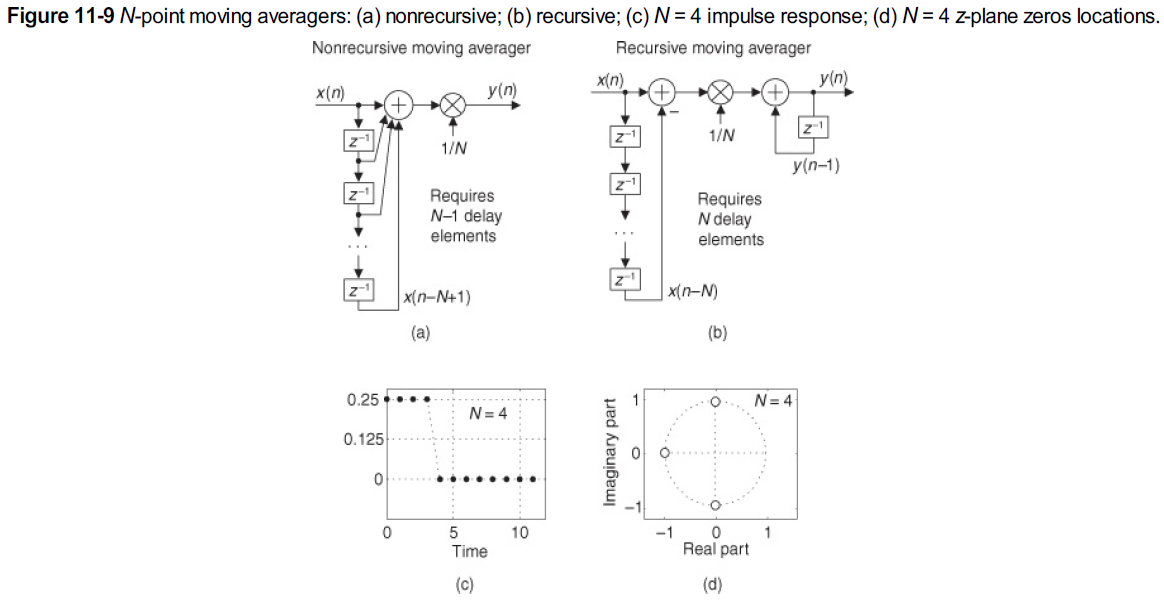


Og i Z-domænet:



#### Sammenligning af recursive og non-recursive moving-averager

Herunder illustreres begge slags implementeringer:



Her ses det at begge implementering kræver ”næsten” samme antal delays. Non-recursive har ved et N-point filter N-1 antal delay elementer, hvor en recursive implmentering kun har N antal delay elementer.

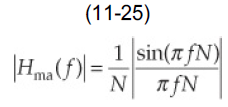
Dog er den store forskel, at recursive-versionen kun kræver 2 additioner per. Output sample UANSET antal delay-elementer. Det betyder, at recursive er ligeså ”langsom responsiv” alt efter antal delay elementer som non-recursive, men recursive filtre kræver langt færre computer beregninger og altså mindre krævende.

Men som nævnt tidligere, at de to forskellige implementeringer identiske tids-domæne impulse responses samt identiske lineær-fase respons.

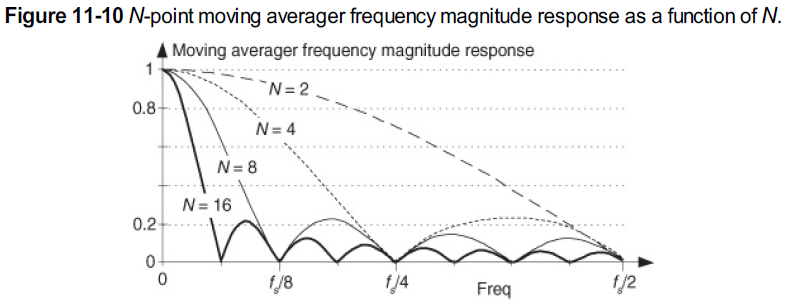
Man skal dog være opmærksom på ved implementeringen af et recursive filter, så kan outputtet vokse imod noget uendeligt stort, hvis altså systemet ikke er stabilt. Så dette er endnu et element som skal tages højde for ved implementeringen af denne type, hvor non-recursive ikke kan være ustabil af natur.

En god pointe ved implementeringen af et N-point moving-averager er at, N skal være et et ”integer power of two”. Fordi således vil det at gange med 1/N blot betyde en enkelt bit-shifting, hvilket begrænser antal multiplikationer.

Begge moving-averager’s frekvens magnitude respons’ beskrives ved:



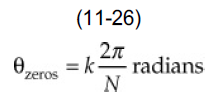
Hvor den normeret vinkelfrekvens f er i intervallet -0.5 til 0.5 (-fs/2 til fs/2). Det ses at funktionen ligner en sinc-funktion, hvilket kommer fordi den diskrete fourier transformation (DFT) af et averager’s rektangulære tids-domæne impulse respons giver en sin(x)/x lignende kurve. Nedenunder ses forskellige frekvens responses alt efter N:



Her ses det, at jo større N er, desto skarpere er filteret, hvilket dæmper mere og mere højfrekvent støj.

Yderligere ses det fra figur 11-10, at sidebåndene periodisk rammer 0 i magnitude. Bl.a. ”nulles” magnitude responset for N=4 ved fs/4 og fs/2. Dette kan ses ud fra pol-nul punkts diagrammet i figur 11-9 længere oppe.

Det viser sig at for en N-point moving-averager, så kan nulpunkternes placering i pol-nul punkts diagrammet beskrives ved:

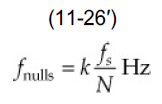


Hvor k = 1, 2, 3, …. N-1.

Her ses eksemplet for N=4 udregnet:

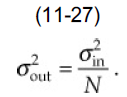
Dette ses overensstemmende med pol-nul punkts diagrammet i figur 11-9.

Den tilsvarende frekvens kan desuden udregnes på nogenlunde samme vis:



Hvor K = 1, 2, 3, … N-1.

Til sidst kan variansen af signalet efter filtreringen for begge slags implementeringer beskrives som:

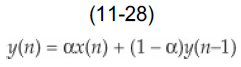


Den største negative side af disse to slags implementeringer (non-recursive og recursive moving-averager FIR filters) er at, de er langsomme overfor pludselig ændringer på indgangen. Altså de har et langt impulse response (transient forløb) hvis altså N er stor. Og N skal være stor hvis filtreringen skal være skarp som set i figur 11-10.

Dette kan overkommes hvis man i stedet implementere et eksponentiel averager.

### Eksponentiel Averager

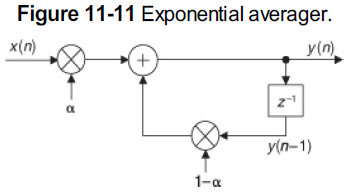
Denne form for midlingsfiltrering gør brug af feedback. Dens differensligning fås som:



Hvor det ses at vi har et gammelt output y(n-1) som feedes tilbage til indgangen.

α er vægtede konstant som beskriver det eksponentielle midlingsfiltre udseende og filtrering, og dets værdi er i intervallet 0 < α < 1.

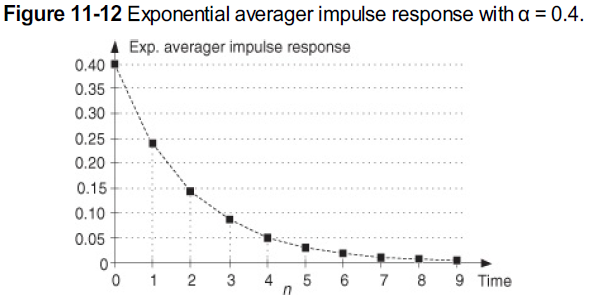
Dette filter implementeres således:



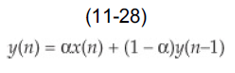
Eksponentielle midlingsfiltre har en række fordele. Det har færre antal computer-beregning end et non-recursive moving-averager. Desuden har det eksponentielle midlingsfiltre reduceret krav til hukommelse – altså der skal gemmes færre gamle værdier, faktisk skal der kun gemmes én, y(n-1).

Herunder ses et eksempel hvor et eksponentielt midlingsfilter med en α = 0.4 er brugt. Input-sekvensen beskrives om en lang række 0-input før t=0, og ved t=0 vil inputtet være 1 og derefter igen 0. Dette beskriver faktisk præcis kravene for et impulse-respons.

Dette giver derfor følgende impulse respons:

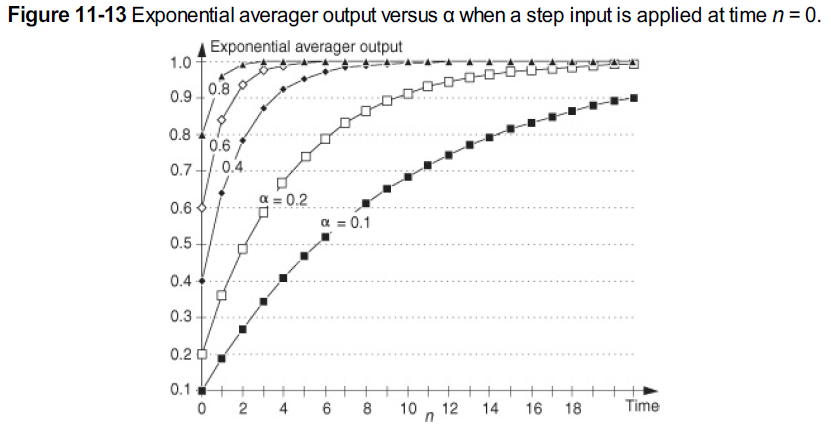


Herunder ses udregninger for de første 3 outputs, hvor x(0) = 1 og α = 0.4:



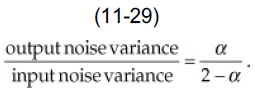
Heraf vås det eksponentielle navn, fordi det gamle output hele tiden bliver multipliceret med 0.6.

Det smarte ved eksponentielle midlingsfiltre er altså, at ved at ændre på blot α, så ændres hele filtreringen. Når α vokser imod 1, vil mindre og mindre af outputtet feedes tilbage ((1- α)\*y(n-1)), dog vil mere og mere af inputtet ”ryge” direkte igennem filtreringen (α\*x(n)), hvorfor systemet så pgså vil reagere hurtigere på hurtigt-varierende inputs – mindre støjreduktion. Sænkes α til gengæld dæmpes det direkte input mere (α\*x(n)), men gamle output samples feedes ”mere” tilbage nu ((1- α)\*y(n-1)). Her vil systemet reagere langsommere på varierende inputs og støj vil således dæmpes bedre. Dette ses meget bedre beskrevet i figuren under:

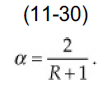


Stepresponset bliver hurtigere og hurtigere, jo større α er og omvendt. Dette vil altså ende ud i et tradeoff imellem et hurtigt system og lav støj-reduktion, eller et langsommere system, men bedre støj-reduktion.

Den eksponentielle averager’s støj-variation reduktion kan beskrives ud fra α som:

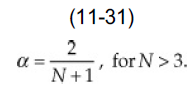


Dette er smart fordi vi da ud fra nogle krav vi selv stiller til en ønsket reduktion i støj variation (effekt/power), kan vi med en støj-variations faktor R, hvor R = (2- α)/α finde α som:



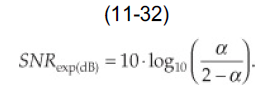
F.eks. hvis vi ønsker et signals støj-variation dæmpet med en faktor R=10, så skal α blot bestemmes som:

Skal det eksponentielle midlings filter sammenlignes (reducere tilsvarende i støj) med et N-point moving-averager filter, så kan antal unit-delays (N) for FIR filteret bestemmes ved:

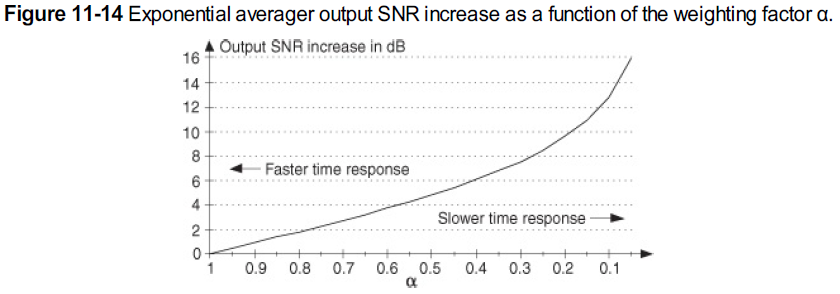


Denne sammenhæng beskriver altså hvad α og N skal være, for at give samme reduktion i støj.

SNR for det eksponentielle midlings filter beskrives som:

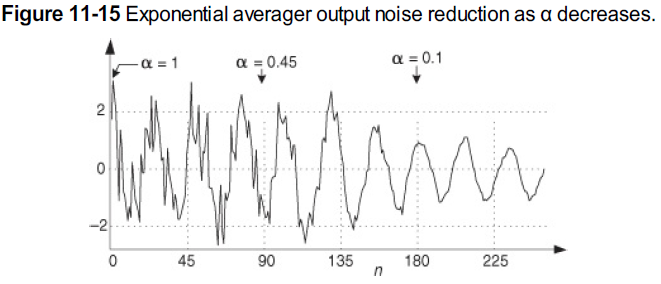


SNR kan afbildes som funktion af α som herunder:



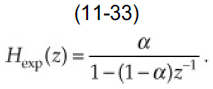
Her ses det tydeligt hvordan et større α giver et bedre SNR, men samtidig vises det også at response-tiden forværres.

Herunder ses et eksempel på lav-frekvent signal + et højfrekvent støjsignal, som filtreres af et eksponentielt midlings filter, hvor α mindskes med tiden (altså filtreringen af støj burde vokse i takt med α mindskes).

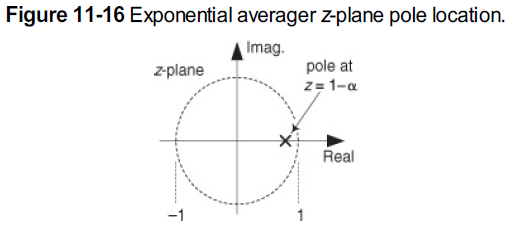


#### Frekvens respons for det eksponentielle midlings filter

Det ses at det eksponentielle midlings filter minder om et 1. ordens IIR filter (hvilket faktisk også er præcist hvad det er). Dets Z-domæne overføringsfunktion ses da som:

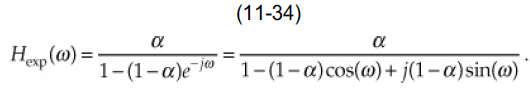


Da det kun er et første ordens filter, har den kun én pol ved z = 1- α.

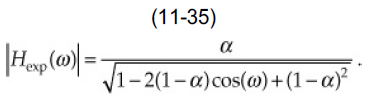


Når α reduceres, ses det ud fra z=1- α, at polen nærmere sig mere og mere enhedscirklen, hvilket betyder en skarpere lavpas filtrering.

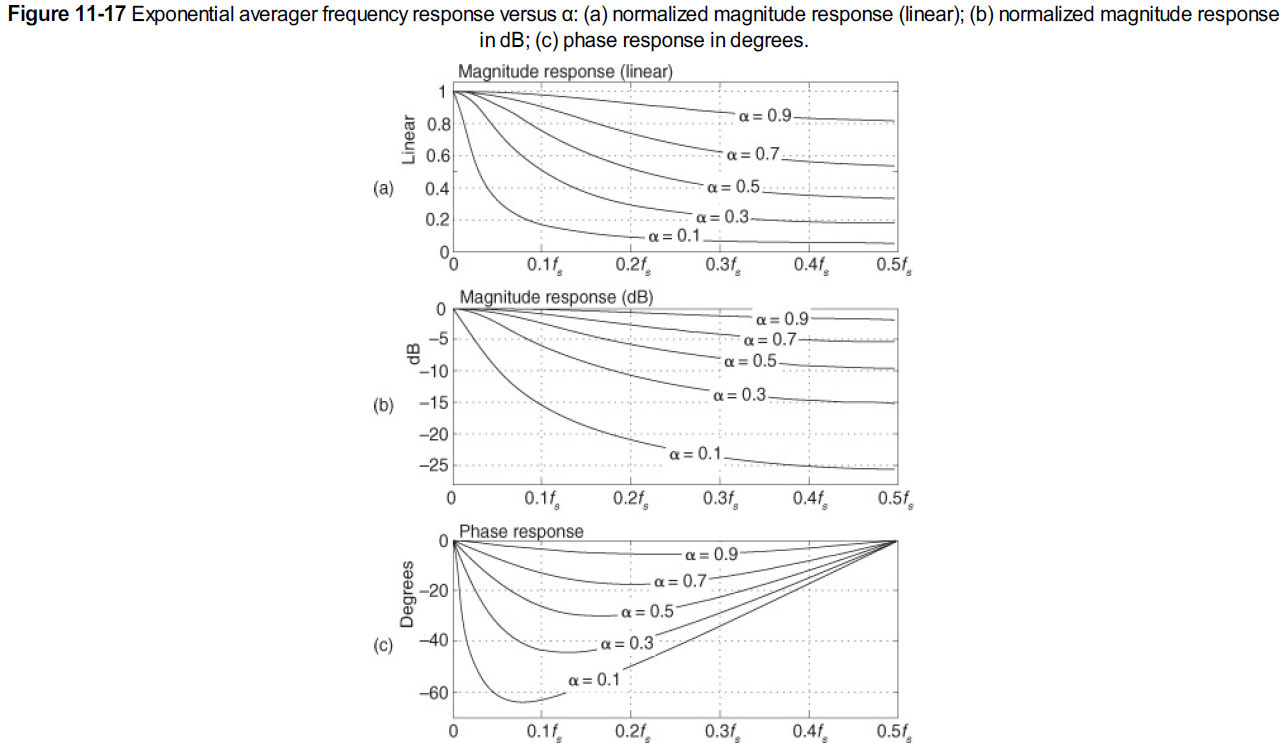
Indsættes på z’s plads, fås frekvens responset som:



Ønskes frekens magnitude responset, fås det som:



Evalueres frekvensresponset for , ses frekvensresponset som:



Det ses her tydeligt hvordan α påvirker lavpas filtreringen.

# 