FFI-rapport 2012/00012

Programvare for modellering og simulering av radar (versjon 1.0)

Terje Johnsen og Jabran Akhtar

FFI Forsvarets forskningsinstitutt

Programvare for modellering og simulering av radar (versjon 1.0)

Terje Johnsen og Jabran Akhtar

Forsvarets forskningsinstitutt (FFI)

1 november 2013

FFI-rapport 2012/00012

1158

P: ISBN 978-82-464-2304-3
E: ISBN 978-82-464-2305-0

Emneord

Radar

Modellering

Simulering

Godkjent av

Trygve Sparr	Prosjektleder
Johnny Bardal	Avdelingssjef

Sammendrag

I prosjektet Luftbåren overvåking er det blitt utviklet et program for å modellere ulike typer radarer og kjøre simuleringer i gitt terreng og omgivelser med valgte mål for å beregne en radars ytelse. Programmet er implementert som objektorientert Matlab-kode. Den foreløpige versjonen 1.0 har visse begrensninger, men tanken er å utvide på sikt til et enda mer kapabelt verktøy som kan brukes over et bredt spekter av konfigurasjoner med tanke på typer antenner, miljømessige forhold, signalbehandling osv. Programmets hovedfokus er på simuleringer utført ved hjelp av stokastiske numeriske verdier. Det har i denne versjonen vært lagt vekt på et sjøscenario hvor clutter fra sjø formes ved hjelp av ulike modeller med bestemte statistiske fordelinger. Atmosfærisk propageringsfaktor beregnes ved bruk av Advanced Propagation Model (APM) og angivelse av en rekke miljøparametere.

De forskjellige programmeringstekniske sidene ved programmet blir også beskrevet i rapporten, noe som vil kunne være til hjelp for fremtidig utvikling og fungere som en teknisk rettet referansebok.

Simuleringene styres av hvor radarbeamen rettes for hver puls og hvor målet befinner seg i forhold til denne. Det mottatte signalet bygges opp med bidrag fra støy, clutter og nivå av tilbakespredt energi fra mål ved hvert tidspunkt/puls. Programmet benytter statistiske variasjoner i bidragene fra puls til puls som følge av angitt Swerling-modell for mål, sjøclutterets korrelasjonsegenskaper, clutterets statistiske fordelingsfunksjoner og termisk støy. Beregninger av deteksjon gjør bruk av de valg som er angitt for MTI-filter, puls-integrasjon, CFAR og tersklingsnivåer.

I rapporten blir det presentert et sjøscenario med beskrivelse av angitte parametere og resultater i form av forskjellige plot. Til slutt er det vist en sammenligning med simulatorkjøring gjennomført mot standard radar i SADM (Ship Air Defence Model) som viser god overensstemmelse mellom de to programmene i dette scenarioet.

English summary

A program has been developed that models different radars and runs simulations in given terrain and environment alongside a target to quantify the radar performance. An object-oriented version of Matlab has been used as the programming environment. Certain restrictions apply in the first version 1.0. However, we plan to develop an even more capable toolkit that can represent a broader set of radars with respect to processing techniques, antennas, environment models etc. The programs emphasis is on numerical simulations with stochastic variables. In this first implementation attention has been aimed at an open sea scenario where sea clutter models, their statistical distribution functions and correlation characteristics have been incorporated. The computation of atmospheric propagation factors are undertaken by using the Advanced Propagation Model (APM).

The report additionally describes the program technical aspects of the code which would be aid in further development of the code and as a reference guide.

For each pulse the radar beam directs its energy in different directions in a rotating radar antenna model. The antenna beam pointing as a function of time and the position of the target relative to the beam drives the simulation forward. The received signal is built from contributions representing noise, clutter and backscattered energy in the direction of the antenna from the target. Statistical variations from pulse to pulse are simulated based on given Swerling target models, clutter correlation properties, clutter distribution functions and noise statistics. Detection calculations are subject to the input settings related to MTI behavior, pulse-integration, CFAR and detection thresholds.

An open sea scenario is presented where parameters and results are described. At the end a comparison with results obtained by use of default radar setup in SADM (Ship Air Defence Model) shows good agreement with the given scenario description.

Innhold

1 Innledning

Denne rapporten beskriver en Matlab-basert radarsimulator utviklet på FFI som har til formål å være et generelt verktøy for å estimere en radars ytelse. Tanken er å ha et fleksibelt program som kan anvendes på forskjellige typer radarer og scenariosettinger og som kan gi svar på grunnleggende spørsmål som ofte dukker opp i forbindelse med evalueringsarbeid og scenarioanalyser.

Det er tidligere blitt utviklet radarsimulatorer på FFI i forbindelse med NASAMS og SPY radarene med disse simulatorene er i stor grad skreddersydd for de bestemte radarene og gir begrenset med muligheter for videre utvikling. SADM (Ship Air Defence Model) er et kommersielt program som er blitt brukt ved FFI for evalueringsarbeid, men hvor man da er begrenset til de modellene og den systemforståelsen som ligger inne i programmet. Et eget utviklet verktøy gir muligheter for å endre alle små og store parametere og modeller i detalj og kan også brukes for å kvalitetssikre resultater fra andre kilder.

Simulatoren er bygd opp med en klassebasert struktur. Dette vil forhåpentligvis gjøre det enklere å bygge ut systemet og gjøre endinger. De forskjellige aspektene ved en radar blir simulert og evnen til deteksjon beregnet. Parametere som beskriver radar og omgivelser blir angitt i inputfiler. Første del av rapporten beskriver simulatoren, betydning og oppsett av de forskjellige parametrene. Den andre delen av rapporten tar for seg et konkret radarscenario og en gjennomgang av prosessen i forskjellige ledd med eksempler og påfølgende figurer av resultater.

Scenarioene som angir posisjon, bane, vinkler og lignende til radaren og mål kan settes sammen i for eksempel STK. STK er et sammensatt verktøy for systemmodellering og analyse for blant annet bruk i forsvarssammenhenger og blir brukt av diverse miljøer på FFI.

Simulatoren kan i prinsippet kjøres i to forskjellige moduser. Den første modusen følger i mest mulig grad en statistisk analytisk tilnærming. Hovedresultatet, sannsynlighet for deteksjon som funksjon av avstand, regnes ut ved hjelp av eksakte deteksjonsformler. Ulempen med denne metoden er at formlene kun er gyldige for veldig generelle tilfeller og utelukker kombinasjoner hvor radaren bruker sofistikerte signalbehandlingsmetoder for eksempel til å undertrykke clutter.

Den andre radarmodusen er basert på numeriske simuleringer med stokastiske variable hvor det er mer naturlig å direkte implementere signalbehandlingsalgoritmer. Dette gjenspeiler også hvordan et praktisk scenario med en radar vil operere. Forskjellige typer filtre kan for eksempel tas i bruk i tillegg til at radaren kan ha varierende PRI (Puls Repetition Interval – tidsforsinkelsen mellom to emitterte pulser) og implementere flere typer deteksjonsmetoder. Siden det her er snakk om stokastiske simuleringer vil resultatene variere noe fra simulering til simulering. På grunn av den økte fleksibiliteten fokuserer denne rapporten hovedsakelig på den numeriske simuleringsdelen som også er hovedmodusen for simulatoren.

2 Input data og programmets klassestruktur

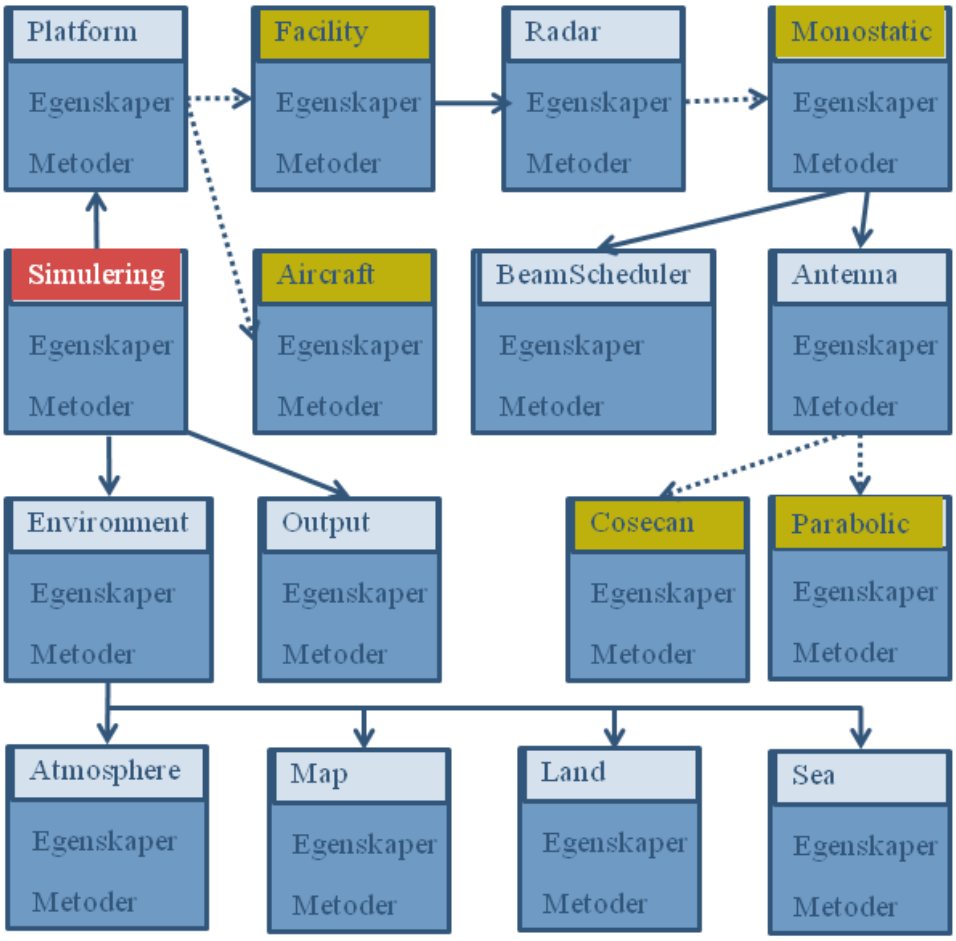
De aller fleste variable relatert til simuleringen, som for eksempel radarparametere, blir definert i egne eksterne filer som leses inn ved oppstart. Dette gjør at en bruker ikke behøver å gå inn i kildekoden og foreta endringer. Filer for kontroll av simuleringene legges under et område som kalles for Prosjekt. Det er hensiktsmessig å legge filene under kataloger som relateres til gitte radarer for enkelt å kunne hente inn/redigere parametre. Under valgt prosjekt defineres et underprosjekt som kan være moder eller instanser av valgte radar med variasjoner i ulike oppsett-filer, for eksempel en annen antenne, forskjellige miljøomgivelser, en annen type terreng osv. I Figur 2.1 er de filene som definerer et underprosjekt visualisert. Øverst ser vi underprosjekt-filen (name.m) som er en Matlab scriptfil. Denne definerer navnene på de filene som inngår i underprosjektet. Det er i dag 7 filer som leses inn av simulatoren ved oppstart. De to øverste .mat filene er Matlab variabelfiler inneholdende scenarioet med radarposisjoner og initial klassestruktur med startparametre. I dette eksempelet angis en fil som angir et missils bane som funksjon av tid. De fire .txt-filene beskriver parametre for henholdsvis Simulering, Target, Antenne og Radar. Eksempel på noe av innholdet er vist i respektive bokser i figuren. I tillegg til disse .txt-filene er det en Environment-fil som er lagt inn som en Matlab script-fil (.m). Denne lastes inn og eksekveres linje for linje slik at variablene settes. Disse representerer parametre som inngår i kallet til APM (Advanced Propagation Model) for å beregne propageringsfaktor. Detaljene for de ulike parametrene beskrives senere i rapporten.

Figur 2.1 Illustrasjon av filinnhold i prosjektrelaterte filer som definerer et prosjekt.

Klassene som er implementert er i simulatoren strukturert som vist i Figur 2.2. Hver boks representerer en klasse med navn gitt i toppen og med definerte egenskaper og metoder. Klassene har i denne versjonen av programmet en instans hver. Tittelfeltet er gitt ulike farger for å skille

klasser (lys blå) fra underklasser (gul) som arver egenskaper fra superklassen. Det er i tillegg gitt en rødfarge til klassen Simulering fordi det er denne klassen som det opprettes en instans av først og som de andre refereres til fra lokale egenskaper. Strukturen i programvaren kan således forstås ved å starte i denne. De heltrukne linjene indikerer hvordan klasser er referert i andre klassers egenskaper mens de stiplede linjene indikerer hvilke underklasser av forgående superklasse som er opprettet.

Simulering kan ha flere instanser av typen Platform tilordnet i en liste. I denne versjonen er to instanser av underklasser av type Facility og Aircraft opprettet. Instanser av klasse Platform kan ha en radarliste og her er det tenkt en installasjon som er gitt en radar av underklasse Monostatic, mens Aircraft-objektet ikke er tilordnet noen radar. Radaren har tilordnet seg et objekt fra en antenneklasse av typen Cosecan og et objekt av type BeamScheduler-klasse. Simulering er også tilordnet en Environment-klasse som igjen har referanser til fire klasseobjekter av typen Atmosphere, Map, Land(Clutter) og Sea(Clutter). I tillegg har Simulering et Output-klasseobjekt tilordnet for plotting og utskrift av simuleringsresultater.



Figur 2.2 Klasser og underklasser og deres struktur i radarsimulatoren

3 Simulatoroppbygning

3.1 Simulering av returnert signal

Oppbygning av simulert reflektert signal baserer seg på radarlikningen. Radarlikningen angir signal til støy forhold og blir i simulatoren beregnet ved fastsatte avstander i radarbeamen med jevne mellomrom. Hvor tett denne samplingen skal være og radarens maksimale dekningsområde er variable som blir definert av brukeren. Beregningene skjer etter simulert transmisjon av hver eneste puls.

Radarlikningen som denne simulatoren tar utgangspunkt i kan formuleres som et signal til støyforhold (SNR), gitt ved

$$SNR = \frac{P_t G_t G_r G_c \sigma \lambda^2 F^4}{(4\pi)^3 R^4 k T L B_n}. \tag{3.1}$$

Denne likningen brukes for å kunne anslå hvor mye energi som blir reflektert fra et mål og hvor mye støy som kommer inn i mottageren.

Betydning av de forskjellige komponentene i radarlikningen:

`Pt:` Sendereffekt (Transmit power)
`Gt:` Antennegain sender
`Gr:` Antennegain mottak
`Gc:` Pulskompresjonsgain
σ: Målets radartverrsnitt (RCS)
λ: Bølgelengde (1/frekvens)
`F:` Propagasjonsfaktor
`R:` Avstand til mål
`k:` Boltzmanns konstant
`T:` Antennetemperatur i Kelvin
`L:` System loss
`Bn:` Båndbredde

Elementer som antennegain, kompresjonsgain, frekvens, båndbredde etc. antas å være konstante. Parametere som derimot endres mellom transmisjon av hver eneste puls og simuleres er følgende:
`r`: avstand. Avstanden mellom radar og mål er den naturlige variabelen som endres fortløpende fra puls til puls. Avstandsverdiene trekkes ut fra den gitte målprofilen.
σ: Refleksjonsfaktoren (RCS) fra et mål kan enten være en fast verdi eller den kan endre seg som funksjon av inn og utfallsvinkel i forhold til radaren eventuelt også frekvens og båndbredde. Denne verdien blir simulert som en angitt Swerling-fordeling som er helt eller delvis statisk over en viss koherensperiode. I denne koherensperioden er den da enten konstant eller om ønskelig modelleres den til å forandre seg tilfeldig, normalfordelt, fra puls til puls.

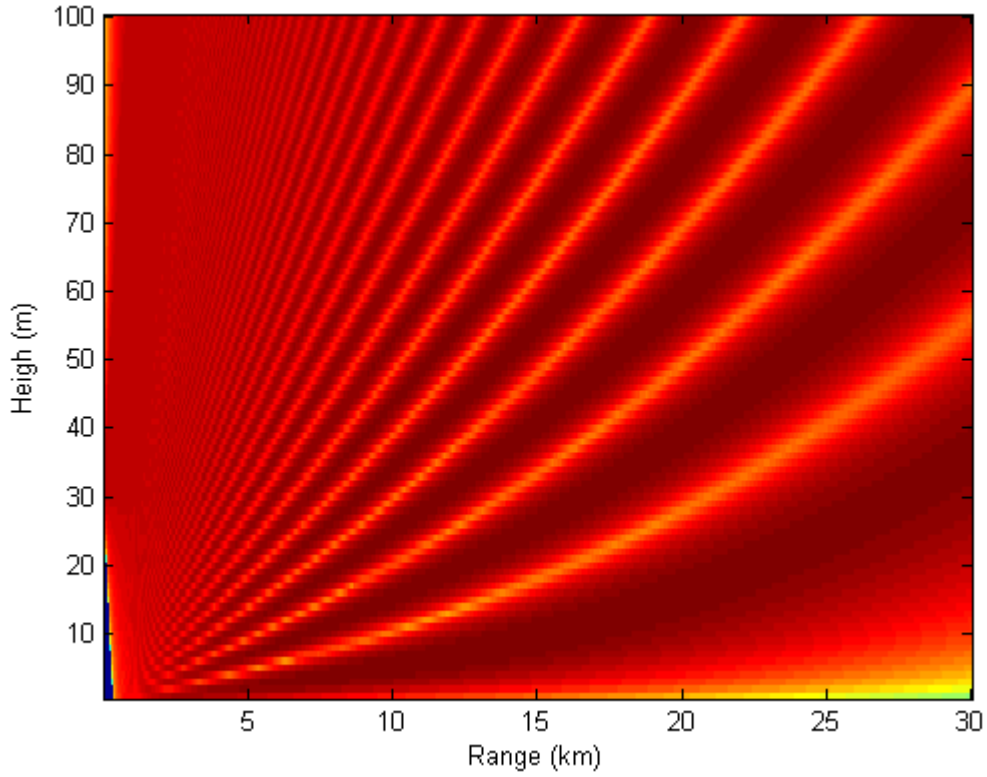
Målets hastighet, angitt av profilen, anvendes i tillegg for å simulere Dopplerskift som legges inn som en del av RCS komponenten.

3.1.1 Propagasjonsfaktor: F^2

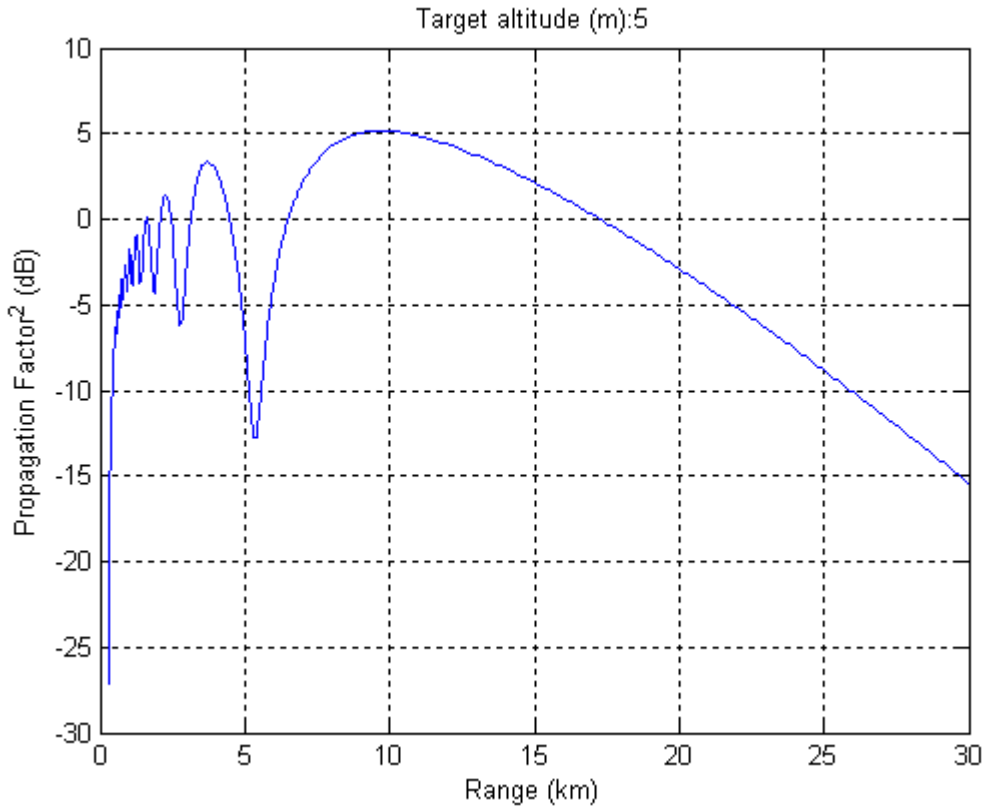
Beregning av propagasjonsfaktor gjøres ved hjelp av Advanced Propagation Model (APM) versjon 2.1.04 med opphav fra US Navy ved SPAWAR Systems Center (SSC) i San Diego. Denne er skrevet i Fortran og er kompilert til en MEX-fil som kan kalles fra Matlab. Programmet forventer en lang rekke inputparametre som beskriver radar og omgivelse. Beregningene gjøres i et vertikalt plan fra radarens antenne og i en gitt retning. Terrenget og det atmosfæriske mediet den elektromagnetiske strålingen propagerer gjennom kan spesifiseres som funksjon av avstand og høyde. Det kan derfor beskrives en lang rekke ulike atmosfæriske forhold som har innvirkning på propageringen og det tapet en radar vil erfare til ulike punkter i rommet. APM beregner enveis-propagasjonsfaktor F^2 som funksjon av avstand og høyde for elektromagnetiske bølger gitt en beskrivelse av atmosfæren, terrenget og radaren gitt som input parametre. I likningen for SNR i (3.1) inngår toveis-propagasjonsfaktor F^4, hvilket tilsier at propageringsfaktoren må kvadreres.

Inputparametre hentes fra angitt inputfil under EnvironmentFiles samt radarparametre gitt i prosjektfilen som beskriver radar under test. Det underliggende terrenget som APM trenger for å beregne F^2 vil for en bevegelig radar eller et bevegelig mål endres som funksjon av tid. Da APM beregner langs et vertikalt snitt hvor radar og mål er inneholdt vil terreng/sjø innhold som funksjon av avstand angis før hver beregning. DTED-filer er lest inn i simuleringsverktøyet og terrengprofiler må estimeres fra disse. For å redusere tiden som brukes på APM vil det før hver ny beregning av APM sjekkes om foregående beregning kan brukes. Kriteriet for dette er lagt inn som en øvre vinkelendring fra stasjonær radar mot mål, som medfører gjenbruk av beregnet propageringsfaktor. Dersom målet beveger seg mot radar langs en rettlinjet bane vil de samme outputverdiene fra APM brukes igjen og igjen, noe som sparer tid i simuleringen.

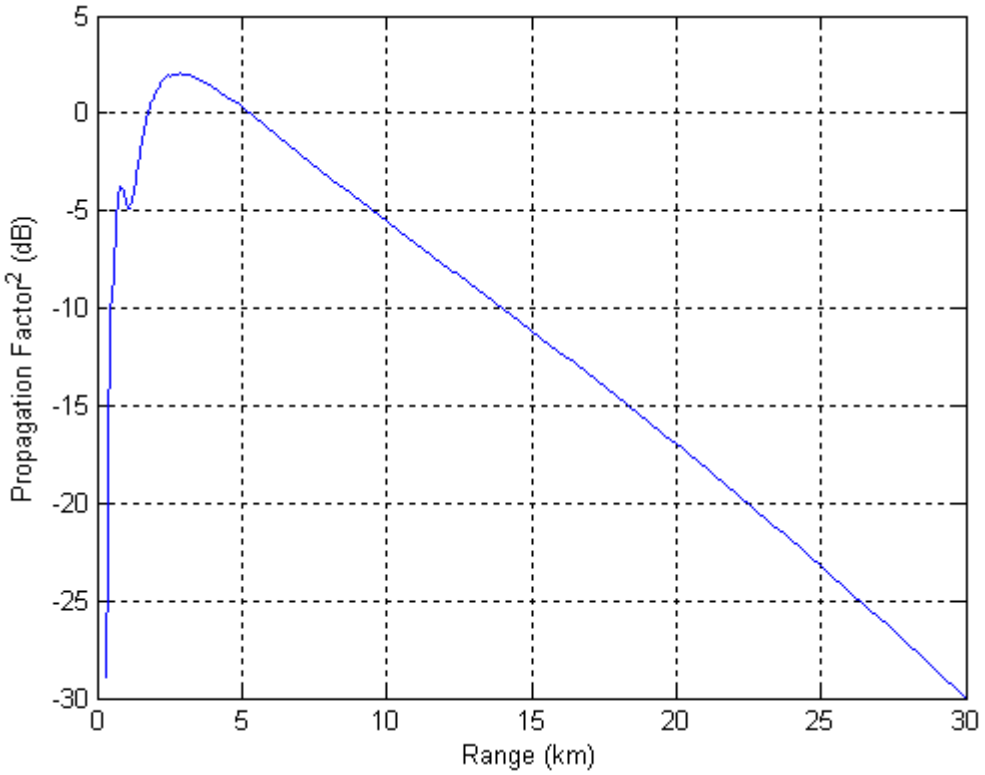
I Figur 3.1 er propageringsfaktoren som APM leverer vist for et gitt sett av inputparametre. I de to påfølgende figurene, Figur 3.2 og Figur 3.3 er verdiene vist for et mål i henholdsvis 5m og 1m høyde som funksjon av avstand. Det siste plottet angir verdier som inngår i beregningene av tilbakespredt sjøclutter.



Figur 3.1 Eksempel på propageringsfaktor-output fra APM som funksjon av range og høyde vist for tilfellet som vil bli behandlet i avsnitt 4. Fargeskalaen angir verdien til F^2.



Figur 3.2 Snitt gjennom F2-matrisen fra Figur 3.1 for mål i konstant høyde på 5m.



Figur 3.3 Snitt gjennom F2-matrisen fra Figur 3.1 i konstant høyde på 1m til bruk ifbm beregning av cluttersignalnivåer.

3.2 Clutter

Avhengig av om den modellerte radaren belyser land eller sjø med sin antennelobe vil bidrag fra henholdsvis sjøclutter og landclutter komme i tillegg til bidrag fra mål og støy i mottaker. Det er lagt inn to sjøcluttermodeller og forberedt for bruk av en landcluttermodell i simulatoren.

Sjøclutter er komplisert å modellere og avhenger av mange forhold og er et forskningsfelt hvor det er gjort vesentlig arbeid [1-3]. Sjøen er sjelden ”ren” i den forstand at den består kun av bølger som har blitt generert av vind fra en retning, kanskje med konstant styrke over en gitt tid, men er oftest en superposisjon av gamle og nye bølger med ulike forplantningsretninger. I de ulike avstand-asimut oppløsningscellene til radaren på sjøoverflaten vil nivåene ha en tilfeldig fluktuasjon og det kan se ut som en støyliknende variasjon i amplitude. Til forskjell fra termisk støy inneholder sjøclutter tidsmessige og romlige korrelasjoner. Dette medfører at pulsintegrasjon for clutterceller ikke dekorrelerer på lik linje som bidrag fra termisk støy. De detaljerte korrelasjonsegenskapene til clutteret vil være viktige for integrasjons-gainet.

Det tilbakespredte signalet fra sjøoverflaten vil ha egenskaper som er forskjellig for store og små clutterceller (lavoppløselig og høyoppløselig). Dersom cluttercellen dekker et større areal som inneholder et større antall bølger vil den koherente summen av et større antall spredere bidra til en amplitudestatistikk som kan beskrives som Rayleigh-fordelt. Bidragene for amplitude i I og Q-kanalene er i så fall Gauss-fordelte.

Variasjonene rundt et midlere clutternivå skyldes i hovedsak to faktorer[2]:

1) Variasjonen i betraktningsvinkel, overflatens form og kapillærbølger på de lange bølgene og dønningene
2) Mange spredere som beveger seg uavhengig av hverandre innenfor radarens oppløsningscelle skaper interferens i det mottatte signalet som vi vil oppfatte som såkalt Speckle.

Ut i fra dette kan vi forstå at for en lavoppløselig radar vil det for faktor 2) være mange lange bølger innenfor oppløsningscellen og denne vil nærme seg et bidrag som har en Rayleigh-fordeling av amplitude. For det andre ytterpunktet hvor oppløsningen blir liten og det er kun få eller deler av en lang bølge innenfor oppløsningscellen vil vi ha en vesentlig forskjellig statistikk hvor radaren oppløser strukturen til bølgene på en finere skala. Målinger som er rapportert i litteraturen viser også at for en høyere oppløsning blir amplitudefordelingen mer langhalet eller ”spiky” som clutter ofte omtales som. Vi ser også fra dette at en radarlobe vil ha oppløsningsceller som øker i areal som funksjon av avstand og derfor også en statistikk som er avhengig av avstand. I tillegg til dette vil ulike bidrag fra oppløsningscellen ha ulik Doppler og spredningen og forsyvningen i Doppler vil også være variable som inngår i en karakterisering av sjøclutter.

Speckle-komponenten og bidraget fra tilbakespredning fra de lange bølgene har ulike dekorrelasjonsegenskaper. Speckle har en rask dekorrelasjonstid, typisk noen få millisekunder mens variasjonen i den midlere tilbakespredningen har dekorrelasjonstider på flere sekunder. Fra puls til puls innenfor et pulstog vil derfor ikke den midlere amplitudeverdien dekorrelere men fra scan til scan på typisk noen sekunder kan man forvente en større grad av dekorrelasjon. Romlig dekorrelasjon av sjøclutter fra en cluttercelle til nærliggende clutterceller vil være avhengig av oppløsningscelle og romlig korrelasjon av sjøoverflaten.

Beskrivelsen av sjøclutter for lave betraktningsvinkler og høy oppløsning i avstand med bidrag fra de nevnte to komponentene kan beskrives godt med en sammensatt form av en K-fordeling[4] som angitt i (3.6). De eksperimentelle dataene brukt der viser at det er en god modell for radarfrekvenser i intervallet 1-16 GHz og avstandsoppløsning fra 0.375m til 15m. I hvilken grad modellen bryter sammen for andre verdier er uklart.

På nåværende tidspunkt er ikke alle bidrag til clutterets oppførsel implementert, men kun en første tilnærming med noen effekter. I denne modelleringsaktiviteten er midlere clutternivåer basert på de empiriske modellene GIT og NRL som er beskrevet videre i 3.2.1 og 3.2.2.

3.2.1 Sjøclutter med GIT-modellen

Simulering av sjøclutter gjøres ved hjelp av den såkalte GIT-modellen utviklet ved Georgia Institute of Technology [5]. Modellen som er basert på empiriske data og beregner sjøens midlere reflektivitet per kvadratmeter sjøoverflate som angis som σ^o. Reflektiviteten er en funksjon av flere forhold som frekvens, polarisasjon, betraktningsvinkel, vindhastighet og vindretning.

Empiriske modeller som baserer seg på målte data har ikke separert ut effekten av atmosfærisk påvirkning i dataene. Således er det en effektiv σ^o som er modellert som mer korrekt beskriver den målte verdien av $\sigma^o F^4$ hvor atmosfærisk påvirkning er inneholdt i dataene. Hvis man gjør antagelsen at de empiriske dataene som ligger til grunn representerer en såkalt standard atmosfære, så må beregningene ta høyde for dette når andre propageringsforhold er gjeldende. Dette er gjort i sjøclutterberegningene internt i APM og på likt vis i verktøyet The Ship Air Defence Model (SADM) fra BAE Systems som brukes på FFI. Hvis vi antar standard atmosfære for GIT-modellens verdier, kan vi skrive

$$\sigma^o_{GIT} = \sigma^o F^4_{Std}, \tag{3.2}$$

hvor Std angir bruk av refraktivitetsprofil i overensstemmelse med standard atmosfære. En ser da at i (3.2) må man dele på F^4_{Std} før det ganges med F^4 for gjeldende refraktivitetsprofil. Sjøens radartverrsnitt er gitt ved σ, og dette er det totale radartverrsnittet radaren erfarer fra en oppløsningscelle eller cluttercelle på overflaten. Ved små betraktningsvinkler kan dette arealet skrives

$$A_c = R\theta_{Az\_bw}(c\tau / 2), \tag{3.3}$$

hvor θ_{Az_bw} er radarstrålens bredde i asimut og τ er komprimert pulsbredde. Sjøclutteret kan dermed skrives

$$\sigma = \frac{\sigma^o_{GIT}}{F^4_{Std}} A_c, \tag{3.4}$$

som igjen er en funksjon av avstand *R*.

Midlere refleksjonsverdi sammen med generelle radarparametre og en angitt formfaktor *v* gir opphav til en skaleringsfaktor c_s:

$$c_s = \sqrt{\frac{4\nu}{P_t G_t^2 \frac{\lambda^2 F^4}{(4\pi)^3 R^3}(\sigma_0 \theta_{Az\_bw} \frac{c\tau}{2})}} \tag{3.5}$$

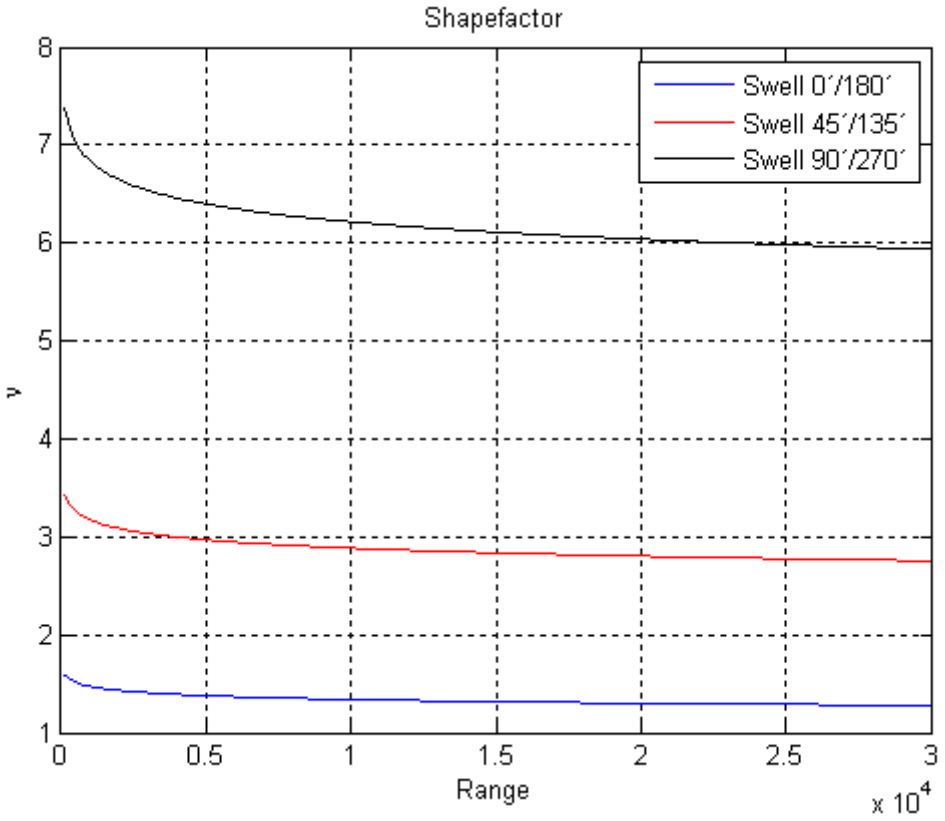
Skaleringsfaktor og formfaktor bygger opp en K-fordeling for clutter:

$$f_K(a) = \int_0^\infty f(a \mid y) f(y) dy \approx \frac{2c_s}{\Gamma(\nu)} \left(\frac{c_s a}{2} \right)^\nu K_{\nu-1}(c_s a), \tag{3.6}$$

hvor K_v er en v-te ordens Bessel funksjon og a angir amplitudeverdien av reflektert cluttersignal. Denne K-fordelingen brukes i simulatoren for å trekke tilfeldige verdier for clutter for hver avstandscelle. Det er ikke lagt inn muligheter for romlige korrelasjoner i modellen. Formfaktoren v angir tilstanden på bølger. Store verdier gir mer normalfordelt sjø, mens lave verdier angir en "spiky" struktur. Formfaktoren kan enten angis å være et fast tall men simulatoren gir også mulighet for at denne faktoren blir bestemt ved hjelp av diverse modeller for å få mer realisme i simuleringene. Ward-Tough-Watts modellen[2] er blant annet implementert i simulatoren. Formfaktoren blir da bestemt av flere forhold:

$$\log_{10}(v) = \frac{2}{3}\log_{10}(\phi) + \frac{5}{8}\log_{10}(A_c) - k_{pol} - \frac{\cos(2\theta)}{3} \qquad (3.7)$$

hvor ϕ angir innfallsvinkel i grader, A_c er dekningsområde i m^2, k_{pol} er polarisasjon (1.39 for VV og 2.09 for HH) mens θ gir vinkel i forhold til retningene på dønningene (svell). Et eksempel på hvordan formfaktoren da endrer seg som funksjon av avstand er gitt i Figur 3.4.



Figur 3.4 Formfaktor som funksjon av avstand gitt av Ward-Tough-Watts modell for ulike bølgeretninger ift radarens pekeretning.

Det er en egen parameter som bestemmer koherensperioden for clutter gitt i antall pulser. Denne kan være lik eller ulik koherensperioden for målets radartverrsnitt. Endring i clutter, fra puls til puls, modelleres, om ønskelig, som en enkel tilfeldig vandring med normalfordeling eller K-

fordeling. Etter at koherensperioden er over trekkes det ut nye tilfeldige clutterverdier for samtlige celler.

3.2.2 Sjøclutter med NRL-modell:

I tillegg til GIT-modellen er også NRL-modellen som ble utviklet ved Naval Research Laboratory, implementert [6]. De har generert en empirisk sjøcluttermodel for lave innfallsvinkler ved en tilpasning til et stort sett av eksperimentelle data presentert i Nathansons bok [7]. Spesielt ved lave sjøtilstander er det til dels store avvik mellom GIT-modellen og de eksperimentelle data. Modellen er også gyldig for sjøtilstand 0, noe som ikke er tilfelle ved GIT-modellen. Modellen er definert som funksjon av radarfrekvens, polarisasjon, sjøtilstand og innfallsvinkel.

3.2.3 Landcluttermodell

Det er foreløpig kun en enkelt landcluttermodell som er under implementering. Denne er ikke operativ ennå og vil komme inn etter hvert.

3.2.4 Bruk av DTED kart

Modellen leser inn DTED-avledede kart som på forhånd er lagret i kartutsnitt med størrelse 1x1 grad. Årsaken til at ikke kartutsnittene hentes direkte fra DTED-filene med MatLab sine rutiner er at dette krever mapping-toolbox som er begrenset av antallet lisenser på FFI. Utsnittet som skal leses inn beregnes fra radarenes plassering og målenes bevegelse gjennom scenarioet. Det innleste utsnittet er dermed det minste firkantede arealet som innholder alle posisjoner som scenarioet bruker.

3.3 Radarprosessering og lobestyring

Radarmodellen inneholder i denne første versjonen kun mulighet for roterende antenne med fast elevasjonsvinkel. Ytterligere antenner vil bli implementert etter hvert. Radarprosessering bygger opp en sum av det reflekterte signalet og støy, via radarlikningen, og modellert clutter som resulterer i et simulert signal som prosesseres videre for analyse og deteksjon. I gjeldende versjon av programvaren er det prioritert først å få på plass MTI-prosessering og CFAR-prosessering som vil bli mer utfyllende presentert i punktene under.

3.3.1 Lobestyring

Gitt PRF og rotasjonsfrekvens i inputfilene vil kall til BeamScheduler lage en liste av radarlober med riktig pekeretning for hver enkelt puls som funksjon av tid. Ved kall til metode for å hente neste beam velges neste element i listen med riktige parametre. Dette sammen med posisjon til mål danner geometrien for gitte puls og er utgangspunktet for å beregne signalnivåer og videre prosessering. For at ikke simuleringen skal bruke svært lang tid sjekkes det om målet er innenfor lobens belyste volum før videre prosessering initieres. Da mesteparten av pulsene sendes i retninger hvor det ikke er noe mål vil dette reduserer tiden betraktelig. For en radar som har et mer komplisert utlegg av radarlober er det naturlig å implementere utvidede metoder i denne klassen som kan representere ulike mønstre på sikt.

3.3.2 MTI-prosessering

Tilstedeværelse av clutter i mottatt signal gjør at det vil være naturlig for en radar å implementere prosesseringsmetoder for å dempe statisk clutter. Simulatoren kan derfor settes opp til å foreta en MTI (Moving Target Identification) filtrering i et forsøk på å undertrykke refleksjoner fra statiske mål. Det enkleste MTI-filteret tar utgangspunkt i det nåværende reflekterte signalet og trekker fra det foregående signalet. Over to pulsperioder vil kun eventuelle endringer forårsaket av bevegelige objekter skille seg ut i det filtrerte signalet og vil kunne detekteres enklere. MTI-filtrenes viktigste egenskap består i å redusere statiske refleksjoner og øke gainet for bevegelige mål. Som en bivirkning, vil også støy bli skalert noe opp.

Det er implementert tre forskjellige MTI-filtre som tar utgangspunkt i en, to eller tre foregående pulser. Bruk av lengre MTI-filtre resulterer i et skarpere skille mellom demping og forbedring. Antall pulser tilgjenglig for videre prosessering reduseres derimot avhengig av hvor mange pulser som først må igjennom MTI filter. Simulatoren vil avvente inntil minimum antall nødvendige pulser er tilgjengelig for MTI filtrering før det foretas videre prosessering og deteksjon.

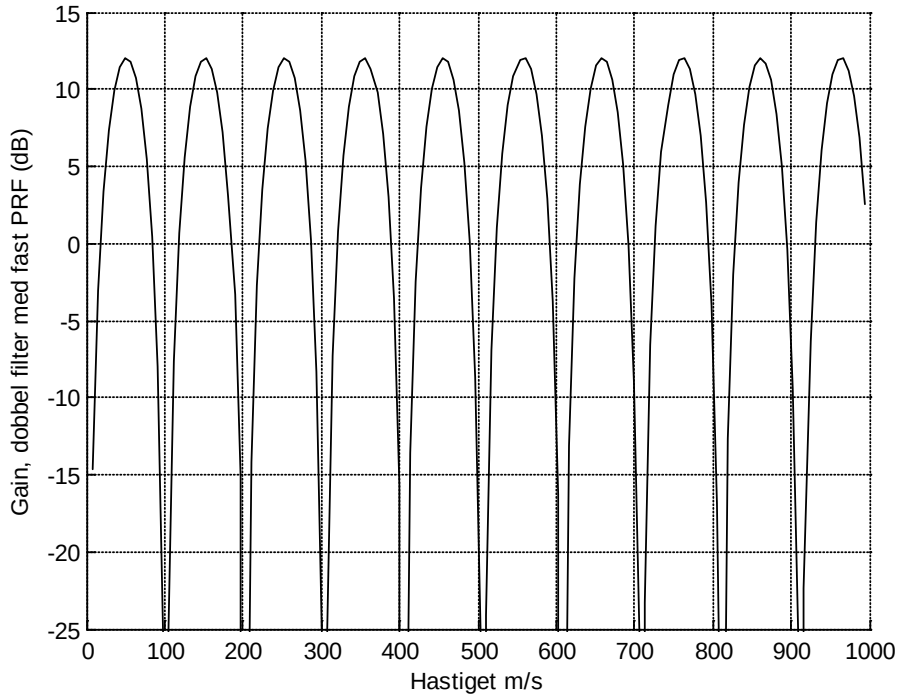
Følgende filtre er implementert:

Én-puls filter, ”single”: $\hat{S}[k]^t = S[k]^t - S[k]^{t-1}$

To-puls filter, “double”: $\hat{S}[k]^t = S[k]^t - 2S[k]^{t-1} + S[k]^{t-2}$

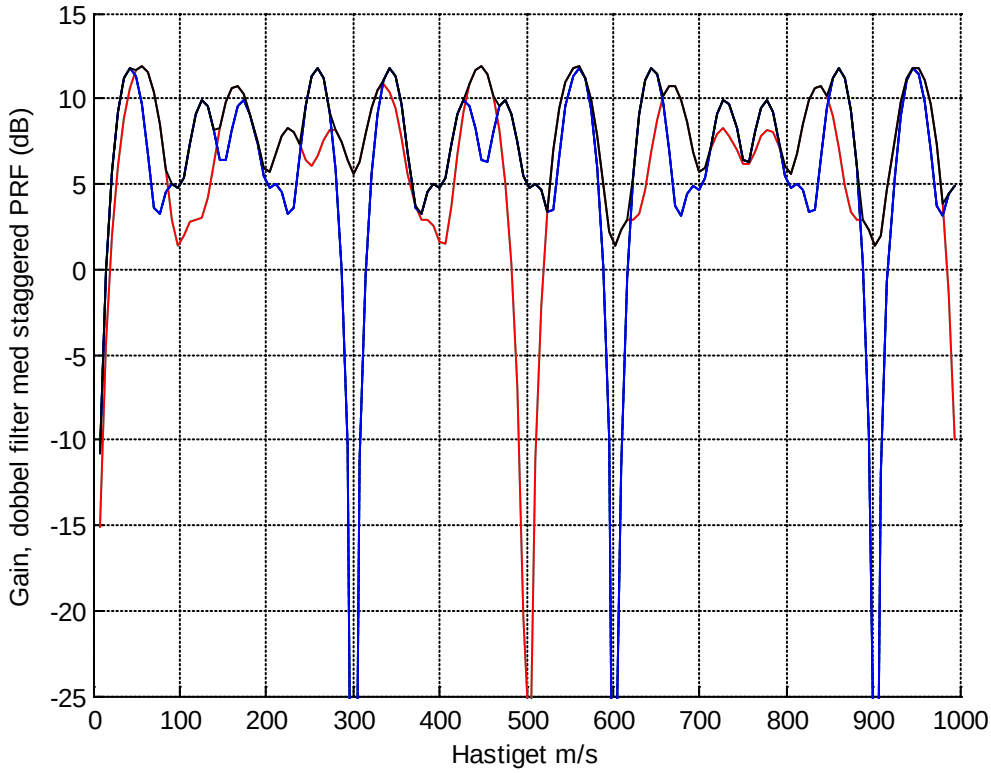
Tre-puls filter, “triple”: $\hat{S}[k]^t = S[k]^t - 3S[k]^{t-1} + 3S[k]^{t-2} - S[k]^{t-3}$

Et typisk MTI-filter demper refleksjoner med lave hastigheter men folding og tvetydigheter fører til at objekter med andre høyere hastigheter også vil kunne oppleve en demping. Demping eller forbedring som funksjon av hastighet vises i filterresponsplottet i Figur 3.5.



Figur 3.5 Filterrespons ved bruk av fast PRF.

For å redusere nullpunkter ved hastigheter større enn null kan radaren emittere pulser med forskjellig tidsforsinkelse (staggered PRF). Dette kan simuleres i programmet og den effektive PRF-verdien kan endres fra puls til puls med brukerdefinerte verdier. Med varierende PRFer vil filterresponsen til MTI-filteret ha færre blindhastigheter og de blir spredt avhengig av hvilke pulser som blir kombinert. En typisk kombinasjon med forholdstall 5:4:3, hvor man reduserer og øker PRF med 25% i forhold til det opprinnelig, gjennom to-puls MTI-filter gir for eksempel følgende filterrespons som vist i Figur 3.6.



Figur 3.6 Filterrespons ved bruk av adaptiv PRF.

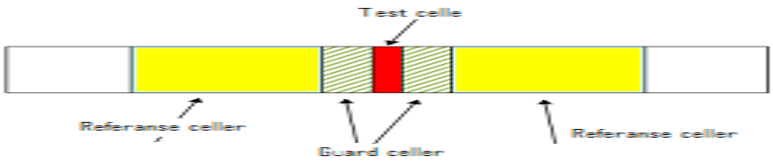
```
prf_rotvec = [25, 0, -25, 0, 25, 0, -25, 0];
```

Økning og reduksjon i PRF med 25% med den angitte kombinasjonen gir to forksjellige typer filterresponser som tegnet med rødt og blått i figuren. Den ene kombinasjonen (blå strek) resulterer i blindhastigheter rundt 300 m/s, 600 m/s osv mens det andre alternativet (rød strek) gir nullpunker ved 500 m/s, 1000 m/s osv. Ved å integrere bidrag fra to av kombinasjonene, som vil forekomme ved annenhver puls, kan man unngå at et mål konsekvent vil kunne bli undertrykt. Det vil likevel fortsatt kunne forekomme kombinasjoner hvor et mål blir dempet ned, som i dette eksempelet ved 1500 m/s (ikke vist i figuren). Dette må man være klar over på forhånd og hvis det er snakk om mulige forventede hastigheter velge en annen type staggered PRF kombinasjon.

Ikke-koherent integrasjon av et gitt antall pulser gjennomføres før deteksjon. Til gjengjeld vil det være færre pulser tilgjenglig for å foreta en deteksjon.

3.3.3 Constant False Alarm Rate (CFAR) Deteksjon

Mange radarsystemer bruker CFAR (Constant False Alarm Rate) metoder for å anslå en deteksjon og det er implementert flere varianter i simulatoren: `CA-CFAR/GO-CFAR/OS-CFAR.` Fordelen med CFAR er at tersklingsnivået bestemmes adaptivt og forandrer seg fortløpende med endringer i clutter og støy. Dette gjøres ved at algoritmen sammenligner energinivået for en gitt testcelle med visse referanseceller, både foran og bak testcellen, og avgjør om energinivået i testcellen er signifikant stort for å kunne tilfredsstille en positiv deteksjon.



Figur 3.7 CFAR celle-struktur med testcelle, guardceller og referanseceller

Et viss antall naboceller (såkalte guardcells) til testcellen ignoreres for å unngå eventuelle energilekkasjer fra målet. Hvor mange celler som skal neglisjeres og hvor mange referanseceller som skal inkluderes foran og bak testcellen defineres av brukeren.

Referansecellene anvendes litt forskjellig i ulike CFAR algoritmer. Den aller enkleste metoden går på å ta et gjennomsnitt av energien i alle referansecellene og sammenligne med energien i testcellen. Det er implementert tre forskjellige CFAR-algoritmer, og de subtile forskjellene mellom CFAR-metodene er beskrevet nedenfor.

3.3.3.1 CA-CFAR

Cell-Averaging (CA)-CFAR består i at man tar et visst antall referanseceller på begge sider og regner ut gjennomsnittsverdien. Energinivået i en testcelle må dermed overstige dette snittet med en viss skaleringsfaktor for å få en indikasjon for positiv deteksjon.

3.3.3.2 GO-CFAR

I Greatest Of (GA)-CFAR ser man på et visst antall celler på begge sider (foran og bak) og tar et separat gjennomsnitt for cellene bak og foran. Energinivået i en testcelle må overstige den største verdien av de to med en skaleringsfaktor for å få en positiv deteksjon.

3.3.3.3 OS-CFAR

I Ordered Statistics (OS)-CFAR benyttes referanseceller på begge sider og de blir sortert i stigende rekkefølge. Den k'te verdien trekkes ut, og energien i en testcelle må overstige dette nivået med en skaleringsfaktor for å få en positiv deteksjon. Verdien til k er satt til ¾ av antall referanseceller i simulatoren som anses som optimal i forhold til visse kriterier [8].

4 Sjøscenario med parametervalg

Denne delen av rapporten tar for seg et konkret simuleringscenario og går i detalj gjennom de fleste parametere i simulatoroppsettet. Formålet er å beskrive bruken av simulatoren ved hjelp av parameterfilene, noe som også gir en innføring i parametervalg.

I scenarioet antas det at en cosecant roterende antenne er påmontert et skip 30moh. Et sjømålsmissil flyr mot radaren med en hastighet på 310 m/s med høyde 5moh. Radartverrsnittet (RCS) antas å være Swerling 3 fordelt med middelverdi på 0.01m2 (-20dBm).

Sjøen modelleres som tilstand 4 (sea state 4) noe som betyr betydelig clutter, særlig i nærområdet til radaren. Scenario og missilbanen er laget i STK.

Følgende filer leses inn ved kjøring av Main filen:

Simulatorparametre:
Miljøparametere:
Sjøclutterparametere:
Radarparametere:

4.1 Radarparametre som defineres i RDR1.txt

`mode` = montecarlo % montecarlo (random numerical) / analytical (hybrid analytical/statistical based)

`det_method` = Sw3 % Target model / Detection method (Sw0-Sw4)

`coh_interval` = 15 % Sw1/Sw3: RCS coherent/correlated across # of pulses (as long as within one scan)

`sw_model` = normal0.99 % RCS pulse-to-pulse correlation model (none/normalXXX, (1-XXX)=possible change in % from pulse to pulse)

Den første parameteren avgjør simulatormodus, her angitt som MonteCarlo, som tilsvarer den numeriske simuleringen basert på uttrekning av tilfeldige valg. Andre valg gjennomgås ikke i denne rapporten.

Det andre valget angir målets antatte tverrsnittfordeling som er satt til en Swerling 3 distribuering. Andre valg som kan angis her er `Sw0-Sw4`.
`Sw0` representerer konstant RCSverdi, mens ved `Sw2/S4` trekkes det ut nye tilfeldige tall fra Swerling 2 eller 4 fordelingen for målets RCS ved transmisjon av hver eneste puls.
I `Sw1/Sw3` er den trukne RCSverdien konstant for et visst antall pulser og det trekkes ut nye tilfeldige tall først etter at koherensperioden, angitt av `coh_interval` er over, eller eventuelt at målet kommer ut av radarbeamen.

For `Sw1/S3` er det videre modellert en enkel puls til puls korrelasjonsmodell noe som muliggjør tilfeldige små variasjoner over korte tidsintervall. Til dette formål brukes parameteren `sw_model` som kan settes til `none`, altså ingen modellering fra puls til puls. Alternativ kan den settes til `normal`XXX hvor XXX er et tall mellom 0 og 1 og angir graden av endring fra puls til puls. Mer spesifikt:

$$\sigma^{t+1} = \sigma^{t}(1+(1-\rho)\tilde{n}) \tag{4.1}$$

ρ (XXX) er spesifisert vektfaktor, mens $\tilde{n}$ trekkes tilfeldig og er normalfordelt.

Følgende grunnleggende radarparametere er stort sett selvforklarende og er hovedsakelig relatert til oppbygningen av radarlikningen:

```
fc = 5.6e9                  % radar frequency
bw = 20000                  % effective receiver bandwidth
pwidth = 1*10^-6            % pulse width
power = 1*10^5              % Transmitted power (Watt)
prf = 3750                  % PRF
gain = 1584.89              % Transmit antenna gain(1584=32dB)
rgain = 1584.89             % Receive antenna gain (1584=32dB)
cgain = 2                   % pulse compression gain (2=3dB)
noiseFigure = 3.98          % added noise from receiver (3.98=6dB)
loss = 10                   % additional radar loss (10=10dB)
noiseTemp = 290             % antenna temperature [K]
theta = 0.0525              % antenna pattern beamwidth
pol = HH                    % polarization (VV/HH)
scan_time = 1.5             % Antenna scan time (s)
ant_el_angle = 0.0          % Middle of el_beam point direction
beam_mode = constantAzEl    %Mode for antenna beam
instrRange = 30000          % max radar range (m)
blindRange = 150            % min radar range (m)
u_velocity = 0              % max unambigious target velocity (0=use standard
                            Pulse-Doppler radar value,   -X = X*P-D)
```

Radarens maksimale og minste dekningsavstand angis i meter og her spesifiseres også den maksimale utvetydige Dopplerhastigheten i m/s som et mål kan ha. Det er mulig å angi 0 og simulatoren vil automatisk beregne den maksimale hastigheten basert på en antagelse om en standard Puls-Doppler radar. En negativ verdi kan brukes hvor den maksimale hastigheten settes lik den utvetydige hastigheten for en standard Puls-Doppler radar multiplisert med den positive av den angitte verdien.

```
pfa = 1*10^-6               % Probability of false alarm
```

Ønsket sannsynlighet for falsk alarm rate, pfa. Per i dag er denne ikke modellert. I CFAR algoritmene angis tersklingsnivået direkte.

```
% clutter / MTI filter:
shapemodel = none           % clutter shape model (none/ward)
shapefactor = 1             % default clutter shape parameter(0.7=spiky,20+=Rayleigh)
mtifilter = 1               % 0 = no MTI clutter filter, 1 = enable MTI clutter filter
mti_type =  double          %MTI filter type (none/single/double/triple)
mti_max = 40                % maximum MTI improvement factor (dB)
c_model = normal0.99        % clutter pulse-to-pulse correlation model(simpleXXX/
                            normalXXX, XXX=correlation factor [0,1])
```

`Shapemodel` angir modellen som brukes for å bestemme clutter formfaktoren *v* for K-fordelingen. Alternativt kan `shapemodel` settes til `none` og formfaktoren angis direkte som et tall i `shapefactor` for alle mulige avstander.

`mti_type` bestemmer hva slags MTI filter som skal anvendes, enten en-, to- eller tre-pulsfilter; mens `mti_max` angir makismal mulig gevinst fra filtrering. Årsaken til dette er at praktiske systemer ofte vil ha begrensning i forhold til for eksempel antall bit som brukes for sampling og maksimal clutter demping vil typisk ha en øvre grense.

`c_model` brukes for å bestemme puls til puls clutterkorrelasjon i løpet av koherensperioden. Her angis korrelasjonsfaktoren i tillegg til modellen.
`simpleXXX`: K-fordeling, `normalXXX:` Normal fordeling.
Dette modelleres som

$$c^{t+1}[k] = c^{t}[k](1 + (1-\rho)\tilde{n}), \tag{4.2}$$

hvor ρ er korrelasjonsfaktoren (XXX) mens $\tilde{n}$ er enten normalfordelt eller K-fordelt med den opprinnelige fordelingen. Større ρ-faktorer vil naturligvis føre til større svingninger og clutterdempende filtre vil ha mindre effekt.

% -- parameters for montecarlo mode:

`rbins` = 400	% number of sampling range bins
`coh_interval` = 15	% Sw1/Sw3: RCS coherent/correlated across # of pulses (as long as within one scan)
`coh_interval_clutter` = 15	% clutter coherent period/correlated across # of pulses (as long as within one scan)

Antall samplingsceller og koherensperioden angis. Koherensperiode definerer den tidsperioden hvor det antas at clutterverdiene og RCS forandres lite. Denne perioden angis i antall emitterte pulser og det trekkes ut nye tilfeldige verdier først etter at koherensperioden er over. Innenfor dette tidsrommet anvendes puls til puls korrelasjonsmetoder som beskrevet.

`adaptive_prf` = 1	% 1=use adaptive PRF, 0=fixed PRF
`prf_rotvec` = 0,-25,0,25	% PRF change in percentage from pulse to pulse (comma separated), suggested: 0,-25,0,25 if using double MTI filter (5:4:3 combination)

Simulatoren gir mulighet for adaptiv PRF som kan slås av eller på. Adaptive PRF verdier angis i prosent i forhold til opprinnelig verdi og PRF endringene simuleres fra puls til puls. En kombinasjon som 0,-25,0,25 innebærer at den første pulses sendes med den opprinnelige PRF verdien mens den andre pulsen har en effektiv PRF verdi redusert med 25%. Så går raten opp til den opprinnelige verdien og til slutt økes den med 25%. Ved neste emisjon itereres det fra begynnelsen.

Parametere for deteksjon bestemmes i den siste parameterblokken.

detection_mc = os-cfar	% Detection method (thresholdXX: -XX=dBm power level, CA-CFAR/GO-CFAR/OS-CFAR
`npulse` = 3	% # of pulses to integrate over
`rcells` = 3	% CFAR: reference cells on each side
`gcells` = 1	% CFAR: guard cells on each side
`scalingf` = 12	% CFAR: detection scaling factor [K] (dB)
`swindow_off`=0	% 1=turn OFF sliding window integration across unequal PRF blocks

`detection_mc` angir deteksjonsmetode, her kan det velges mellom en enkel tersklingsmetode eller CFARmetoder som CA-CFAR, OG-CFAR eller OS-CFAR. `thresholdXX` er en enkel deteksjonsmetode hvor energinivået for hver enkel celle undersøkes om den overstiger verdien `XX.` I så fall har vi en positiv deteksjon. `XX` kan fritt settes til en verdi mellom 1 og 99, angitt i dB. De andre valgene er CFAR metoder.

`npulse` bestemmer antall pulser som kombineres ikke-koherent før deteksjonsalgoritmen kjøres.

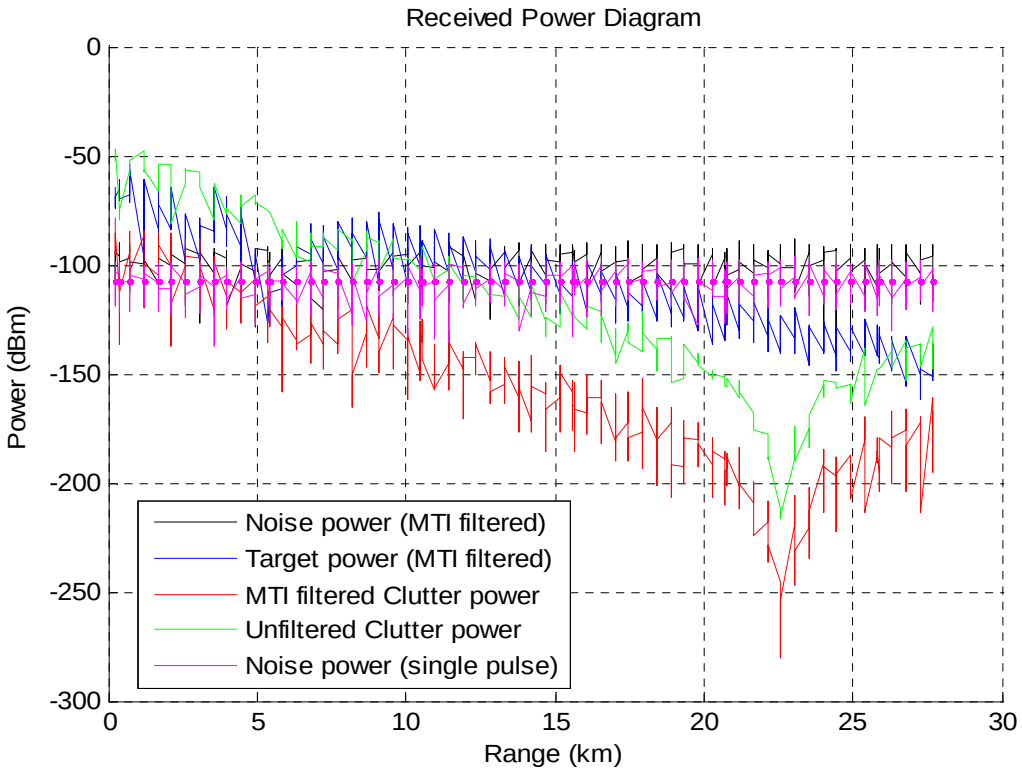
Parametere av betydning for CFAR algoritmene er:
`rcells:` Antall referanseceller som tas foran og bak testcellen.
`gcells:` Antall naboceller som ignoreres foran og bak testcellen.
`scalingf:` Skaleringsfaktor for deteksjon angitt i dB. Energien i en testcelle må overstige gjennomsnittnivået i referansecellene med denne faktoren for å få en positiv deteksjon. Valget av denne faktoren har sammenheng med falsk alarm rate, noe som kan plukkes ut fra tabeller eller beregnede plott.

5 Simuleringskjøring:

Etter at alle parameterfilene er satt vil man kunne starte simulatoren med å eksekvere MainRadarModelling.m. En simulering vil startes og flere figurer vil komme frem etter endt kjøring.

5.1 Signalnivå, clutter og noise

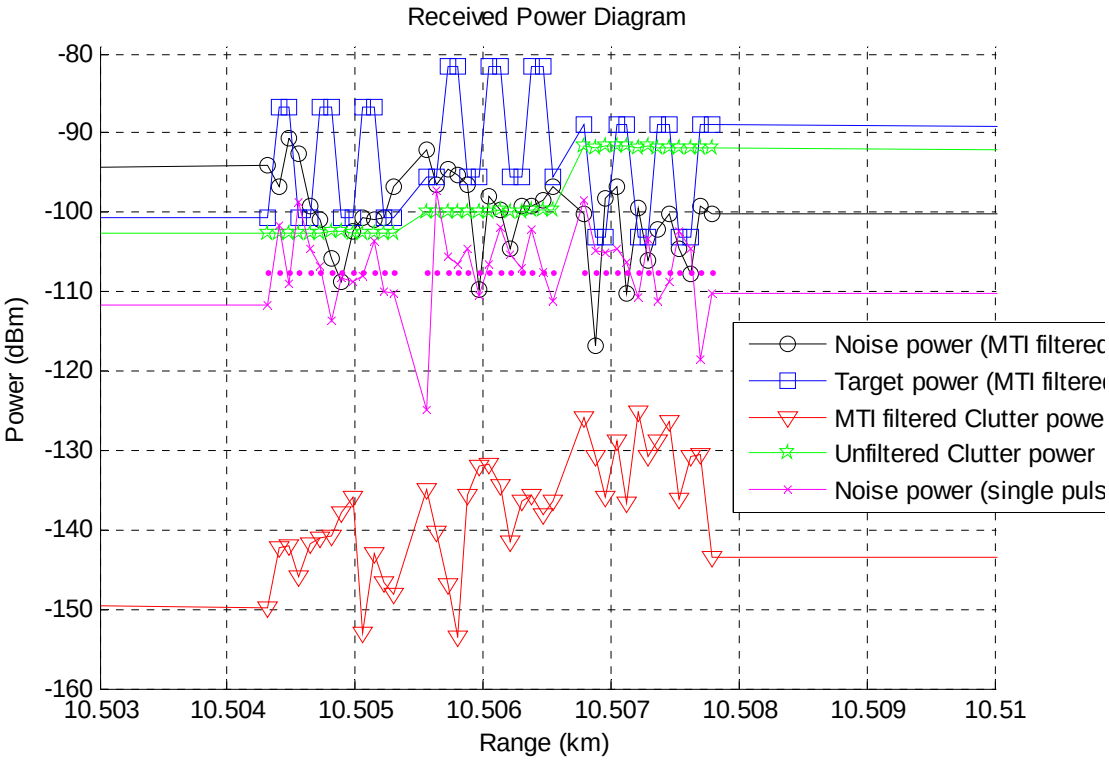
”Received Power Diagram” er at av de mer sentrale plottene og viser energinivåene for målets refleksjon, clutter (før og etter MTI filtrering) og støy som funksjon av avstand til målet. Energinivåene er beregnet for avstandcellen hvor målet befinner seg.



Figur 5.1 Mottatt energi før og etter MTI filter

Vi ser i Figur 5.1 at målets reflekterte energi (blå kurve) øker etter hvert som den nærmer seg radaren men er samtidig begrenset av propagasjonsfaktoren (Figur 3.2), noe som fører til kraftig demping i 5-7 kilometers området. MTI-filteret er med på å dempe clutteret betraktelig vist fra grønn kurve som viser energinivåene før filtrering og rød etter. Målets energi overstiger clutter- og støy-nivåene kun i avstandsregionene 0-5 km og 7-11 km. Uten MTI-filter hadde målet stort sett vært skjult av clutter og støy bortsett fra i den siste regionen på 7-11 km, men også da med et kraftig innslag av clutter. Deteksjon er ellers støybegrenset på en avstand lengre enn 14-15 kilometer.

Zoomer man inn på figuren ser det ut som i Figur 5.2:

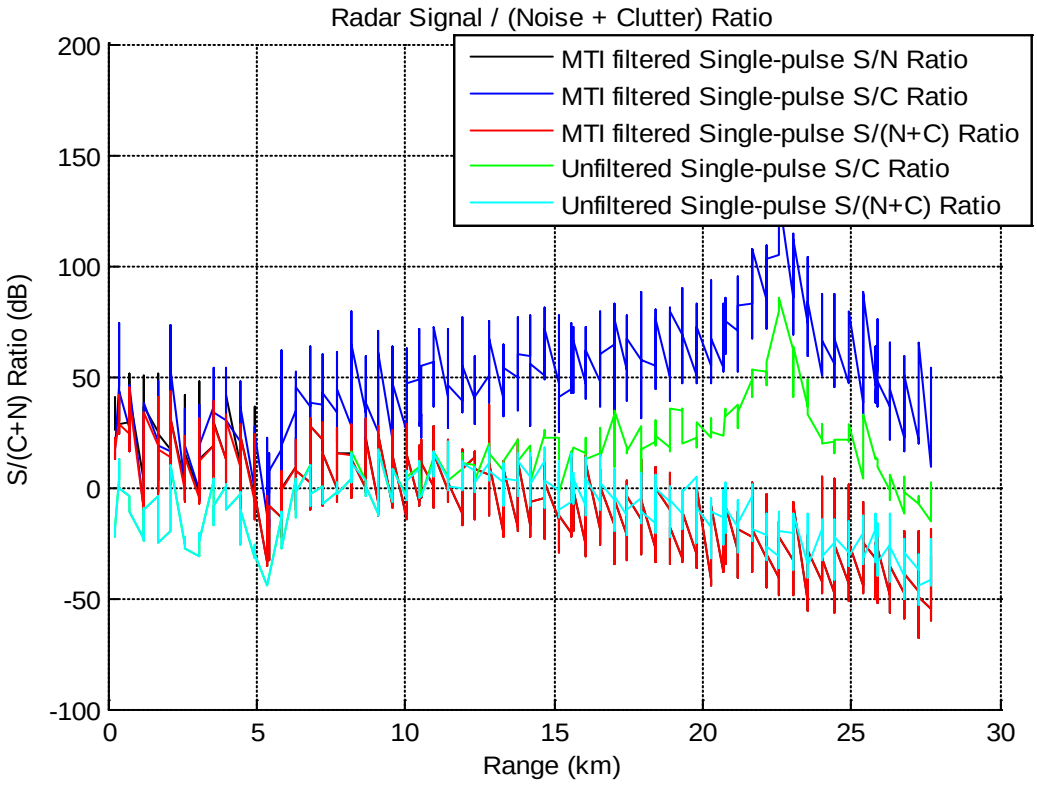


Figur 5.2 Mottatt energi før og etter MTI filter, forstørret

Her kommer det ganske klart frem at energien i det reflekterte målsignalet (blå) hopper konsekvent opp og ned mellom to nivåer, noe som skyldes at det er to filterkombinasjoner fra adaptiv PRF og påfølgende MTI filtrering. Avhengig av pulskombinasjonene fører det til enten demping eller forbedring. Uten adaptiv PRF kunne man ha havnet i en situasjon hvor målet kun dempes eller skaleres.

Det er ellers områder både til høyre og venstre for plottet hvor det er relativ stille noe som skyldes radarens rotasjon og at målet dermed kun er i antenneloben for bestemte avstander. Målet er innenfor loben i en periode på ca 40 pulser pr scan, mens i denne simuleringen antas at koherensperioden er på 15 pulsintervaller. Dette ser man klart i den grønne kurven hvor det er tre adskilte nivåer for ufiltrert clutter, og clutteret er mer eller mindre konstant i denne perioden.

Figur 5.3 viser signal-til-støy (SNR) og clutter-til-støy (CNR) forholdet.

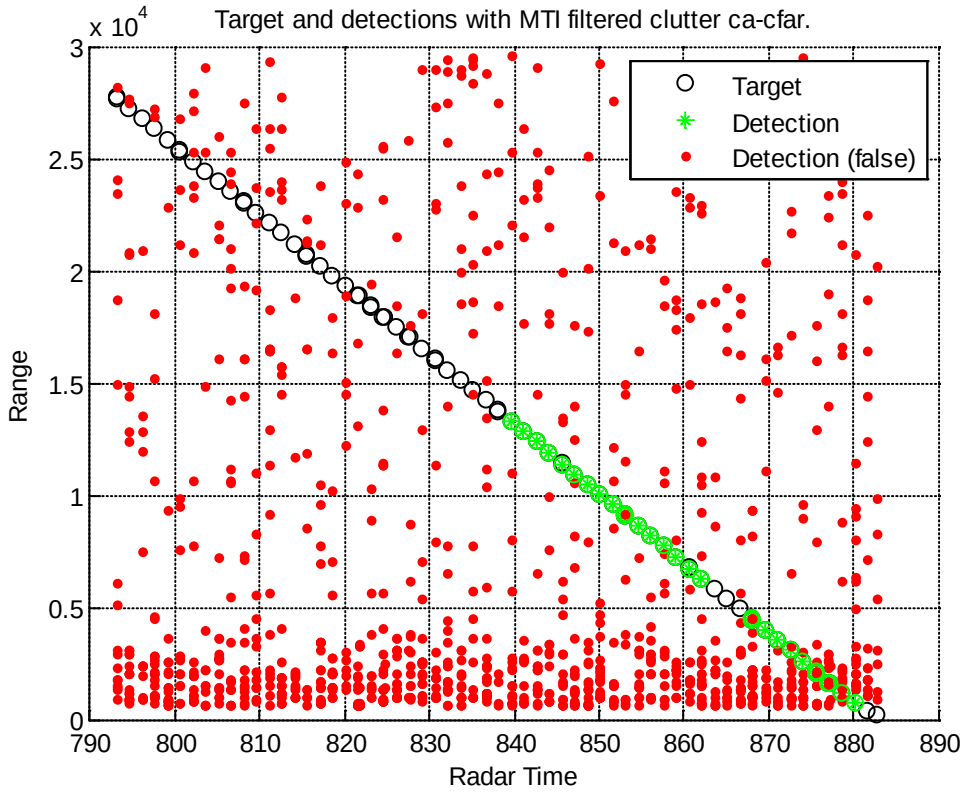


Figur 5.3 Forholdstall mellom signal, clutter og støy (SNR/CNR)

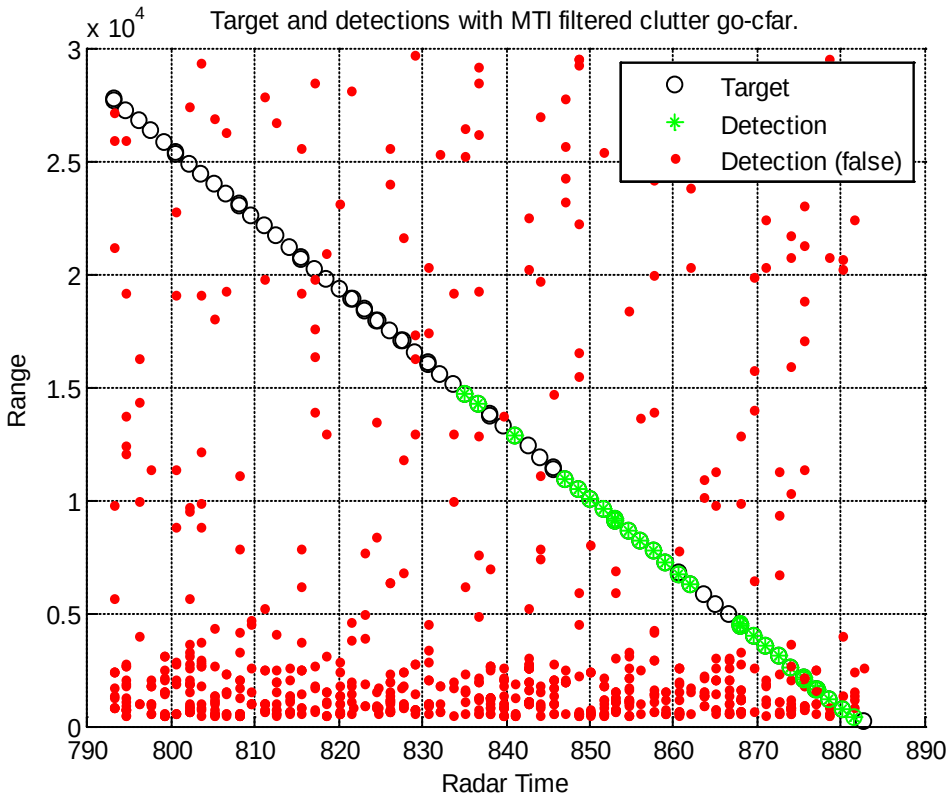
5.2 Deteksjonsplott for forskjellige CFAR deteksjonsmetoder:

Deteksjonsplott i Figur 5.4 til Figur 5.7 viser hovedresultatene fra CFARdeteksjon og figurene er komprimert i tid slik at deteksjonsresultatene vises kun for den perioden målet befinner seg i radarbeamen. En svart sirkel angir målets reelle posisjon, mens en deteksjon angis med rødt eller grønt. Rødt når algoritmen returnerer en falsk deteksjon, mens en grønn sirkel angir en korrekt deteksjon.

Tersklingsnivået i CFARalgoritmene er satt til `scalingf = 12`.
Resultater med CA-CFAR og GO-CFAR deteksjonsmetoder uten integrasjon av pulser er vist i Figur 5.4 og Figur 5.5.



Figur 5.4 Resultat fra deteksjonsplott med CA-CFAR, ingen puls integrasjon.

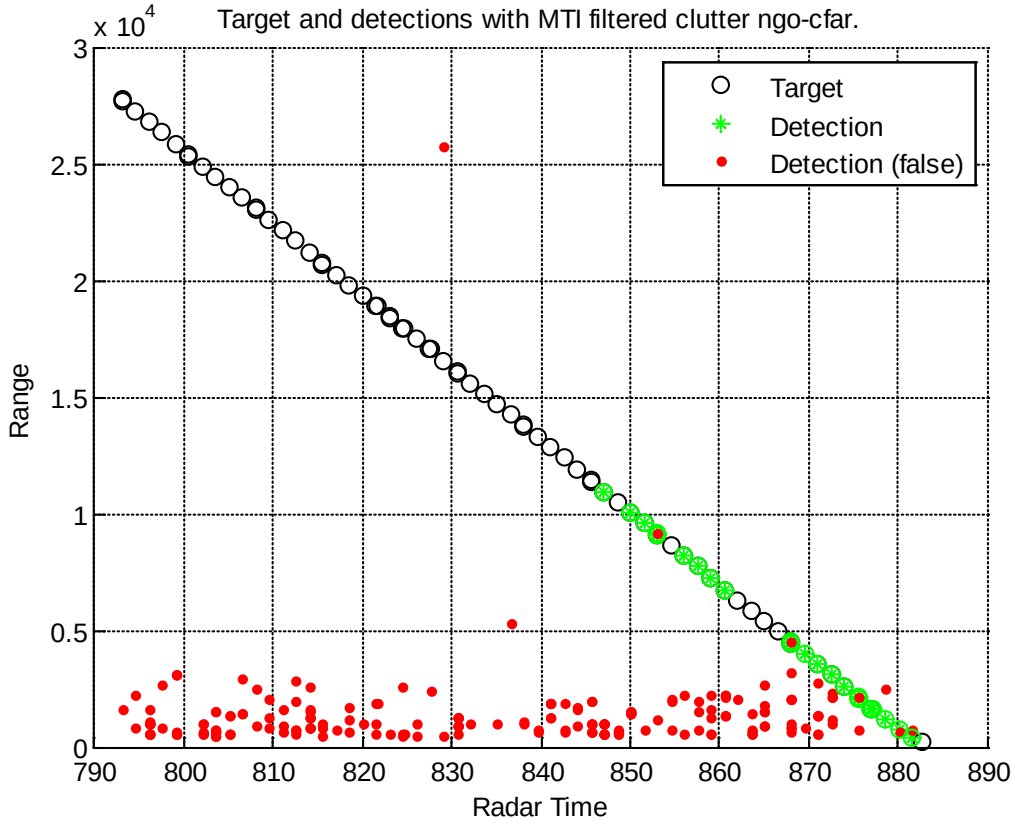


Figur 5.5 Resultat fra deteksjonsplott med GO-CFAR, ingen puls integrasjon.

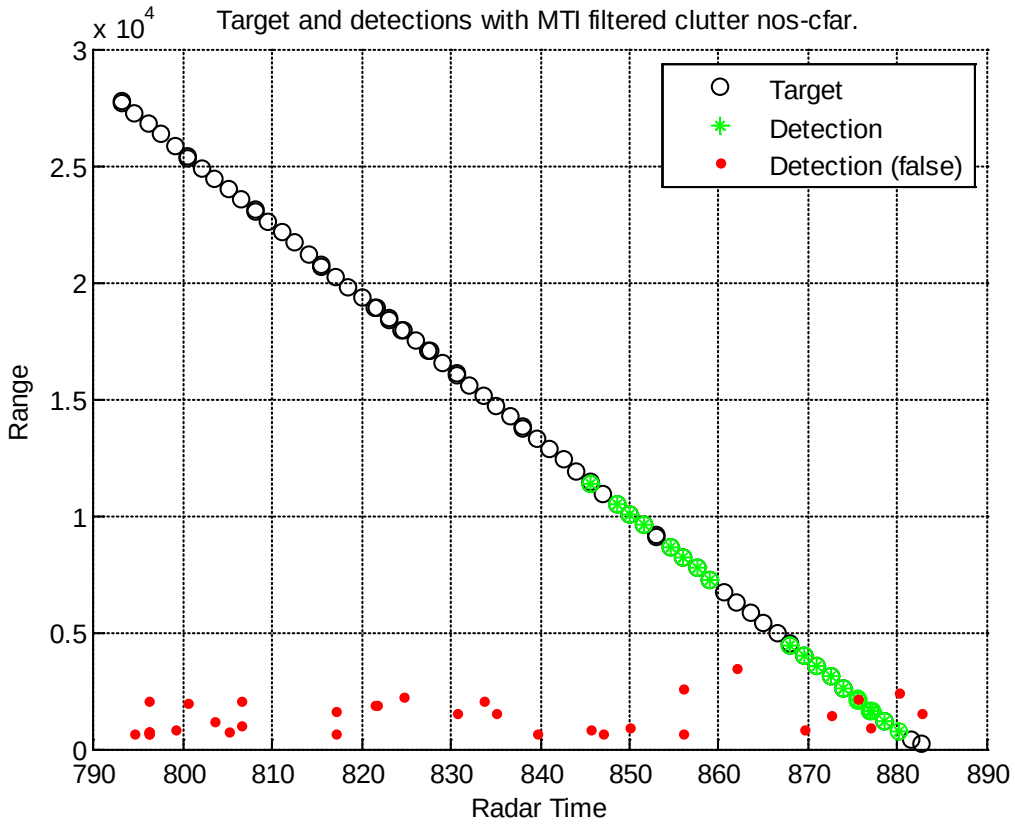
Vi ser at det er ganske mange feildeteksjoner. Dette skyldes at det ikke er noen integrasjon av pulser og i enkelte pulser vil målet være kraftig dempet mens i andre oppleve noe forbedring. Missilet blir i alle tilfeller detektert på ca 14-15 kilometers avstand, men med så stor feildeteksjonsrate er det usikkert om en operatør eller trackingalgoritme hadde klart å fange den opp helt i ytterkant.

For å redusere feilraten kan en for eksempel integrere flere pulser, før CFARalgoritmen kjøres, dette vil jevne ut clutternivåene og støyen i tillegg til å midle målets filtrerte energi.

Kjøring med GO-CFAR og OS-CFAR med integrasjon av tre MTI-filtrerte pulser resulterer i plottene vist i Figur 5.6 og Figur 5.7.



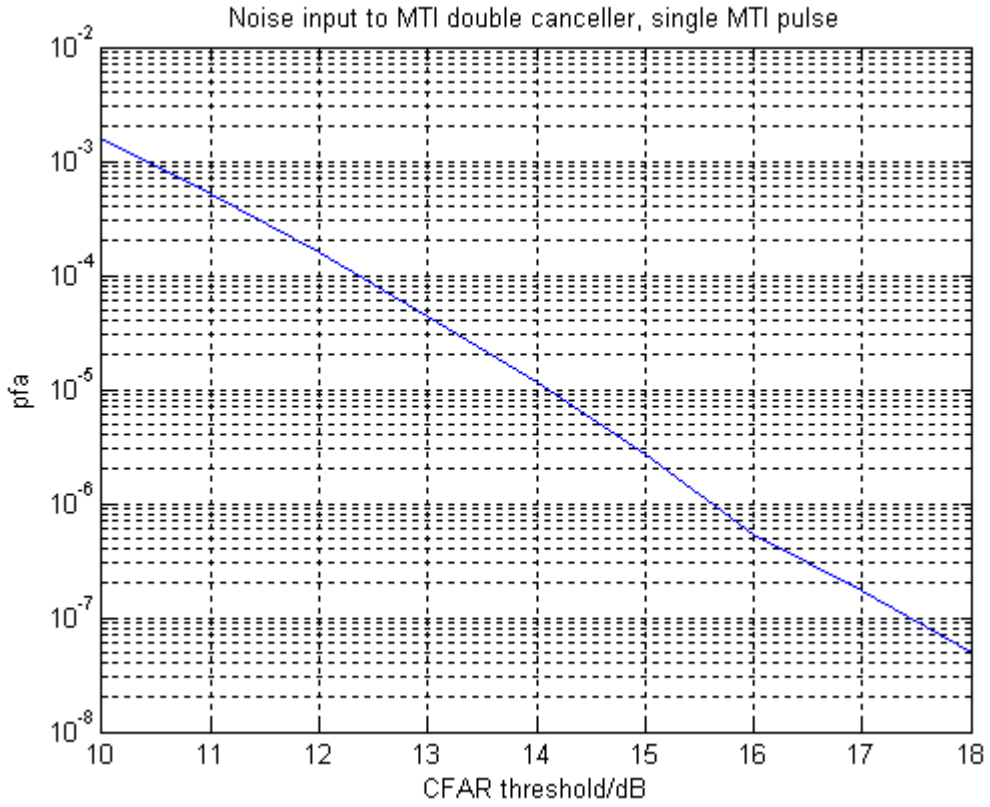
Figur 5.6 Resultat fra deteksjonsplott med GO-CFAR og integrasjon over tre pulser.



Figur 5.7 Resultat fra deteksjonsplott med OS-CFAR og integrasjon over tre pulser.

Vi ser at feildeteksjonsraten faller kraftig og målet blir detektert noe nærmere, nå først ved ca 11-12 kilometers avstand. Det er fortsatt enkelte feildeteksjoner i nærhet av radaren, noe som i praksis ofte dempes ned med et egnet filter som justerer nivået på nærområdet (STC = Sensitivity Time Control). Disse plottene vil kunne gi god pekepinn på hvordan denne radaren oppfører seg og deteksjonsevnen i forhold til det innkommende missilet.

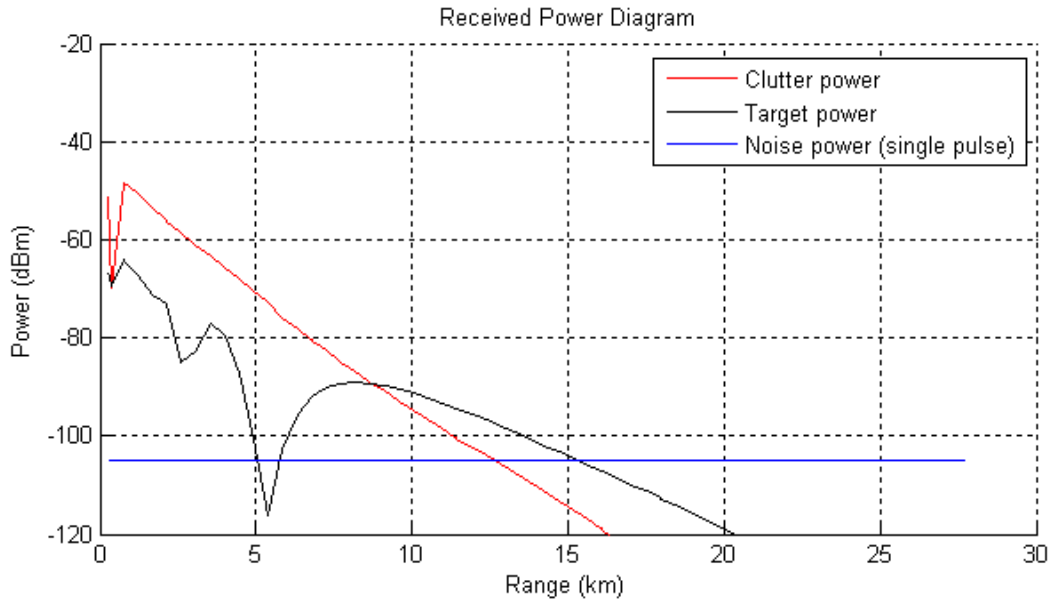
Tersklingsnivået i CFARmetodene relatert til pfa (probability of false alarm – falsk alarm rate) kan leses ut fra simuleringer med støy og clutter som vist for et eksempel av parametre i Figur 5.8. Liknende plot kan genereres for andre prosesseringsvalg, støynivåer og clutteregenskaper. En mer grundig gjennomkjøring med ulike settinger bør gjøres på et senere stadium for å få en forståelse av hva tersklingsnivået bør være i ulike simuleringer.



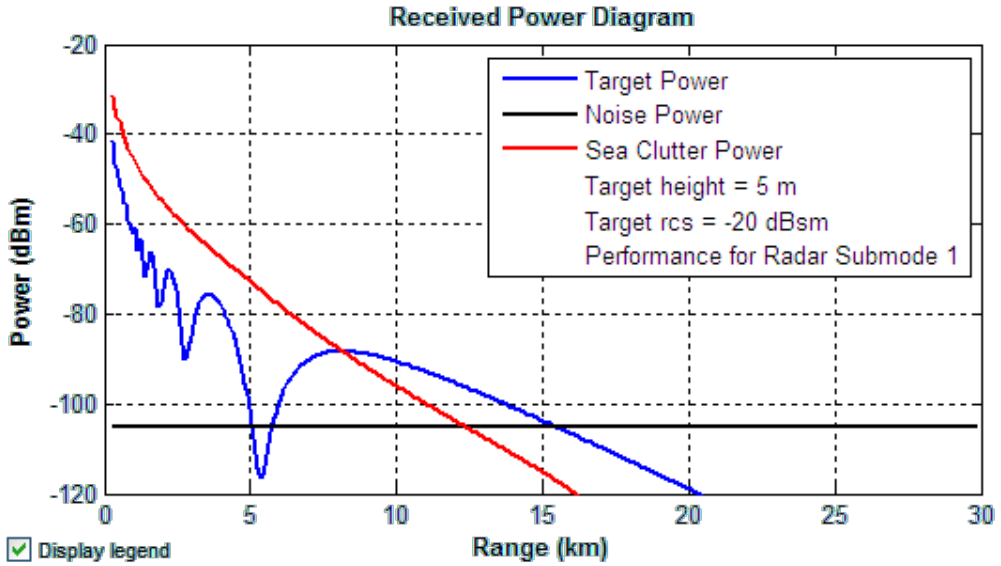
Figur 5.8 Probability of false alarm (pfa) som funksjon av CFAR terskelnivå.

Vi ser at et tersklingnivå på 12 gir en pfa på ca 10^{-4} for disse valgene.

5.3 Sammenligning av resultater mot SADM default radar simulering

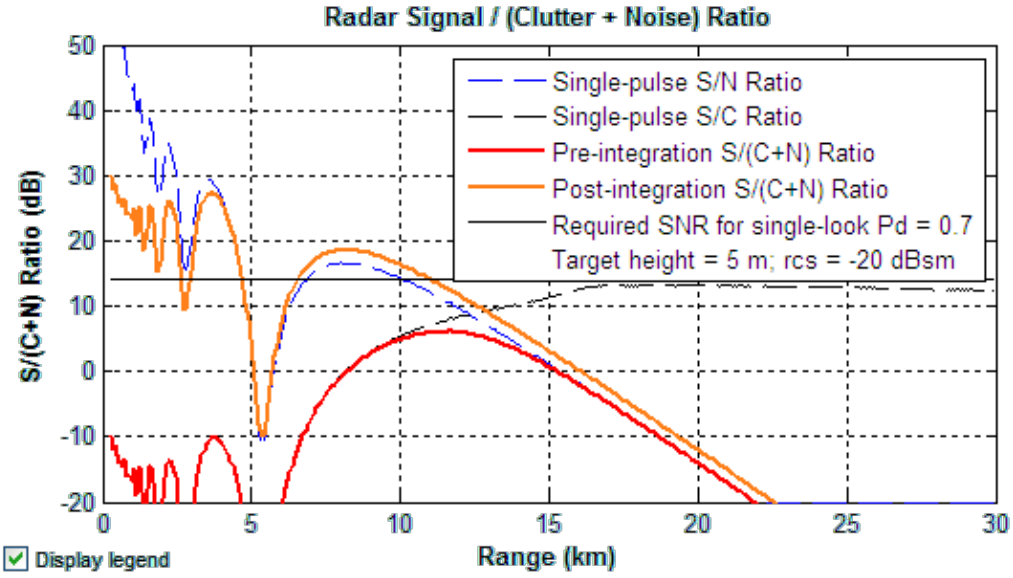


Figur 5.9 Midlere mottatt signalenergi i radarmodellen beskrevet i denne rapporten



Figur 5.10 Mottatt signalenergi ved bruk av SADM modellen

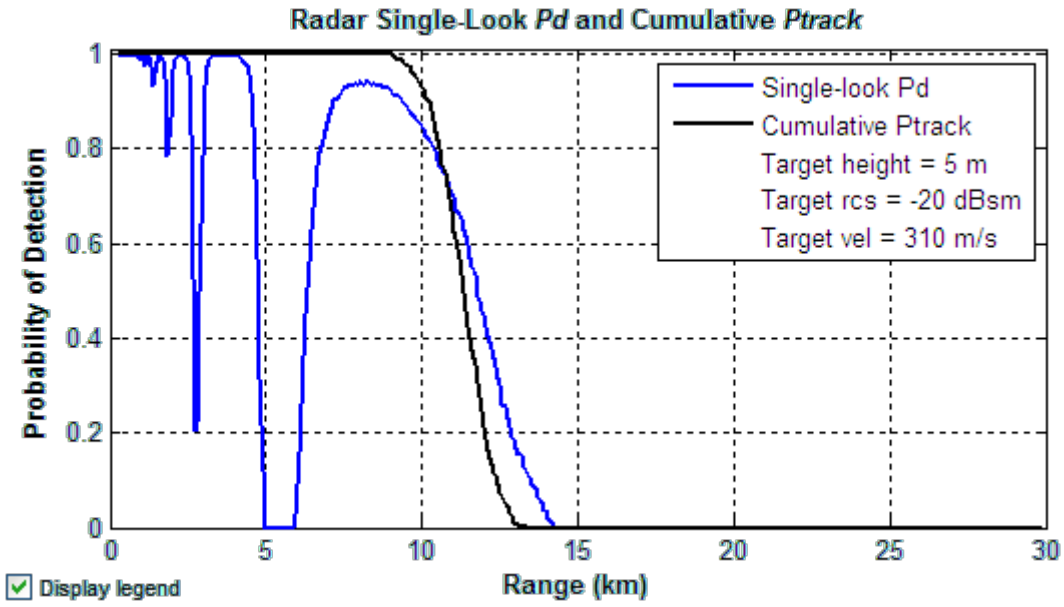
Figur 5.9 og Figur 5.10 viser mottatt energi for clutter, støy og mål som funksjon av avstand for henholdsvis radarmodellen presentert her og SADM. I radarmodellen er alle fluktuasjoner tatt bort og kun snittnivåene er presentert for å gjøre mer lesbart i denne sammenhengen. Vi ser at de to modellene er godt overens om det vesentligste forløpet. Vår modell representerer en iterasjon hvor målet nærmer seg radar og har således en avlesning hver runde for antennen som tilsvarer ca hver 450 m langs banen, mens SADM har en mer kontinuerlig kurve. Vi ser at vi finner igjen propageringsfaktoren som vist i Figur 3.2 som en vesentlig former av forløpet i begge modellene.



Figur 5.11 S/N, S/C og S/(C+N) fra SADM for enkelt puls og etter integrasjon.

I Figur 5.11 vises resultatplott fra SADM for S/N, S/C og S/(C+N) før og etter integrasjon/prosessering. Resultatene fra modelleringen viser liknende oppførsel, men på grunn

av fluktuasjonene som er implementert i modellen for clutter og støy er plottene mer dynamiske, se Figur 5.1 til Figur 5.3.



Figur 5.12 Sannsynlighet for deteksjon og kumulativ sannsynlighet for tracking fra SADM

Figur 5.12 viser beregningene i SADM for Pd (sannsynlighet for deteksjon). I det gitte eksempelet med hovedparametre lik denne kjøringen vil modellen gi resultater etter CFAR som vist i Figur 5.4 til Figur 5.7. Vi ser at modellene er enige om på hvilken avstand man kan forvente deteksjoner.

5.4 Definisjon av scenario og målbaner

Scenarioet som er kjørt her, som et første testscenario, har blitt generert fra STK. Dette ble valgt innledningsvis som et mulig miljø hvor radarer kan plasseres i terrenget og ulike målbaner defineres gitt eksisterende kunnskap i STK om oppførsel. Denne delen av det totale systemet er ikke tillagt stor vekt så langt og en gjennomgang av funksjonaliteten i STK, innlesingskode i modelleringsprogrammet samt å legge til metoder for import av scenariodata på andre formater vil måtte implementeres og utvides i senere versjoner.

Appendix A Klasser og metoder i simuleringsmodellen

Koden er bygget opp ved bruk av objektorientert programmering ved implementering av en rekke klasseobjekter i Matlab. De enkelte klasseobjektene og deres metoder vil kort bli referert i dette appendikset. De klassene som er definert er listet under. Dette representerer en klassestruktur som er valgt for denne første versjonen av programvaren. En evaluering vil bli gjort for å eventuelt strukturere annerledes etter ferdigstillelse av første versjon.

`Classdef` Simulation
`Classdef` Platform
`Classdef` Facility (subklasse av Platform)
`Classdef` Aircraft (subklasse av Platform)
`Classdef` Radar
`Classdef` MonostaticRadar (subklasse av Radar)
`Classdef` Antenna
`Classdef` ParabolicAntenna (subklasse av Antenna)
`Classdef` Environment
`Classdef` SeaClutter
`Classdef` LandClutter
`Classdef` MapClass
`Classdef` Atmosphere
`Classdef` BeamSchedulerClass
`Classdef` OutputCl

Alle klassene inneholder en funksjon for å sette og hente klasseegenskaper ved bruk av funksjonene:

function set(obj,varargin)
function varargout = get(obj,propStr)

A.1 `Classdef` Simulation

Klassen inneholder variable som styrer valg av ulike algoritmer for beregning, plottesettinger og klasseobjekter til de andre objektene i simuleringen. I strukturen kan den sees på som den øverste klassen hvor andre klasser kan refereres fra. Klassen har ingen metoder.

A.2 `Classdef` Platform

Platform er en superklasse som kan inneholde ulike plattformer, være seg bakkebaserte, på sjøen eller luftbårne. Den har lokale egenskaper som posisjon ((Lat, Long, Altitude) eller jordsentriske koordinater), hastighet, RCS, tid og en radarliste. Den kan således huse flere radarer. Dersom egenskapene endrer seg over tid vil være lagt inn som vektorer. Plattformen kan derfor ha enten konstante eller tidsavhengige egenskaper.

A.2.1 function add(obj,varargin)

Det kan tilordnes klasseobjekter av typen Radar til Platform instansen. Denne funksjonen kan brukes dersom en ny radar skal legges til.

A.2.2 function remove(obj,varargin)

På lik måte som funksjonen add kan legge til et radarobjekt kan denne funksjonen fjerne et radarobjekt.

A.2.3 function readDefinitions(obj,fname)

Leser inn klasseegenskaper fra definert fil i prosjektet som kjøres og tilordner til instans av klassen.

A.2.4 function getTargetState(obj, t_sim)

Henter ut posisjon, (yaw, pitch, roll), range hastighet og rcs til objektet ved tiden t_sim.

A.2.5 function getTargetAngleOffset(obj, t_sim, beam_ptr, theta)

Sjekker om target er innenfor radarbeamen. NB: I denne første versjonen sjekkes kun i asimutretning.

A.3 `Classdef` Facility (subklasse av Platform)

Dette er en subklasse av Platform som kan brukes på en fast installasjon. I eksempelet som er brukt i denne rapporten har det vært referert en radar til en instans av Facility. Posisjonen til radar var her konstant, men plassert langt til havs, så enten er det en fast installasjon eller på en båt som ligger i ro. Siden alle avlede klasser av Platform har en radarliste man kan utvide med ytterligere radarer av ulik type ved behov. Klassen har i denne første versjonen ingen ytterligere egenskaper eller metoder.

A.4 `Classdef` Aircraft (subklasse av Platform)

Dette er en subklasse av Platform med tilleggsegenskaper for yaw, pitch og roll som funksjon av tid. I eksempelet som er kjørt i denne rapporten er et laftkommende missil brukt hvor dennes flukt er beskrevet i en instans av denne klassen.

A.5 `Classdef` Radar

Klassen har mange egenskaper som refererer til prosessering, radarparametre og resultatvektorer. Det er også referanse til tilordnede Antenne og BeamScheduler instanser. Det kan tilordnes flere antenner til en radar og det er derfor muligheter for å kjøre med ulike antenner i samme simulering, men det er ikke implementert i denne fasen. Denne versjonen har hovedløkken liggende som en metode i klassen Radar og utviklingen av simuleringen er styrt av radarens antenne og dennes pekeretning.

A.5.1 function add(obj,varargin)

Det kan tilordnes klasseobjekter av typen Antenne og BeamScheduler til Radar instansen. Denne funksjonen kan brukes dersom en ny antenne skal legges til.

A.5.2 function remove(obj,varargin)

På lik måte som funksjonen add kan legge til et antenneobjekt kan denne funksjonen fjerne et antenneobjekt.

A.5.3 function readDefinitions(obj,fname)

Leser inn klasseegenskaper fra definert fil i prosjektet som kjøres og tilordner til instans av klassen.

A.5.4 function run_scenario_mc(obj, varargin)

Hovedløkken for kjøring av simuleringen i Monte Carlo mode (mc). Simuleringen kjører i tidsskritt bestemt av hvordan radarens beam flyttes rundt og scenarioets varighet er gitt av tiden target (Aircraft) er tilstede. Alternative måter å kjøre simuleringen på kan implementeres med å legge til nye metoder eller en overordnet metode med ønsket fleksibilitet på måter simuleringer ønskes kjørt. Dette er tema for videre oppfølging.

A.5.5 function run_scenario(obj, varargin)

Hovedløkken for kjøring i analytisk-statistisk/hybrid mode. Denne metoden har begrensninger fordi formlene kun gjelder for de mest generelle tilfellene og kan for eksempel ikke brukes sammen med mer sofistikert clutterundertrykking. Gangen i modelleringer går i tidsskritt på lik linje mede det som gjelder for forrige metode: run_scenario_mc.

A.6 `Classdef` MonostaticRadar (subklasse av Radar)

Har lokale egenskapsvektorer for signal til støyforhold (SNR), signalstyrke fra mål og støysignal som funksjon av tidsskritt i en monostatisk radarkonfigurasjon. Det er ikke per i dag laget en tilsvarende klasse for bistatisk radar, men det kan implementeres ved behov.
Klassen har en rekke metoder:

A.6.1 function snr_levels(obj,target)

Beregner lokale klasseegenskaper som SNR, signalstyrke og støynivå.

A.6.2 function detect_mc(obj,sig_mat,sig_pos,par,propStr)

Avgjør om det er en deteksjon basert på tersklingsnivå og valgt metode: De ulike metodene som er implementert i denne versjonen er en ren terskling, ikke koherent cell-averaging CFAR, ikke koherent greatest-of CFAR og ikke koherent integrert ordered-statistical CFAR.

A.6.3 function detect(obj,target,propStr)

Beregner sannsynlighet for deteksjon når modelleringen bruker den analytiske metoden. Ulike metoder er implementert og kan velges ved parametersetting. De ulike metodene er:

- Swerling 0 fra Løvli (1994)
- Swerling 0, use approximate formula Mahafza&Elsherbeni eq. 2.27
- Swerling 1-4, beregn Pd "Direct evaluation of radar detection probabilities", X.Y. Hou, N. Morinaga, T. Namekawa, IEEE AES July 1987
- Swerling 1-4, beregn Pd vha "Radar Target Detection Handbook of Theory and Practice" by D.P. Meyer,H.A. Mayer 1973
- Shnidman metoder

A.7 `Classdef` Antenna

Har lokale egenskaper som beskriver polarisasjon, høyde, beam-vinkler, beam-pattern, gain og type antenne.

A.7.1 function readDefinitions(obj,fname)

Leser inn klasseegenskaper fra definert fil i prosjektet som kjøres og tilordner til instans av klassen.

A.7.2 function getRange(obj,ecr_target)

Beregner avstand fra radar antenne antatt samlokalisert med radar til objekt.

A.7.3 function angleDeviation(obj, pos_in_local, beam_ptr)

Beregner vinkelavvik mellom pekeretning og retning mot objekt.

A.8 `Classdef` ParabolicAntenna (subklasse av Antenna)

Har lokal egenskap som inneholder diameter til antenne.

A.9 `Classdef` CosecantAntenna (subklasse av Antenna)

A.9.1 function getGain(obj,theta, theta3)

Beregner gain for en cosecant antenne.

A.10 `Classdef` Environment

Har et sett av lokale egenskapsvariable som beskriver parametre for kall til APM og resultatveltorer fra APM samt referasevariable til klasseobjekter av typen LandClutter, SeaClutter, MapClass og Atmosphere.

A.10.1 function CalculateAPM(obj)

Kaller APM og tar vare på propageringsfaktorvektorene i klassevektorer. Beregner f2 for standard atmosfære og for angitte parametre.

A.10.2 function initializeAPMparameters(obj, Radar)

Initialiserer APM-parametre fra radar og antenne egenskapsverdier.

A.11 `Classdef` SeaClutter

Lokale klasseegenskaper for sjøclutterverdier både fra GIT og NRL modeller.

A.11.1 function calculate_GITmodel(obj, Radar, Vw, wdir)

Beregner tilbakespredning fra sjøen som funksjon av avstand (σ(r)) gitt frekvens, polarisasjon, antennehøyde, vindstyrke og vindretning basert på model basert på empiriske data. Modellen er utviklet ved Georgia Technical Institute.

A.11.2 function calculate_NRLmodel(obj, Radar, Vw)

Beregner tilbakespredning fra sjøen som funksjon av avstand (σ(r)) gitt frekvens, polarisasjon, antennehøyde, vindstyrke og vindretning basert på en modell utviklet ved Naval Research Laboratory (NRL). Modellen er utviklet for bedre å representere eksperimentelle data ved små betraktningsvinkler og har i tillegg verdier for sjøtilstand 0.

A.12 `Classdef` LandClutter

Klassen har lokale klasseegenskaper for tilbakespredning fra land. I tillegg har klassen private egenskaper for ulike terrengtyper til bruk i modellering av clutter ved lave betraktingsvinkler i modell av E.V.Tarnavsky, G.P.Kulemin [9].

A.12.1 function calculate_TKmodel(obj, Radar, distance_vec, terr_type_vec)

Beregner tilbakespredning fra terreng ved små betraktningsvinkler og angitt terrengtype, antennehøyde og frekvens.

A.13 `Classdef` MapClass

Klassen har lokale egenskapsvariable for å holde scenarioets kartutsnitt fra DTED avledede filer.

A.13.1 function loadMap(obj, mapVals)

Laster inn de respektive kartutsnittene fra 1x1 grads filer slik at lokalt kart dekker området radaren og flygende objekter beveger seg i.

A.14 `Classdef` Atmosphere

Denne klassen har foreløpig ikke innhold utover standard set og get metoder.

A.15 `Classdef` BeamSchedulerClass

Klassen har private egenskaper som lagrer sekvensen av beamer radaren vil gå igjennom samt parametre som PRF, skannetid og mode for å generere videre beamer. Det er kun en mode som så langt er implementert og det er en constantAzEl mode hvor elevasjon er konstant og radaren roterer med konstant azimut beam og rotasjonsrate.

A.15.1 function store_beam_sequence(obj,varargin)

Lagrer sekvensen av radarbeamer som skal traverseres som funksjon av tid for gitt mode. Nåværende versjon har kun en mode som har konstant rotasjon og fast elevasjonsvinkel. Hver lobeposisjon representerer enkeltpulsers pekeretning.

A.15.2 function get_beam(obj)

Returnerer parametre for neste beam.

A.16 `Classdef` OutputCl

Lokale egenskapsvektorer for lagring av en rekke verdier som funksjon av tid for bruk i plotting av resultater.

A.16.1 function plotAndReport(obj, sim, project_path, project)

Overordnet metode for styring av hvilke plot og rapporter som skal genereres. Parametre lest inn fra simuleringens prosjektfil styrer valg av plot. Hvis GenerateFigureFiles er valgt, lagres plotfiler på prosjektområdet som kjøres.

A.16.2 function plotPower_Range_dbm(obj, sim, save_path)

Plotter mottatt signalstyre, clutternivå og støy som funksjon av tid.

A.16.3 function plotSNR(obj, sim, save_path)

Plotter S/N og S(N+C) for ufiltrert og MTI-filtrert signal avhengig av valgt prosessering.

A.16.4 function PlotDetRes(obj, sim, save_path)

Plotter deteksjoner i avstand-tidsplot hvor deteksjoner tilhørende objekt og falske deteksjoner er angitt.

A.16.5 function GeneratePlotReport(obj, sim, save_path)

Denne metoden er foreløpig tom men er tiltenkt automatisert rapportgenerering fra kjøringer.

A.17 MainRadarModelling.m

Dette er startpunktet for modelleringen. Her leses først prosjektdefinisjoner og tilhørende parameterfiler inn. Klassestrukturen settes opp og initialiseres før kontrollen overføres til hovedløkka som løper som en metode i Radar (Radar.run_scenario_mc). Grunnen til at hovedløkka kjører i Radar er at det er radarens utlegg av lober som styrer simuleringens gang i denne versjonen. Ved retur plottes resultater fra Main.

A.18 Common rutiner

Det er definert en rekke rutiner som ikke er underlagt klasser som metoder. De viktigste av disse vil bli kort listet og forklart.

A.18.1 bincoef.m

Regner ut binomialkoeffisienter.

A.18.2 calcRefractionCorr.m

Refraksjonskorreksjon for radar data.

A.18.3 clutter_k.m

Beregner clutter skaleringskoeffisient gitt clutter formfaktor og radarparametere i følge likning (3.5).

A.18.4 clutter_shape.m

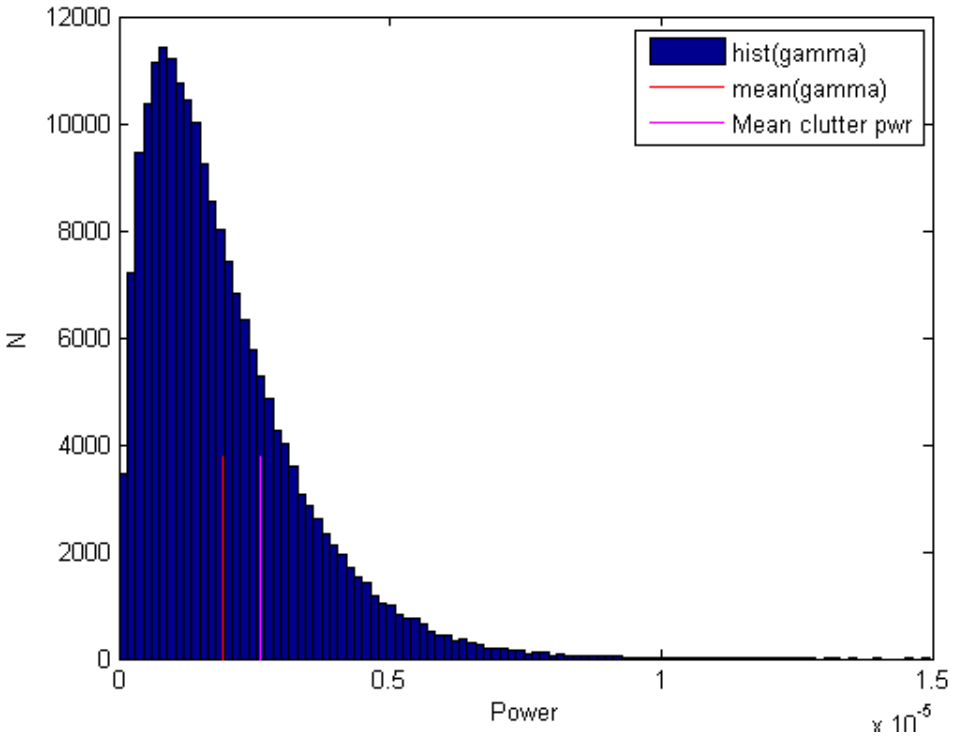
Modellerer og returnerer clutter formfaktor, enten fast verdi eller i forhold til Ward-Tough-Watts modell [2].

A.18.5 convert_k.m

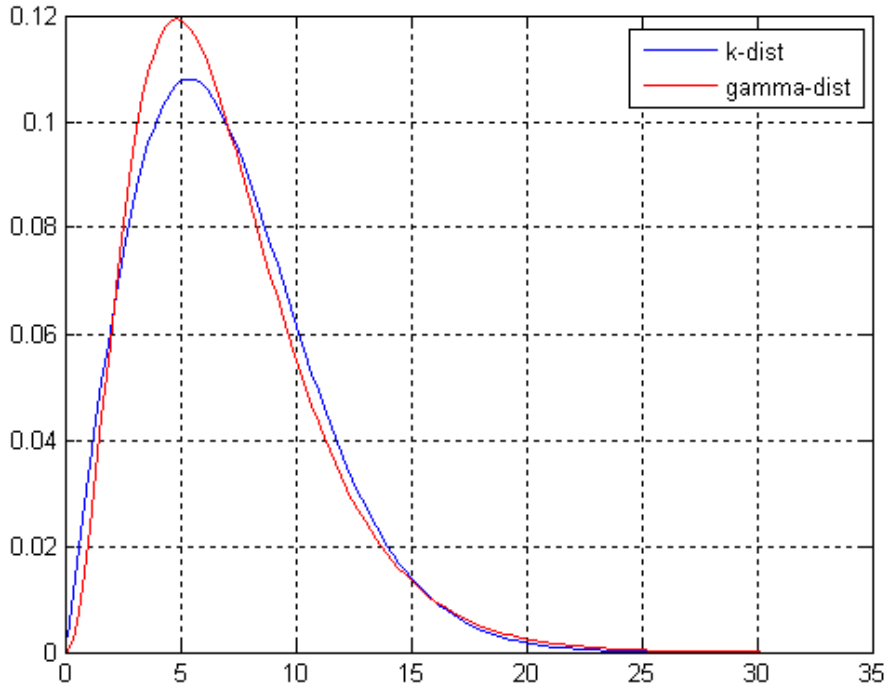
Approksimerer en K-forderling ved hjelp av ikke-sentrerte gammafordelinger for videre bruk i den analytiske simulator tilnærmingen angitt av Shnidman, [10].

A.18.6 convert_k_gam.m

Approksimerer en K-forderling ved hjelp av en gammafordeling.
Approksimasjonseksempel er vist i Figur 5.13.



Figur 5.13 Gammafordeling som en tilnærming til K-fordeling vist i forhold til middelverdien for fordelingen (mean(gamma)) og midlere clutter power som input til algoritmene. Det er en liten forskyvning av fordelingen mot lavere verdier men den er ikke spesielt stor.



Figur 5.14 Figuren viser tilpassning av gammafordeling til en k-fordeling.

Tilpassning av gammafordeling til k-fordeling er vist i Figur 5.14.

A.18.7 coord_transf_FFImod.m

Koordinat transformasjon fra et NED koordinatsystem til et annet.

A.18.8 ecef2lla.m

Koordinattransformasjon.

A.18.9 kpd.m

K-fordelingsfunksjon.

A.18.10 lla2ecef.m

Koordinattransformasjon.

A.18.11 meyer.m

Eksakt beregning av sannsynlighet for deteksjon for bruk i den analytiske kjøremodusen.

A.18.12 mti_filt_func.m

Filtrer en K-fordeling i forhold til angitt MTI filter og returnerer parametere for en ny type fordeling som kan inngå i Shnidmans formler. Brukes kun i den analytiske tilnærmingen.

A.18.13 mti_filt_signal.m

Filtrerer et signal i forhold til angitt MTI filter.

A.18.14 mti_num_filtresp.m

Plotter ut frekvensrespons for den angitte PRF sekvens og MTI filter typen.

A.18.15 pd.m

Bergning av eksakt deteksjonssannsynlighet i et clutter fritt miljø.

A.18.16 r2bn.m / rdcc2cc.m / re2t.m / rn2b.m / rt2e.m

Diverse rotasjonsfunksjoner for konvertering av koordinater.

A.18.17 shnidman.m

Implementasjon av diverse Shnidman formler for eksakt beregning av deteksjonssannsynlighet.

A.18.18 snr.m

Returnerer signal-til-støy forholdet med utgangspunkt i radarlikningen.

A.18.19 treshold.m

Beregner deteksjonsterskel gitt feildeteksjonsrate for clutterfritt tilfelle.

A.18.20 sw_random.m

Returnerer stokastiske tall trukket fra gitt Swerling fordeling.

Referanser

Bibliography

[1] I. Antipov, "Simulation of Sea Clutter Returns," Salisbury, Australia,DSTO-TR-0679, 1998.

[2] K. D. Ward, R. J. A. Tough, and S. Watts, *Sea Clutter: Scattering, the K distribution and Radar Performance*. London, UK: IET, 2006.

[3] I. Antipov, "Statistical Analysis of Northern Australian Coastline Sea Clutter Data," Edinburgh, Australia,DSTO-TR-1236, Nov.2001.

[4] T. Hair, T. Lee, and C. J. Baker, "Statistical properties of multifrequency high-range-resolution sea reflections," *IEE Proceedings-F*, vol. 138, no. 2, pp. 75-79, Apr.1991.

[5] M. M. Horst, F. B. Dyer, and M. T. Tuley, "Radar sea clutter model," 1978, pp. 6-10.

[6] V. Gregers-Hansen and R. Mital, "An empirical sea clutter model for low grazing angles," 2009.

[7] F. E. Nathanson, *Radar Design Principles, 2.ed*. New York: McGraw-Hill, 1991.

[8] H. Rohling, "CFAR Thresholding in Clutter and Multiple Target Situations," *IEEE Transactions on Aerospace and ElectronicSystems*, vol. 19, no. 4, pp. 608-621, July1983.

[9] E. V. Tarnavsky and G. P. Kulemin, "Modeling of radar land clutter map for small grazing angles,", 5484 ed. Romaniuk and S. Ryszard, Eds. 2004, pp. 702-706.

[10] D. A. Shnidman, "Radar Detection in Clutter," *IEEE Transactions on Aerospace and ElectronicSystems*, vol. 41, no. 3, pp. 1056-1067, July2005.