CW en SSB

tekstversie 06aug2017 PA0WV

Inleiding

Je hoort in het algemeen dat CW meer verbindingen mogelijk maakt dan met SSB mogelijk zijn onder marginale condities.

Dat verbaast natuurlijk niet en is volstrekt geen wonder, want SSB is een spraakverbinding met tenminste 2,5 kHz bandbreedte, die bij DX bovendien nog een keer nagenoeg onverstaanbare accenten en taalmoeilijkheden kan opleveren, terwijl CW slechts een aan/uit geschakeld toontje is met een smalle bandbreedte van 150 hertz of minder (ruim 4 maal seinsnelheid in woorden per minuut is de bandbreedte in Hz) en tevens een vast vocabulaire heeft van pakweg 200 afkortingen waarmee conversatie al mogelijk is, op het niveau van een bevolking met een taal die 200 gebruikte woorden omvat, zoals Bonobos. Kortom de gemiddelde zendamateur..

Dit artikel is bestemd voor echte zendamateurs, die zich willen ontplooien door te experimenteren, te onderzoeken, zelf te bouwen en aldus tastend en denkend kennis proberen op te doen. Ik beschrijf als voorbeeld mijn moeizame stappen op die weg, ter leeringhe ende vermaeck.

Dit is een multimedia-artikel, dat wil zeggen dat u met uw PC via Internet geluidsbestanden kunt beluisteren die bij dit artikel behoren en zijn te vinden op

http://pa0wv.home.xs4all.nl/wav/ achter de laatste breukstreep moet u dan de opgegeven bestandsnaam invullen. Hoofd– en kleine letters maakt verschil.

Probleemstelling

Ik wil door experimenteren onderzoeken wat nu in dB en dus in zendvermogen, het verschil is tussen CW en SSB als je beide nog net zonder QRM in de ruis kunt opnemen.

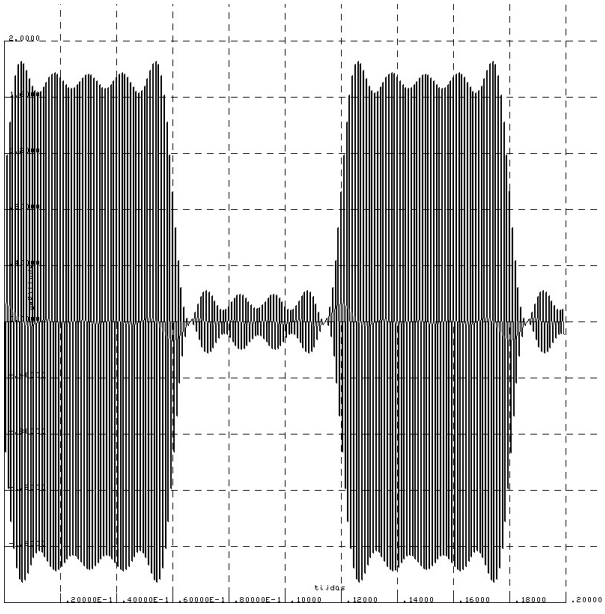
Elk S-punt meer is 6 dB en dat is 4 maal vereist antennezendvermogen.

Te onderzoeken uitgangspunt

Een ruwe berekening wil ik verifiëren. Die berekening luidt als volgt:

Bandbreedte van SSB is 300 - 2800 Hz. Daarmee is spraak, dus pakweg 140 woorden per minuut (wpm) mogelijk, zoals: "Alfa bravo charlie delta echo foxtrot" in plaats van in Morse: abcdef .

CW vereist bij 20 wpm - dat is een datasnelheid iets lager dan 17 bits/s - dan hooguit 150 Hz bandbreedte op de band, dat is al zeer ruim geschat, omdat je dan van een blokvormige aan/uit modulatie van 8,3 Hz, zijnde de halve datasnelheid, t/m de negende harmonische van de modulerende frequentie in AM modulatie van de draaggolf meeneemt. De vuistformule die luidt: ruim 4 maal de seinsnelheid in wpm is de bandbreedte in Hz, gaat kennelijk tot en met de vijfde harmonische als vereiste bandbreedte. 150 Hz bandbreedte is dus voldoende voor ruim 35 wpm. Tot en met de negende harmonische meenemen bij 20 wpm betekent een overgang van 0 naar volle zendvermogen in 8 ms, terwijl een morsepunt (dit), evenals de spatie tussen

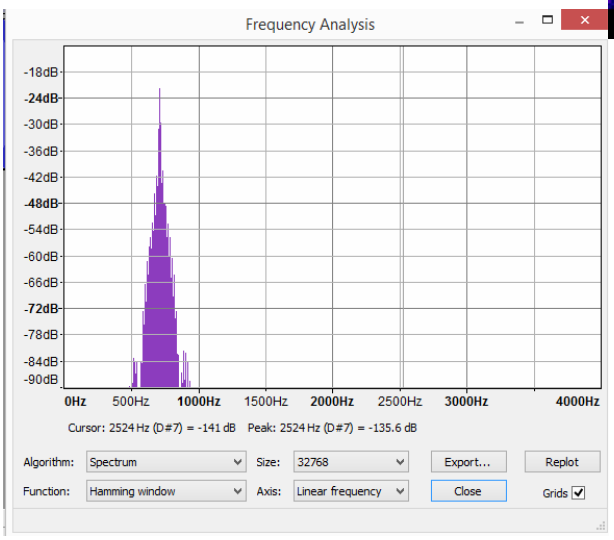


de punten, dan 60 ms duurt gemeten op de halve zendamplitude. Bovenstaande figuur geeft een dergelijk HF Morsesignaal bestaande uit een reeks punten (dits) in grafiek weer.

Aan die flanksteilheid van de omhullende moet je dan dus voldoen; trager mag korter niet, om binnen de bandbreedte te blijven. Dan heb je een mooi schoon signaal zonder sleutelklik. Een bijna aldus gevormd geluidsbestand dat in Morsecode de letters abcdef met een snelheid van 20 wpm bevat met 712 Hz toonhoogte kunt u horen op de eerder in de inleiding genoemde link door abcdef.wav aan te klikken.

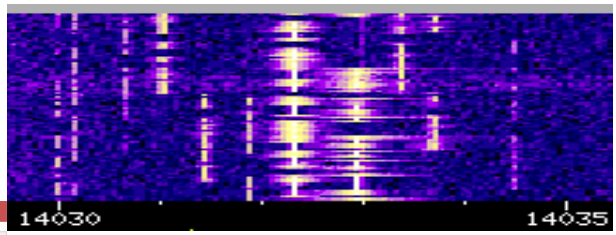
De stijg- en daaltijden van dit signaal zijn cosine squared dat wil zeggen dat de flanken van de omhullende verlopen zoals een sinus van minimum naar maximum of omgekeerd verloopt, met in dit voorbeeld een duur van 10 ms.

Het spectrum van dit signaal, bepaald door de fast Fourier transform die in het op Internet te vinden freeware programma Audacity zit ingebouwd, met 16 bits samples van de wavfile abcdef.wav, 8000 samples/s mono is met smoothing window type Hamming hieronder afgedrukt.



Dat levert dus een eerste voordeel qua bandbreedte en dus signaalruisverhouding van 2500 Hz/150 Hz in vermogen als je een CW-filter gebruikt, oftewel 12,2 dB , omdat het ruisvermogen evenredig met de bandbreedte toeneemt. Het signaalvermogen wordt immers onverzwakt doorgelaten door het filter en van de ruisvermogen dat over 2,5 kHz evenredig verspreid is, wordt alles buiten de smalle doorlaatband van het filter niet doorgelaten

Een jappenbak heeft CW als optie, soort ondergeschoven kindje c.q. koekoeksjong, waardoor je jezelf kunt onderscheiden van een brekiebrekietokkelCeeBeejer, die geen CW optie kent. Men werkt echter in de jappenbak met SSB als standaard, dus de eindtrap is een linear oftewel instelling klasse AB, en die heeft een rendement van minder dan 50%, of je nu CW werkt of niet dat blijft zo terwijl dat voor CW helemaal niet nodig is. Helpen doet het trouwens niet, want als ik naar een plaatje kijk van een waterfall display (verticaal is de tijdas en horizontaal de frequentie-as) op een contest weekend, sorry tautologie, elk willekeurig weekeinde, dan zie ik als spectrum van de jappenbakken in de CW stand:



Je ziet duidelijk dat een paar van die koopbakken schandalig brede spectra maken met CW en dat dit absoluut onnodig is, blijkt uit andere CW-signalen in het plaatje.

Een zender die uitsluitend voor CW of FM ontworpen is heeft echter een eindtrap in klasse C; van de sturing wordt slechts het signaaldeel boven niveau 0,5 van de topwaarde gebruikt als de top van de sinus 1 is. De openingshoek is dan slechts 120 graden van de 360 graden van een sturende sinus. Dat is dus geen linear maar een non-linear eindtrap. Het rendement van zo'n eindtrap is dan 80%. Oftewel ten opzichte van SSB , met minder dan 50% rendement 2 dB extra.

Totaal zitten we nu er als tussenresultaat op,

dat CW al 14,2 dB beter is dan SSB, bij dezelfde input van de zendereindtrap, mits je een CW filter gebruikt bij de ontvanger, en in CW niet gaat uitspellen "Alfa brave Charlie", wat je ruim een factor 7 in snelheid scheelt, zodat er van die spraaksnelheid van 140 wpm een gebruikelijke CW snelheid van 20 wpm overblijft; en de eindtrap, die uiteraard zelfbouw is bij de bekende definitie van "echte" zendamateurs, in klasse C staat. Goed voor de *meleu-hype* trouwens. Deze verhouding tussen spellen met spraak of zenden met Morse blijkt trouwens ook uit twee bestanden kanweg.wav en cw.wav te beluisteren op de genoemde URL die respectievelijk het navoalfabet nagenoeg spatieloos uitspreekt in krap 17,7 seconde en (omdat het 290 dits zijn) in dezelfde tijd het in morse seint met 19,7 wpm. Bij 19,7 wpm speel je dus quitte. Logisch ook want het seinen van dahdit duurt beslist korter dan het zeggen van "november".

Een geluidsbestand met alfa t/m zoeloe duurt dan dus even lang als een morseuitzending met a t/m z. Ik ga er hierbij voor de duidelijkheid van uit, dat je aan de grens zit wat signaal ruisverhouding betreft en je dus je over te brengen bericht moet spellen in SSB.

PEP, piekvermogen en gemiddeld vermogen

Gemiddeld vermogen wordt bij een sinus bepaald door de effectieve waarde van de spanning te nemen, dus de topwaarde van de spanning gedeeld door wortel 2. Bij een bemonsterd bestand, zoals een wav file kun je die bepalen door het kwadraat van elke monster te nemen en te vermenigvuldigen met de tijd tussen twee monsters, dat geeft voor de hele file opgeteld de energie die geleverd zou worden in 1 ohm als je voor de monstergrootte de eenheid volt neemt, en door te delen door de speelduur van het bestand heb je dan het gemiddelde vermogen Pg te pakken, dat is namelijk de gemiddelde geleverde energie per seconde.

Het piekvermogen van een sinus is het vermogen dat geleverd wordt op de piek van een sinus, en bij een bemonsterd bestand dus het kwadraat van de monsterwaarde maal de tijdsduur tot het volgende monster, dat is de energie en het piekvermogen volgt dan door net te doen of dat een seconde lang aanwezig zou zijn, met als resultaat dat het piekvermogen uitkomt op het kwadraat van de piekspanning. Voor de volledigheid: een en ander als een belasting van 1 ohm wordt aangehouden. In mijn betoog speelt die fictieve belasting geen rol, omdat het om vermogensverhoudingen gaat.

Een SSB zender kan lineair worden uitgestuurd tot de PEP, dan wordt een HF sinus geleverd met de piekwaarde als amplitude, die sinus levert dan PEP vermogen, dat is de helft van het zojuist beschreven piekvermogen. Bij CW gebeurt dat bij key down. Heb je een sterk gepiekt audiosignaal dan is er geen sprake van een sinus, de piek treedt in het audiogebied even op en de zender levert dan net zo lang een HF sinus met de maximale amplitude die de zender (legaal) kan leveren, dezelfde amplitude als bij CW en het vermogen dat die HF sinus dan levert is de PEP.

Om CW met SSB te vergelijken houden we de PEP waarden gelijk (volle zenderuitsturing in beide gevallen dus) en we kijken bij welke gemiddeld toegevoegde ruisvermogen over de gebruikte signaalspectrumbreedte het signaal nog net neembaar is.

Omdat we bij CW het ruisvermogen beperken tot een bandbreedte van 150 Hz moet ter eerlijke vergelijking de 150 Hz ruisvermogen worden vermenigvuldigd met de factor 2500 Hz/150 Hz. Dat lijkt wellicht eerlijk maar dat is het weer niet, dat lijkt dus wel landspolitiek; en dat komt omdat ons gehoor en brein tezamen in staat zijn ruis van andere toonhoogte dan de CW, uit te sluiten. We hebben dus een CW filter in de hersenen zitten. Kortom we zijn evolutionair bestemd om CW als voorkeursmode te gebruiken, dus: ***"Handen af van de CW banden"***.

Nu heb ik een jaar of wat terug een spraakopname gemaakt met een PC, die achterel-

kaar nagenoeg zonder adempauze het hele NAVO spellingalfabet opleutert. Daarvan heb ik destijds met een rekenprogramma de PEP en het gemiddelde vermogen Pg bepaald. Door de piekerige vorm van het spraaksignaal waarbij de pieken de zender net vol uitsturen is voor de PEP de effectieve waarde van het signaal gedurende de piekspanning gebruikt. Gedurende die audiopiekwaarde levert de zender immers meerdere HF sinusperioden met de maximale amplitude, en dus de PEP. Die verhouding bleek Pg/PEP = 4% te zijn. Dat is uit het opgenomen geluidsbestand te bepalen door de piekamplitude en het gemiddelde vermogen van het bestand te berekenen.

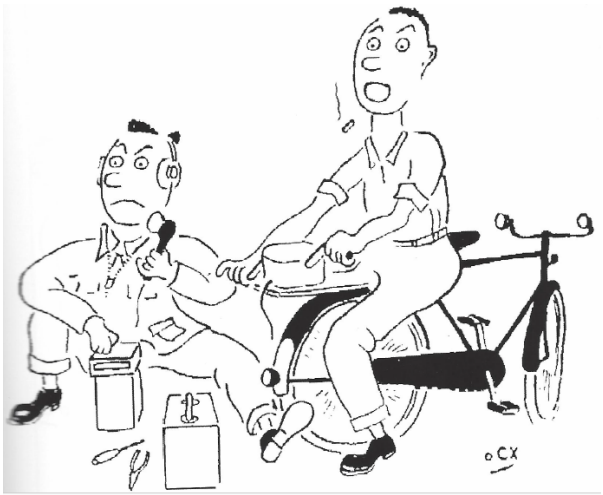
Dat bedrag Pg/PEP vond ik onwaarschijnlijk laag, maar het geschreven programma is toen uitgebreid getest met een proefbestand van een minuut durende halve sinus, zodat je het geleverde resultaat ook analytisch kunt bepalen, en ook nog door het in korte stukjes van 0,8 seconde opdelen van de gemaakte spraakfile en de korte stukjes apart Pg/PEP te beoordelen teneinde een of andere stoorpuls die de PEP zou verhogen te kunnen elimineren. Hielp niet. En achteraf was dat ook zo gek niet omdat 4% vermogen 20% van de spanning is. En dat is wat je traditioneel op een scopebeeld, dat de spanning laat zien, van de omhullende van een spraaksignaal ongeveer ziet als verhouding van de geschatte gemiddelde waarde en de pieken van het spraaksignaal.

Bij CW kun je tijdens een uitzending de PEP ten opzichte van het gemiddelde zendvermogen berekenen als zijnde 44%, omdat je tijdens een CW uitzending 44% van de tijd de sleutel indrukt en dan het volle zendvermogen zendt. Immers het representatieve standaardwoord PARIS is 50 dits lang, waarvan 22 dits key-down.

Inmiddels is na jaren dat geluidsbestand niet eenvoudig terug te vinden en heb ik een nieuw aangemaakt voor de huidige proefnemingen. Op de eerder genoemde Internet-link is dat te vinden als navo8000.wav en met ingekorte woordspaties als kanweg.wav.

Goed, we sturen de lineaire eindtrap zowel bij SSB als CW uit tot maximale zendvermogen en verliezen dan t.o.v. CW (44%)/(4%), dat is in dB: 10,4 dB zodat het verschil CW SSB alles tezamen uitkomt op 24,6 dB oftewel ruim een factor 288 in inputvermogen op de antenne.

Geen wonder dat met eenzelfde antenne een QRP-er met minder dan 4 watt input in zijn simpele zelfbouwzendertje werkend op een batterij of kleine accu, bijgeladen met een zonnecel, of de xyl trappend met een fietsdynamo op het wiel, met CW meer kan bereiken dan de trotse eigenaar van, en leuterend in, zijn jappenbak met 1 kW PEP input.



Doorgaans is trouwens meer te bereiken en niet hetzelfde omdat ook de kans op QRM evenredig met de gebruikte bandbreedte toeneemt.

Voor de vergelijking zorg ik er voor dat de PEP van CW en de PEP van SSB per spellingwoord bij het toevoegen aan de ruis dezelfde waarde hebben, dat hadden ze immers aan de zendantenne ook. Voor een goede vergelijking zal ik bij SSB en CW dezelfde ruis toevoegen over een bandbreedte van 300 tot 2800 Hz, en vervolgens bij CW het smalbandige 150 Hz filter daarachter plaatsen. Dan impliceer je immers de wonderlijke eigenschap van je grijze hersenmassa die in staat is naar een toon te speuren en andere frequenties uit te sluiten en dat effect is dan ook te

bepalen uit de ontvangstresultaten.

De ontvangstpraktijk

Ik wil nu weten hoe lang een CW signaal neembaar is, vergeleken met een SSB signaal in de ruis. Ook als SSB gecomprimeerde audiosignalen bevat, na compressie beperkt tot 2,5 kHz audio bandbreedte. Mijn bovenstaande betoog was immers voor een lineaire SSB zender gedaan om Pg/PEP te bepalen. Pg is het gemiddelde vermogen. Die Pg/PEP is hoogfrequent dan precies hetzelfde als die van de audio uit de microfoon tussen 300 en 2800 Hz, omdat (USB) SSB genereren zonder audiocompressie slechts het verschuiven van het audiospectrum naar de zendfrequentie is. Is er wel audiocompressie dan is weer Pg/PEP van de audio dezelfde als aan de antenne kan worden gemeten vanwege de lineariteit.

Aanvankelijke opzet

Daartoe was het aanvankelijk vanwege een andere opzet nodig dat ik een ruissignaal genereer met een bandbreedte van 300 tot 2800 Hz en een ruissignaal van 150 Hz breed, zo'n beetje de bandbreedte van commercieel beschikbare CW filters. Daar wordt dan een Morsesignaal centraal ingezet van 12 wpm, zodat elke klassieke zendamateur dat op het gehoor behoort te kunnen opnemen, en vervolgens wordt gekeken door diverse wav bestanden bij verschillende signaalruisverhoudingen, te publiceren op eham.net en andere websites van CW clubs, zoals sideswipernet.org, lcwo.net en de CW-archives mailinglist, waar een ploeg ervaren telegrafisten te vinden is, welk van de bestanden die elk 4 groepen van 5 random niet gepubliceerde karakters bevatten, ze nog foutloos op het gehoor kunnen opnemen.

Ruis

Van zogenaamde bandbegrensde witte ruis is voor ons amateurs bekend dat het tussen de twee bandgrenzen per Hz bandbreedte evenveel ruisenergie bevat. Dat wil dus zeggen dat je met een spectrum-analyser over die bandbreedte een vlak ruisspectrum, oftewel een horizontaal grasveldje vindt. De spectrumanalyser-beelden van je buurman zijn overigens altijd groener dan die van je eigen analyser. Binnen die bandbreedte zijn alle spectrumcomponenten dus even sterk. Een spectrumcomponent is een sinus van constante amplitude en fase, zoals het een sinus betaamt. Bekijken we op een bandbreedte van 1 Hz de ruis dan is bij benadering die sinus bijna constant want elke variatie of jitter in amplitude en fase is een vorm van amplitude- en fasemodulatie en die bevat dus bandbreedte in zijbanden en die is er bijna niet binnen 1 Hz.

De volgende stap is dus de gedachte dat je bandbegrensde witte ruis kunt maken tussen frequentie fa en fb door om de hertz een sinus te genereren met constante aan de anderen gelijke amplitude en random gekozen constante fase voor elke frequente tussen en inclusief de bandgrensfrequenties fa en fb.

Geluidsbestanden voor een PC met suffix wav bestaan uit PCM (Puls CodeModulatie) monsters, die ga ik gebruiken, aanvankelijk en ook uiteindelijk met 8000 monsters per seconde. Geen MP3 bestanden dus, die weliswaar veel compacter zijn, maar die niet geschikt zijn om door ruis gemaskeerde signalen getrouw te reproduceren. Voor muziek wordt doorgaans 44100 monsters per seconde stereo gebruikt zodat je tot 22 kHz kunt weergeven, maar bij spraak en CW volstaan 8000 monsters/s mono ruimschoots, omdat je daarmee tot 4000 Hz audio mono kunt weergeven. Dat scheelt ruim een factor 10 in grootte van het bestand en dus ook bestandsruimte op je PC en transporttijd via Internet en last but not least, rekentijd om de bestanden door berekenen te creëren.

Als je met 16 bit PCM werkt in een audio.wav bestand mag je dus maximaal tot 2^15 -1 = 32767 gaan als amplitude van de monsters, want dan kun je alle waarden code-

ren van 32767 tot -32768; en omdat te kunnen garanderen voor de som van alle fb-fa+1 spectrumcomponenten tussen de bandbreedte fa tot fb moet de amplitude van al die sinussen (2^15-1)/(fb-fa+1) zijn. De kans dat al die sinussen gelijktijdig hun maximum bereiken is zeer gering maar niet nul. Daarom wordt een gegenereerd bestand eerst met cijfers achter de komma opgeslagen, daarvan wordt dan de maximale amplitude bepaald en vervolgens wordt het bestand opnieuw gelezen en met de berekende versterking vermenigvuldigd zodat de gewenste piekwaarde wordt bereikt en daarna worden de samples, eventueel opgeteld met signaal, pas afgerond tot gehele getallen en in het .wav bestand opgeslagen als 16 bits gehele getallen.

Je kunt ook eenmalig het geluidsbestand genereren en achteraf de monsters vermenigvuldigen met de versterking, maar dat heb ik niet gedaan omdat ik dan de kwantiseringsruis onnodig vergroot. Immers monsters (samples) met grootte 1, 2 of 3 worden bij versterking met een factor tien dan 10, 20 of 30 en de tussenliggende waarden komen niet meer voor. Dat geeft een onnodige verhoging van de kwantiseringsruis, in dit voorbeeld met 20 dB.

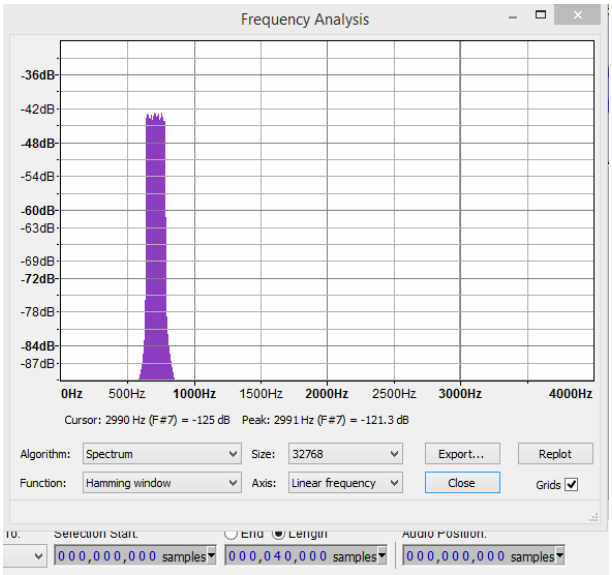
Dus de berekende monsters worden als drijvende komma getallen (type double) eerst versterkt en daarna pas gekwantiseerd tot de montergrootte die een geheel 16 bits getal is.

Op die wijze gegenereerde ruis heeft ook een verhouding Pg/PEP. die we eigenlijk niet interessant vinden, omdat we voor de signaal-ruisverhouding naar het gemiddelde ruisvermogen Pgr kijken dat is de berekende ruisenergie van alle ruismonsters van het wav bestand per seconde. Maar we willen oversturing bij monstergewijze optellen met een signaalbestand uitsluiten, vandaar die tweetraps berekening.

Een geluidsmonster van een bandbeperkte witte ruisband tussen 637 en 787 Hz, dus 150 Hz breed, die op deze manier geconstrueerd is kunt u horen op http://pa0wv.home.xs4all.nl/wav/712_150.wav

Dat bestand heeft dus een centraalfrequentie van 712 Hz.

Het spectrum is op dezelfde wijze (32768 samples, Hamming window), bepaald met Audacity en te zien op onderstaande figuur.



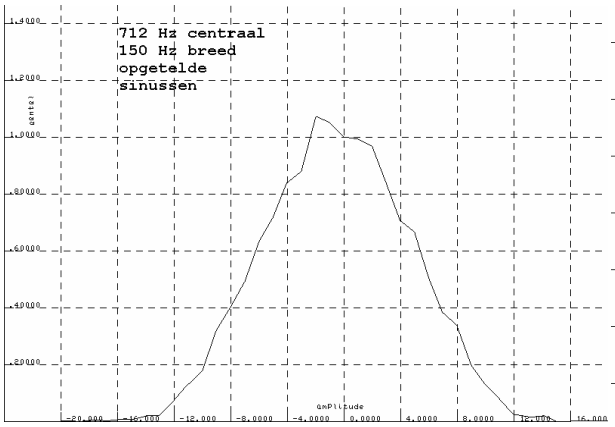
Bij het afspelen van het 150 Hz brede ruissignaal 712_150.wav valt periodiciteit in het geluid op, van naar schatting 1 seconde periodeduur. Dat is wel logisch, want na 1 seconde zijn alle 151 sinuscomponenten weer op precies hetzelfde punt beland. Dat gebeurt dus omdat een spatie der sinuscomponenten van 1 Hz is aangehouden. Houd je 1/10 Hz spatie aan tussen de sinuscomponenten, dan duurt het 10 seconde, maar dat kost wel 10 maal zoveel rekentijd om op die manier een ruisbestand te maken van dezelfde tijdsduur. Dus voorlopig ga ik door met 1 Hz spatie en als alles werkt kan ik dat verlengen en gedurende een nacht of zo als ik op een oor lig, de PC het ruis wav bestand voor de SSB bandbreedte laten berekenen. Het aantal sinuscomponenten per hertz bandbreedte wordt daarom als parameter in het programma opgenomen.

De gewenste bandbreedte van de ruis is met deze werkwijze gegarandeerd. Het signaal klinkt ook als smalbandige ruis (712_150.wav) en om de ruiserigheid van de

ruis te controleren heb ik van elk monster van het wav bestand de amplitude in een van 45 amplitudevakjes gesorteerd en in grafiek gezet.

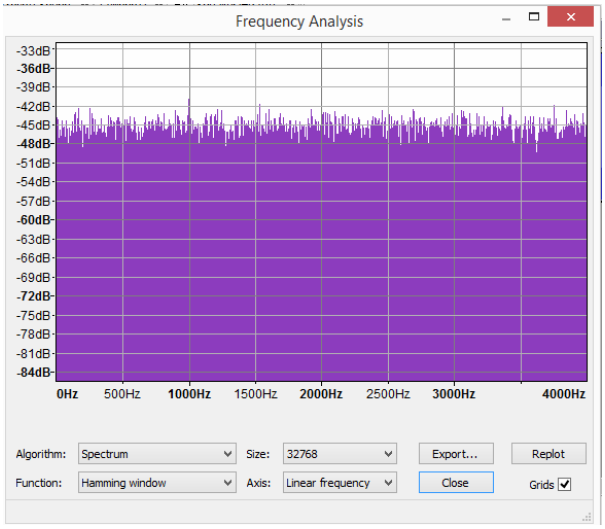
Vertikaal het aantal monsters in een van de 45 amplitudegebiedjes en horizontaal de grootte ervan als unsigned integer. Omdat bij een dubbele lengte van het bestand de grafiek in hoogte zou groeien en bij andere grafieken onderlinge vergelijking van de eigenschappen dan lastig wordt, is bij dergelijke grafieken elke y-aswaarde gedeeld door het aantal van de waarden rond amplitude 0. De grafiek heeft dan in principe als maximum 1, omdat de kleinste amplitudes het vaakst voorkomen, en is dan onafhankelijk van de lengte van het wav bestand. De deviatie van de gemeten monsters is dan goed vergelijkbaar.

Van de monsters van 712_150.wav ontstaan door sinussen met gelijke amplitude en per stuk random gekozen startfase en 1 Hz gespatieerd op te tellen, vindt u de grafiek van de pdf (probability density function) hieronder.

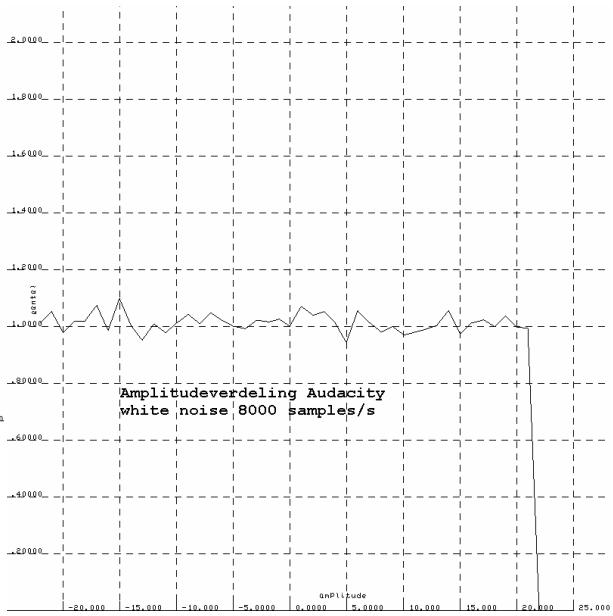


Vreemde vooralsnog onbegrepen problemen doemen op

Geef ik het open source programma Audacity opdracht om witte ruis te genereren, (bestand audwhite.wav bij de wav bestanden op de website) en kijk ik dan naar de grootte van de monsters dan komt elke amplitude nagenoeg even vaak voor. Analyse van een dergelijk bestand levert de daaruit bepaalde pdf hieronder. Het spectrum is vlak tussen 0 Hz en de maximale frequentie f met f=fs/2 waar-



bij fs de samplefrequentie is., Omdat mijn lang geleden geschreven grafiekenprogramma automatische scaling heeft in e 1-2-5, heb ik een kunstmatig punt op 0 helemaal rechts toegevoegd, waardoor op de daardoor geforceerde y-asschaal blijkt dat het een vrijwel vlakke amplitudeverdeling is, elke monsteramplitude komt dus nagenoeg even vaak voor.



Maak ik zelf witte ruis door alle monsters een random getal toe te kennen, dan gebeurt dat uiteraard ook, het klinkt prima, luister maar op witrnd.wav en het heeft eveneens een

vlakke pdfunctie.

Echter die snel aan te maken ruis heeft een vaste bandbreedte die bepaald wordt door het reconstructiefilter in de PC die van de monsters weer analoge audio maakt, dus bij de gebruikte 8000 monsters per seconde een spectrum dat ligt tussen 0 en 4000 Hz. Die kan ik dus niet gebruiken, omdat ik 300-2800 Hz en een 150 Hz brede ruisband rond 712 Hz nodig heb.

Het vreemde is dus dat witte ruis 0-4 kHz die een vlak spectrum heeft, alle amplitudes even vaak voorkomen en met de beschreven sinus-optelmethode is dat niet het geval.

Dat geeft dus te denken. Ik vermoed dat de witte ruis, omdat die de halve bemonsterfrequentie qua bandbreedte omvat, en dus per naburig monster de volledige seinspan kan verschillen, dat dan de Gauss amplitudeverdeling overgaat in een verdeling waarbij elke monsteramplitude even vaak voorkomt.

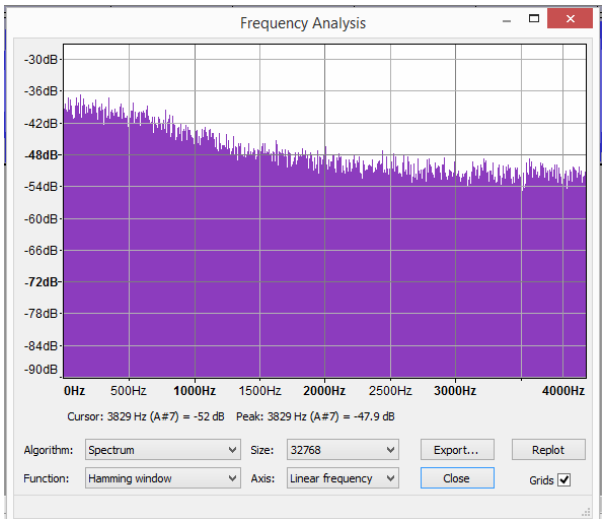
Nog een proefje in een poging het inzicht te vermeerderen: Ik interpoleer steeds lineair 3 monsters tussen 2 met random getallen gekozen monsters. Dat zal dan wel uitdraaien op bandbeperkte ruis tot 1000 Hz bij 8000 samples per seconde. (bestand Nis4.wav) denk ik dan intuïtief. Is niet zo. Het klinkt on (fris_en_fruitig) . Toch vreemd, want je gaat nu nooit in een klap van max naar minimum in 2 aan elkaar grenzende monsters, dat duurt nu minimaal 4 monsters. Het resulterende spectrum (hiernaast) is niet meer vlak en loopt toch door tot 4 kHz waar het pas 20 dB down is; en de pdf hoeven we niet eens te bepalen want die zal zonder enige mogelijke twijfel dezelfde vlakke verdeling hebben. Toch getest, want je weet maar nooit; en dat blijkt inderdaad het geval.

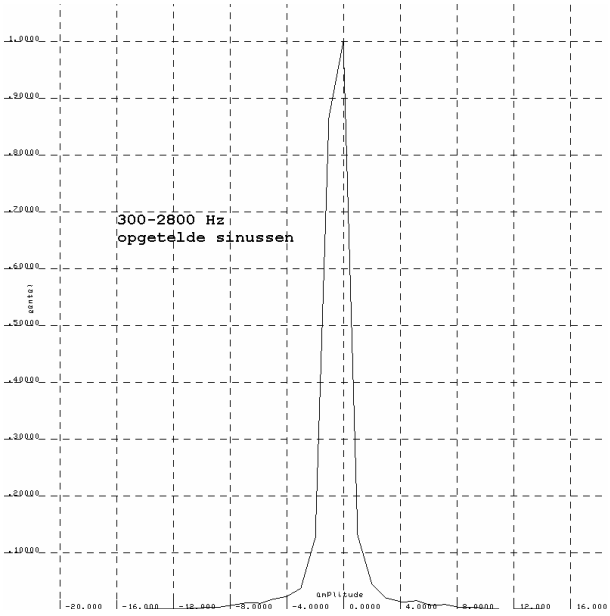
Maar ja, bij nader inzien, als je lineair interpoleert, maak je zaagtanden en die hebben harmonischen, dat kan het 20 dB down doorlopen tot 4 kHz verklaren.

Maar er is toch wel licht aan het einde van de tunnel. Als je een dobbelsteen gooit heb je evenveel kans om elk getal van 1 t/m 6 te gooien. Gooi je met twee stenen dan is het maximum van de som wat je kunt gooien 12 en en minimum 2, en dat zal beide gemiddeld een op de 36 worpen gebeuren. De kans echter dat je 6 gooit met twee dobbelstenen is vijf keer groter omdat dat gebeurt bij de gevallen (1,5) (2,4) (3,3) (4,2) en (5,1) Dat effect staat bekend als het central limit theorem.

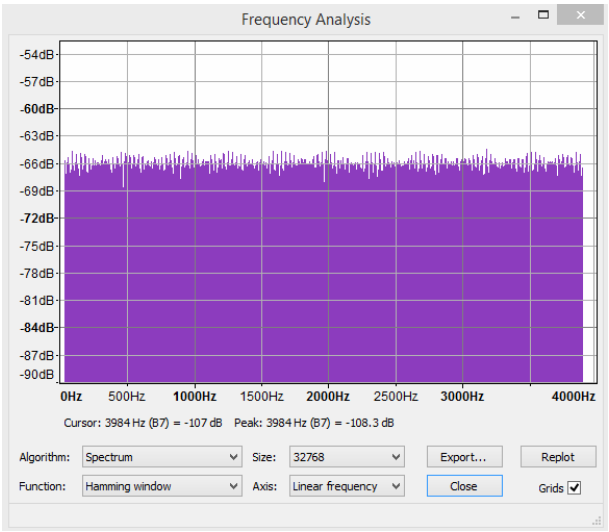
Spectrum

Het spectrum is te testen als ik als bandgrenzen van 0 tot de halve bemonsterfrequentie bandbegrensde witte ruis genereer met de sinus-optelmethode en dan naar de amplitude-





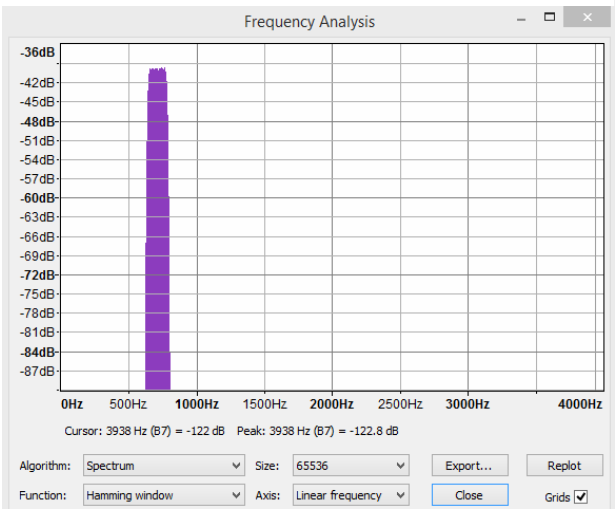
verdeling kijk. Om de test in een redelijke tijd te kunnen doen beperk ik de bemonsterfrequentie weer tot 8000 en de frequentie van de sinussen dus tot 4000 Hz, ik neem 3900 Hz om de goden niet te verzoeken. Eerst 1 en



later met 5 componenten per Hz,

Niet alleen heeft dat signaal een naar en raar spetterig geluid, en is de Pg/PEP aanzienlijk geringer, het klinkt ook verre van gezonde ruis. Het genererende programma heb ik op allerlei manieren die ik kon verzinnen getest (bijvoorbeeld 1 Hz brede ruisband en slechts 2 sinuscomponenten dus) maar ik kan geen fout ontdekken. Het spectrum blijkt er ook perfect uit te zien, het is vlak zoals te verwachten en begrensd binnen de geprogrammeerde bandgrenzen.

Ik heb ook ruis uit de luidspreker van mijn



ontvanger opgenomen en die voldoet ook aan een normaalverdeling, een klokvorm dus, van de monsteramplitudes.

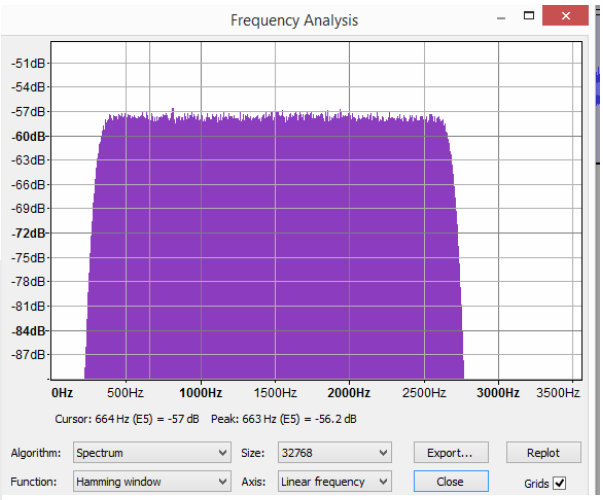
Wellicht is daarom nu een betere methode gewenst om ruis te genereren met de gewenste bandgrenzen.

Bandbegrensde ruis genereren

Twee methoden:

1. Ruis met de gewenste bandbreedte uit een ontvanger opnemen

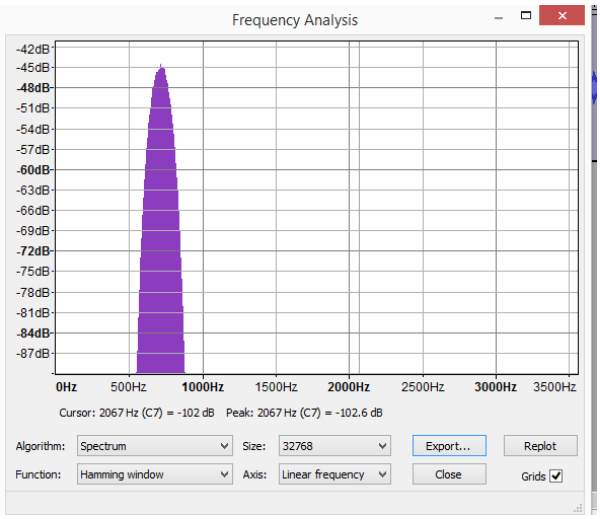
2. Met een in C geprogrammeerd digitaal bandfilter de gewenste ruisband uitfilteren uit een witte ruisband 0-4 kHz, gemaakt met random getallen, en die ruisbestanden ter gebruik opslaan om CW en SSB op te superponeren of beter nog ook de SSB en de CW door dat filter te sturen, zodat eventuele componenten die buiten de band vallen worden weggesneden. Dat is bij SSB testen van belang als het audiosignaal gecomprimeerd wordt om de verhouding PEP/Pg te verbeteren en dus het gemiddelde zendvermogen ten opzichte van de PEP te verhogen. Bovendien is het spectrum van een met de PC opgenomen geluidsbestand ook breder dan de gewenste band van 300-2800 Hz.



Op de websdr van TU Twente (http://websdr.ewi.utwente.nl:8901) kun je signalen afstemmen met verschillende bandbreedtekeuzes. Ik stem af op ruis in een dode band

rond de 25 MHz.

Je kunt het signaal laten opnemen in Twente in een wav bestand en dat vervolgens down-

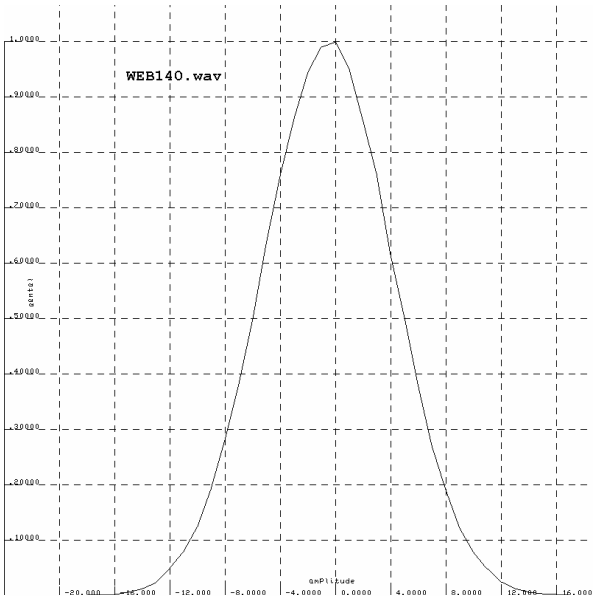


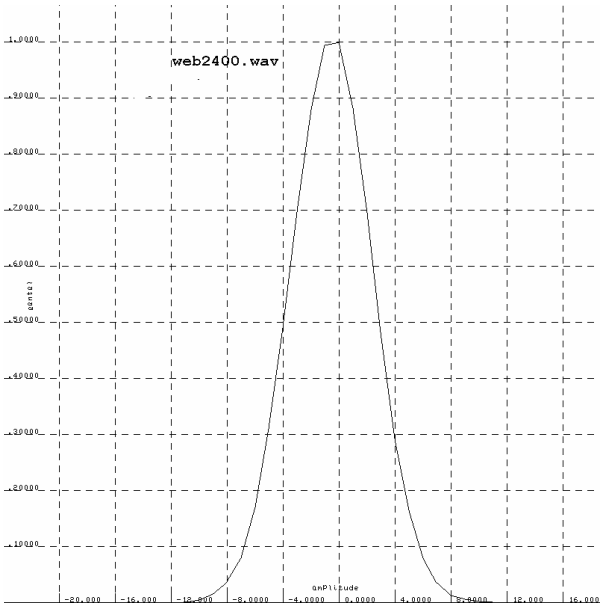
loaden. Dat downloaden gaat zo supersnel, dat de conclusie gerechtvaardigd is, dat dit bestand op je eigen computer wordt voorbereid tijdens het luisteren en opnemen.

Aldus heb ik enkele ruissignalen van diverse bandbreedte gecreëerd.

Deze signalen blijken opgenomen met een bemonsterfrequentie van 7119 samples/s . De header van de wav file voldeed niet geheel aan de standaard, maar dat maakt verder hier niet uit, en is nadat ik PA3FWN er opmerkzaam op maakte inmiddels verholpen.

De genormeerde probability density functies





van de opgenomen ruis is voor de gevallen SSB en CW bandbreedte in grafieken gezet: spectrum en amplitudeverdeling (pdf)

De SSB bandbreedte blijkt hier ook een smallere amplitudeverdeling te vertonen voor de SSB bandbreedte. Ook hier zijn de amplitudeverdelingen dus klokvormig.

De CW ruis is toevallig opgenomen op een centraalfrequentie van 712 Hz, dat is de reden dat dit bedrag als CW beatnote is gebruikt.

Spraak

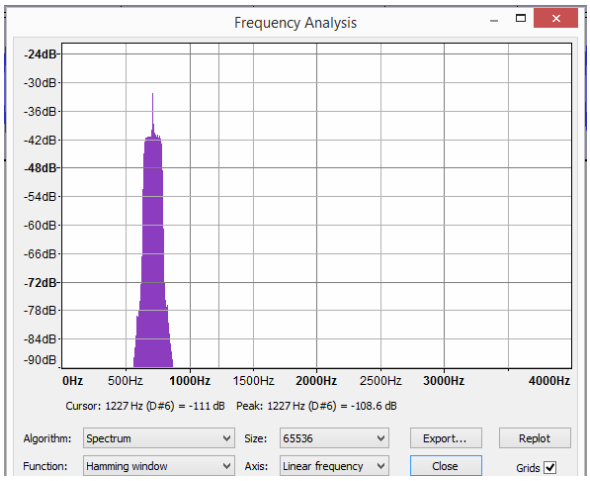
Een nieuw wav bestand is aangemaakt met hetzelfde aantal samples per seconde als de twee ruisbestanden dus 7119 samples/s .

Een dergelijk bestand (navo.wav) blijkt 23 seconde te duren. Zend ik a t/m z achterelkaar in Morse 20 wpm dan duurt dat (cw.wav) minder dan 19 s.

CW wint het dus van het tempo dat je met spraak haalt als je noodzakelijkerwijs moet spellen bij lage signaalruisverhouding, waar dit artikel over gaat. Het bestand out.wav bevat het bestand zonder spaties tussen de spelwoorden. za.wav lepelt ze van achter naar voren op, bij wijze van test om te verifiëren dat ik inderdaad de juiste startposities en lengte van de spelwoorden uit het bestand navo.wav correct heb overgenomen en inge-

typt.

De bedoeling is nu dat de ruis per monster wordt opgeteld met de spraakmonsters, zodanig dat er geen oversturing plaats vindt. Om testsignalen te kunnen maken zocht ik met Audacity in het navobestand de 26 samplenummers waar de spellingwoorden beginnen en de tijdsduur in monsters van die spellingwoorden. Vervolgens schreef ik een program-



ma dat een tekstbestand van een random gekozen 10-lettergroep omzet in een wav bestand met de bijbehorende spellingwoorden, opgeborgen als bestanden met doubles, (drijvende komma getallen per stuk in 8 bytes gecodeerd) om de versterking ervan aan de beschikbare integer ruisbestanden van tutwente of zelf gemaakte ruisbestanden (in double-format) te kunnen aanpassen. . Dat is geen heksentoer en bespaart een bende werk bij het genereren van verschillende tekstbestanden met diverse signaalruisverhoudingen, die nodig zijn om vast te stellen wat ervaren ham-operators nog net correct kunnen registreren. Per gekozen spellingwoord wordt in het bestand de maximale amplitude bepaald en dan aangepast door versterken aan de gespecificeerde max amplitude van het hele bestand. Elk woord heeft dus dezelfde PEP in het bestand met 7 spelwoorden.

De ruisbestanden zijn bij gebruik van tutwente websdr afkomstig van een wav bestand en kunnen dus zonder introductie van extra kwantiseringsruis niet versterkt of verzwakt worden, vandaar dat ik nu het audiodeel dat opgeslagen is als doubles, omdat het bij spraak met Audacity is opgenomen, denk te gaan wijzigen in volume en vervolgens toevoegen aan de ruisbestanden. Een en ander is niet uitgevoerd, omdat de tweede methode beter is, en omdat het spectrum van de opgenomen spraak doorloopt beneden 300 en boven 2800 Hz,

De tweede methode van bandbeperkte ruis maken

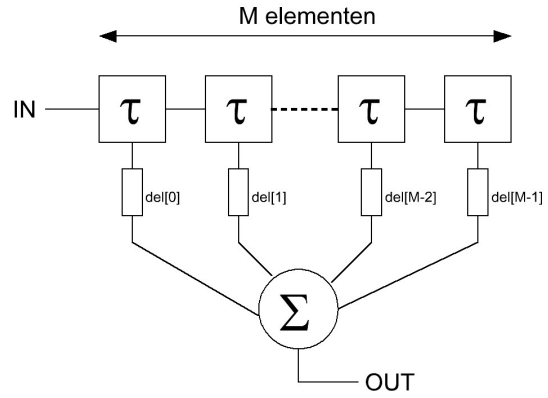
De tweede methode die ik noemde is zelf ruisbanden maken van witte ruis met digitale filters. Die methode sla ik niet over want de zelfontplooiing van de amateur stelt dat het opnieuw uitvinden van het wiel tot de nuttige klasse bezigheden behoort, waarvoor een zendbevoegdheid is verstrekt. En zo ervaar ik dat ook.

Ik sprak eens een amateur, een Tukker, maar dat zal wel toeval zijn, die kocht alles wat hij nuttig achtte voor zijn hobby tenzij het niet te koop is, slechts dan kwam homebrew in aanmerking; want waarom zou hij het wiel opnieuw uitvinden was zijn retorische vraag. Maar, zoals te verwachten, kon hij dan niet maken want niet te koop is. Echter de oplossing daarvoor was dan dat hij dan gewoon niets wilde hebben dat niet te koop is. Ja, die Tukkers zijn me er eentje.

Goed, digitale filters. Een soort bestaat met het acroniem FIR aangeduid, bestaat uit een schuifregister dat analoge waarden kan bevatten, soort emmertjesgeheugen dus, waar de monsters doorgeschoven worden met als klok de samplefrequentie.

De output van elke cel van het register wordt met een factor, een weegcoëfficiënt, die per cel kan verschillen vermenigvuldigd en de uitkomsten van al die vermenigvuldigingen worden opgeteld en vormen de outputpuls van het filter. Een dergelijke bewerking heet convolutie. Het is namelijk de convolutie van het inputsignaal met de set monsterwaarden

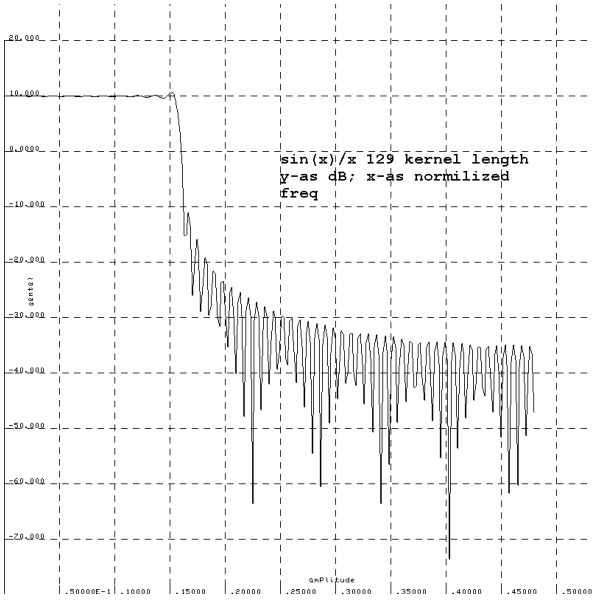
van de weegcoefficienten in het filter.



Als je een brickwall lowpass filter wilt; volledig vlakke doorlaat. buiten de doorlaat oneindige demping, dan kan dat met weegfactoren van de vorm sin(x)/x in zo'n filter.

De sinus heeft een constante amplitude, maar de noemer zorgt ervoor dat die steeds kleiner wordt bij toenemende |x|.

Nu kun je geen oneindig lang filter maken dus probeer ik 129 lang. En ja dat werkt, dat wil zeggen: ik schrijf een computerprogramma dat de beschreven werking realiseert, en dat mag best traag zijn, het gaat om het resultaat. Je kunt dan de werking testen door een wav geluidsbestand te nemen het filter erop los te laten en de output van het filter weer in een wav geluidsbestand op te bergen. Maakt niet uit hoe lang het rekenproces duurt, als het klaar is kun je de output afspelen en beluisteren.



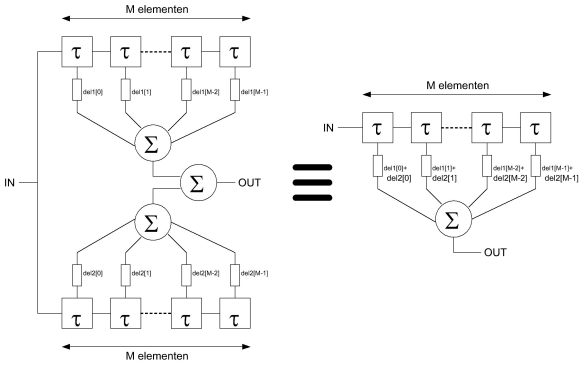
Bandfilters

Nu hebben we een low pass filter, en de vraag is dan hoe je er een highpass van maakt, een bandstop of een banddoorlatend filter.

Eerst de high pass. Die kan op twee manieren:

a) Door een low pass parallel te zetten aan een all pass filter en van de all pass output de low pass output af te trekken. Dan houd je de gewenste high pass over. De versterking en fase van die twee filters moet dan in de doorlaat gelijk zijn, om bij het resultaat van de parallelschakeling daar op 0 uit te komen.

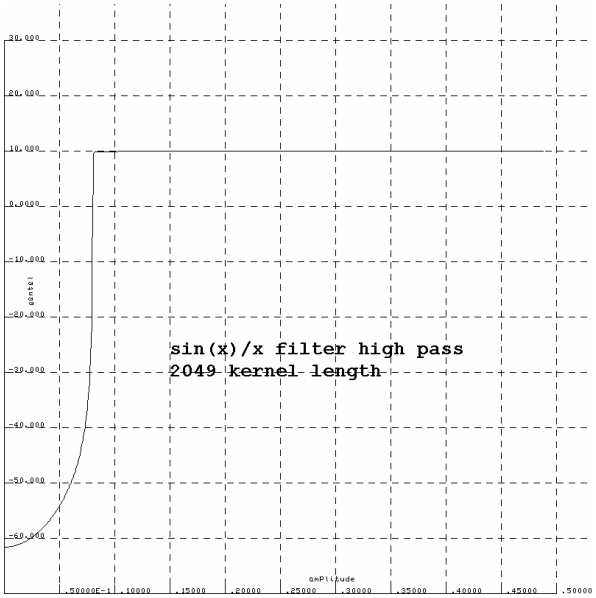
Je hoeft geen twee filters te maken, je kunt omdat het allemaal vermenigvuldigen en optellen is de waarde van de weegfactoren per stel gewoon optellen. Onderstaande tekening verduidelijkt dat.



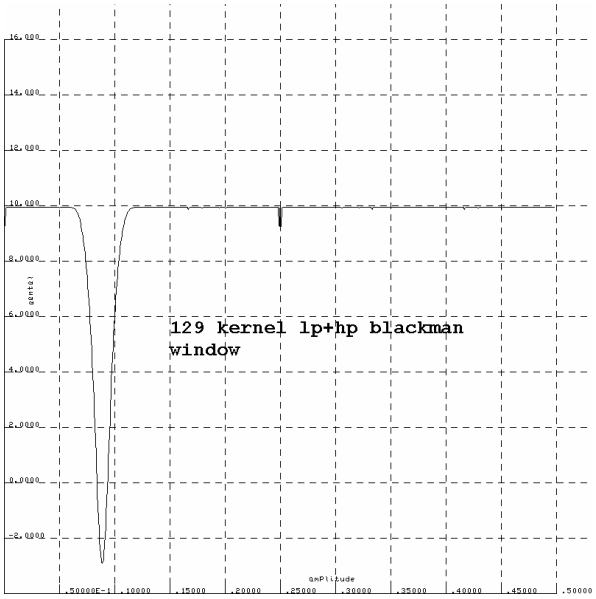
Een all pass heeft slechts **een** van 0 verschillende weegfactor, dus door alleen de middelste weegfactor van de low pass aan te passen verkrijgen we een high pass.

b) Tweede methode is: om de andere waarde van de weegfactoren van de low pass van teken om te keren. In feite vermenigvuldig je dan de output met een sinus fs=0,5 (de halve bemonsterfrequentie) en dus met slechts 2 monsters per periode van die sinus, beurtelings plus en min 1 dus. Het gevolg is dat je de lowpass moduleert op de halve bemonster-

frequentie en dan vind je het spiegelbeeld terug (onderzijband) als hoogdoorlatend filter. Dat kan hier dus ook weer met hetzelfde filter gebeuren, je hoeft alleen maar om het andere (oneven) weegcoëfficiënt van teken om te keren.



Een bandsperfilter kun je maken door een hoog en laagdoorlatend filter parallel te zetten, en ook dat kan weer gecombineerd door met slechts een schuifregister te werken en de weegfactoren van de twee filters, hoog- en laagdoorlatend, op te tellen.



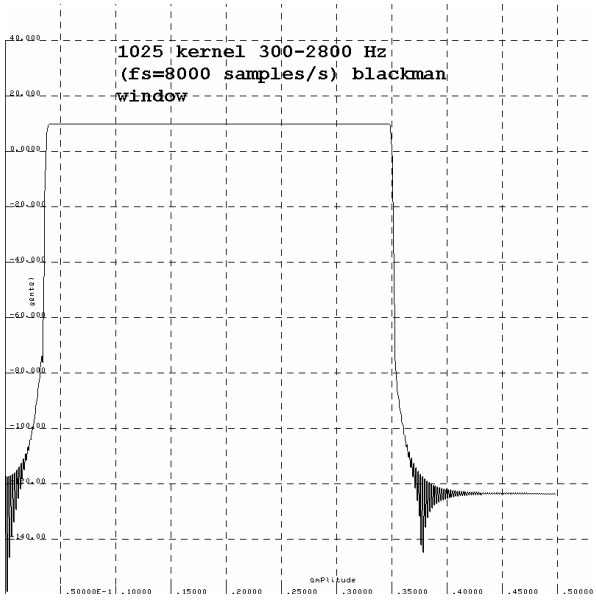
Dan komen we aan de hier van belang zijnde banddoorlatende filters. Ik dacht eerst foutief. Namelijk: maak een laagdoorlatend met als grensfrequentie de bovengrens van het bandfilter en zet dat in serie met een hoogdoorlatend filter met als grensfrequentie de lage grens van het bandfilter. Weer coëfficiënten optellen. Dat werkte van geen kant. Logisch want tot nu toe telden we coëfficiënten op van filters die parallel stonden, hier staan ze echter in serie. Je kunt hier dus niet de twee sets filtercoëfficiënten optellen.

Dat kun je dan oplossen, als je die twee filters in serie wilt omzetten in een filter, door de twee coëfficiëntensets te convolueren. Bezwaar: de set wordt dubbel zo lang. Grap is wel dat je het ene filter kunt voeden met de coëfficiënten van het andere filter als input, de filteroutput is dan die gewenste sdubbel zo lange et coëfficiënten.

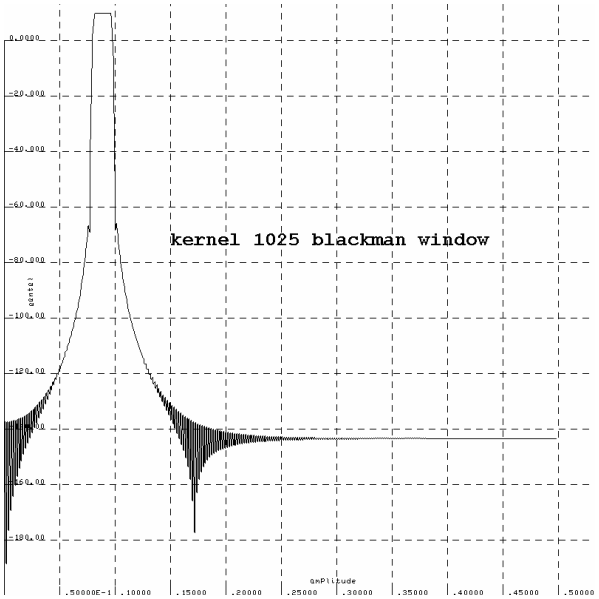
Maar het kan ook anders zonder op dubbele filterlengte uit te komen.

Bijvoorbeeld door het bandsperfilter, dat je wel met twee parallelle laag- en hoogdoorlaatfilters kunt maken, parallel te zetten met een all pass filter. Een all pass heeft alle coëfficiënten 0 behalve de middelste. De middelste moet, om bij aftrekken op resultaat 0 uit te komen, gelijk zijn aan de som van de coëfficiënten van het andere filter. Door alleen de middelste coëfficiënt te wijzigen verander je aldus een bandstop in een banddoorlatend filter. Mooi toch?

Dat is op deze laatste wijze geconstrueerd voor een filter van 1025 lang met een nader



te bespreken Blackman window voor de SSB en CW modellen. De resultaten staan in grafieken hierbij.



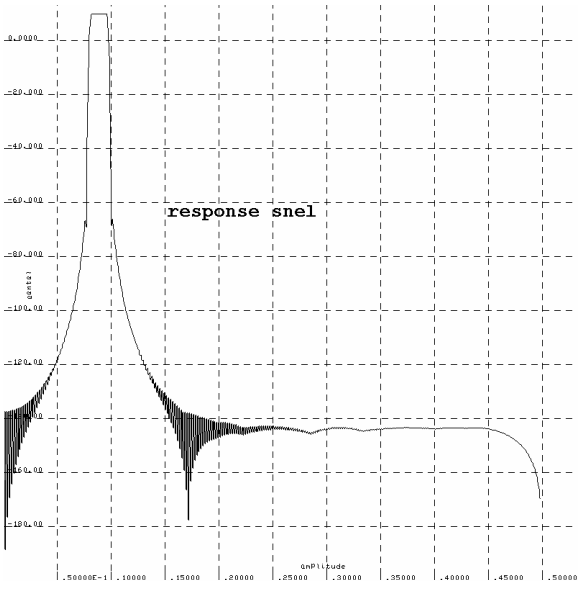
Dempingskarakteristieken

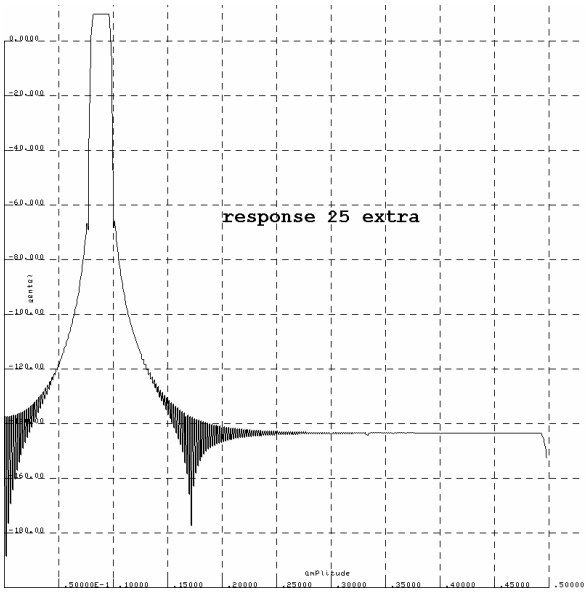
Ik wil ook de dempingskarakteristiek van die digitale filters precies kunnen bepalen. Dat gebeurt dan ook met de computer. Op een groot aantal frequenties genereer ik een sinus in getallen, die getallen (monsters) gaan in het filter, en als er meer monsters van een frequentie inzitten dan de lengte (aantal cellen) van het filter, dan is het verleden, de transient response genaamd, volledig uitgedoofd, en dan wil ik de outputamplitude weten. Een willekeurig monster is dan niet de amplitude, maar doorgaans geringer. Daar kan ik achterkomen door vanaf dat moment door te gaan met die sinusfrequentie als filterinput, tot de sinus minimaal pi verder is in zijn argument; dacht ik. Het maximum is dan zeker opgetreden was het idee. Dat maximum is de amplitude, die wordt omgezet in dB en een punt voor de grafiek van de dempingskarakteritiek voor de inputfrequentie is dan bepaald. Dat doe ik dan voor bijvoorbeeld 500 punten en zo krijg ik dan dus 500 dempingen voor die frequenties, die plot ik en zo vormt zich het plaatje van de diverse filterprobeersels. Het is onnodig om alle vermenigvuldigingen en optellingen te doen in het filter voor de eerste M monsters als het filter M schuifelementen bevat. We kunnen het filter direct vullen zonder die vermenigvuldigingen optelberekeningen, dat scheelt tijd en de transient response wordt dan op een snelle wijze overgeslagen.

De grafieken hebben als horizontale frequentie-as 0 tot 0,5 fs, dat geeft de frequentie aan, uitgedrukt in de bemonsterfrequentie fs. Dus bij 8000 monsters per seconde is 0,5 dan 4000 Hz.

Vertikaal is steeds de 20 log voor basis 10 van de gevonden uitgangsamplitude genomen, dat is dus een dB-schaal. Nadat het filter in M elementen geladen is met sinusmonsters en er dus geen transient response meer kan optreden, moeten we minimaal over een argument pi van sin(2.pi.f.i) monsters invoeren om zeker te zijn dat we een extremum passeren, dus over i monsters met i>= 1/(2f).

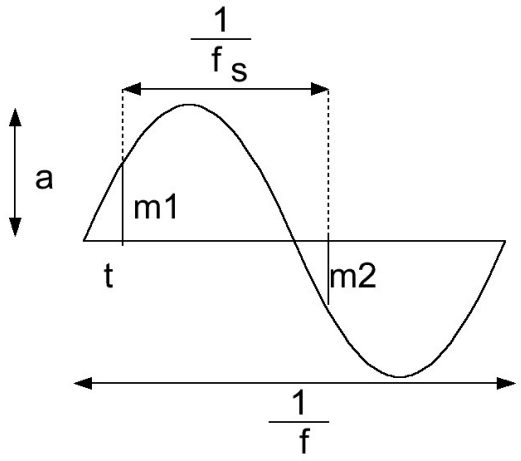
Voor f=0 (gelijkspanning) hebben we dan oneindig veel monsters nodig omdat er geen maximum is, maar alle outputmonsters gelijk zijn. Verdelen we de grafiek van 0 tot 0,5fs in 500 meetpunten, dan vereist de laagste frequentie = 1/1000 fs dan dus maximaal 500 inputsamples. Dat daalt snel bij toenemen van de frequentie om bij de andere limiet f=0,5 fs te eindigen in *een* input monster. Dat vereiste aantal neemt snel af met het toenemen van f. Links op 10% van de breedte van de grafiek is f=0,05 en volstaan dus 10 monsters al. Ik heb op alle punten 25 mon-





sters extra genomen, omdat bovenstaande redenering niet geheel, of beter geheel niet, waterdicht is. Immers als er weinig monsters per sinus zijn, dan is de kans niet groot dat die precies op het maximum van de sinus vallen. Dat is dus de reden dat van de geproduceerde grafieken in de buurt van fs de demping schijnbaar toeneemt, de kans is daar groot dat je geen monster vindt met de topwaarde van de sinus.

Dat leidt er dan toe, om toch te kijken, als een vorm van perfectionisme, of het mogelijk is uit slechts twee naburige monsters de amplitude van de erdoor bepaalde sinus te berekenen. Onderstaande tekening licht dat toe. Je

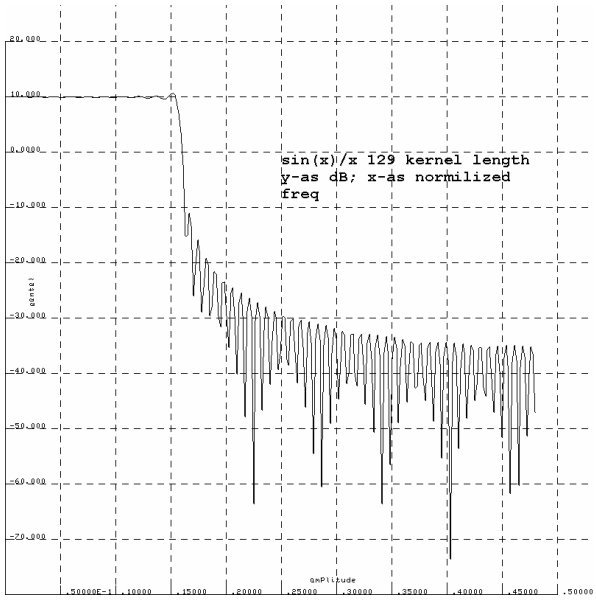


$$m1 = a \sin(2.\pi .f.t)$$

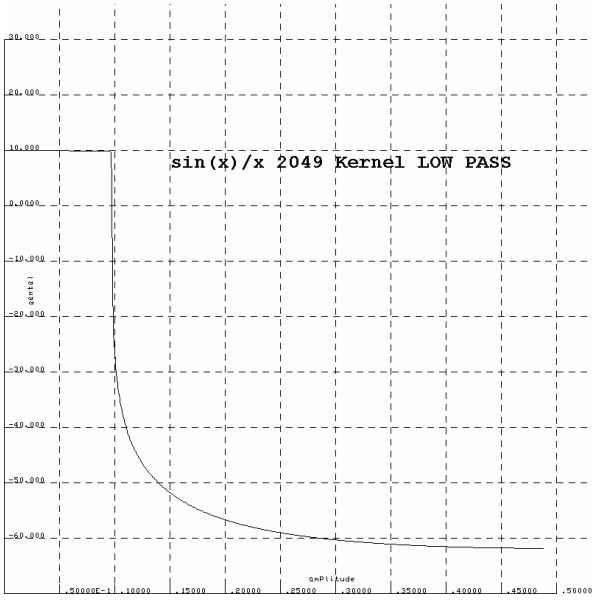
$$m2 = a \sin[2.\pi .f.(t+1/fs)]$$

weet de periodetijd van de sturende sinus en je weet dat de twee naburige monsters 1/fs in tijd verschillen. Dat levert dan twee transcendente vergelijkingen die met iteratief proberen zijn op te lossen.

Ik begon met een low pass als beschreven op blz 12 van 129 cellen. In het spergebied zitten nogal wat terugkomers, en de sperdemping valt tegen.

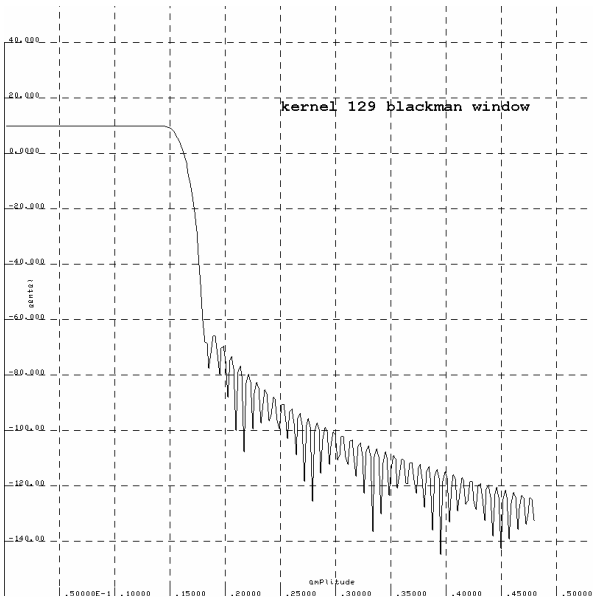


Ik kan ook 2049 lang filter nemen, dan kan ik een kop koffie drinken voor ik de grafiek heb. Die is wel beter zoals hier te zien is.



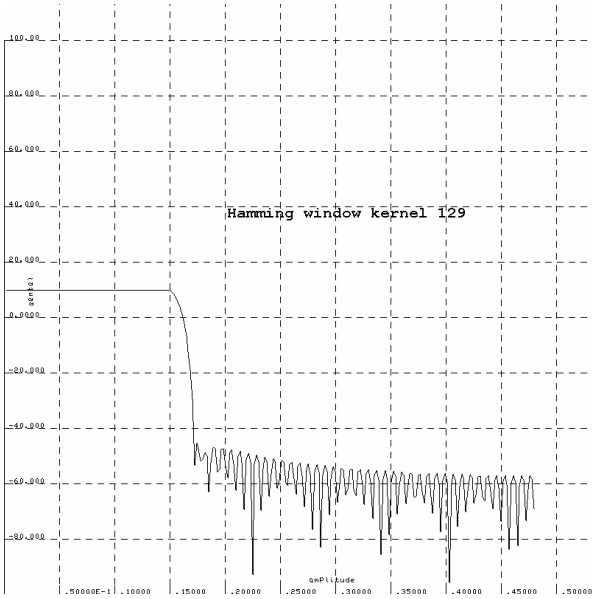
maar dat houdt nog niet over. In de literatuur vond ik dat je over de sin(x)/x dan een windowfunctie kunt toepassen die het abrupte

afkappen van plus en min oneindig tot bijvoorbeeld +64 en - 64 verzacht. Soort op een gekwadrateerde cosinus lijkende functie. Er zijn diverse soorten en ik heb het Blackman window genomen, als blijk van afkeer van discriminatie van zwartmensen maar niet

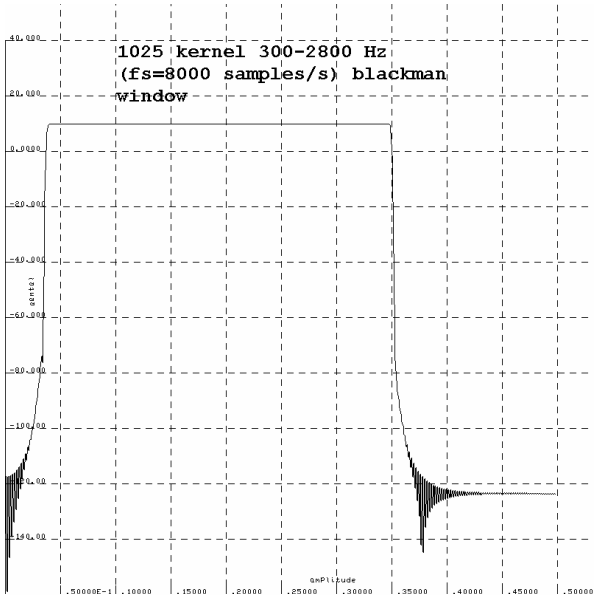


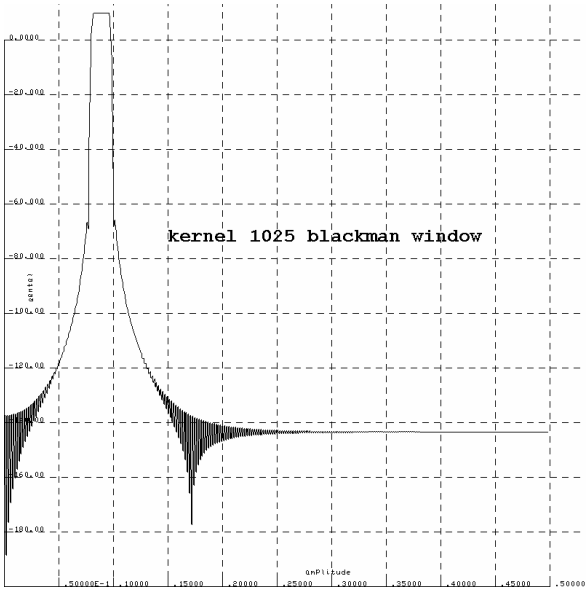
heus. Het Hamming window is ook bekend en die is iets steiler op de flank, maar heeft hogere terugkomers.

De platen hieronder en hierboven tonen het 129 elementen filter als hiervoor, echter nu met een Blackman en met een Hamming window,



Nu wil ik banddoorlatende filters die bij 8 kHz bemonstering 300-2800 Hz doorlaatband hebben en 150 Hz rond de 712 Hz, dat is de toevallige centraalfrequentie van de ruis die ik van TU-twente als wav-bestand downloadde, met 150 Hz bandbreedte. Ik wil die filters flink steil hebben, en dan stuur ik daar witte ruis door en de verkregen bandbegrensde ruis ga ik dan gebruiken voor de meetbestanden, waar er dan SSB of CW signaal bij is opgeteld. Ik ben dan ook niet verplicht de ruisbestanden als 16 bits integers te bewaren, ik kan ze als drijvendekomma getallen (double) opslaan. De filters zijn op de beschreven wijze samengesteld met M=1025 en een Blackman window. De op de beschreven wijze bepaalde frequentiekarakteristiek is hieronder aangegeven





Nu moet je uiterst voorzichtig zijn, en dat is vaak nog onvoldoende, anders konden we geen gebraden kwartel op de menukaart in een decadent restaurant vinden.

Gebaseerd op dat principe ga ik het SSB bandfilter van 300 tot 2800 Hz dat tot 4 kHz input mag hebben vanwege de sample-frequentie van 8000/s, testen met een langzaam in frequentie oplopende sinus (0,01 Hz per 1/8000 s), dus 80 Hz/s tussen 100 en 3000 Hz, Wat schetst mijn verbazing als ik bij oplopende frequenties boven de doorlaatband (>2800Hz) ineens de frequentie weer hoorbaar krijg in DALENDE frequentie terwijl de inputfrequentie van het filter oploopt; (te beluisteren op de eerder genoemde link met bestand digfil10.wav). Dat moet vouwoverspraak zijn van een onderzijband van een harmonische van de monsters; zal wel aan de bejaarde PC geluidskaart liggen, een oudje denk ik dan, maar nee, op een modernere PC idem. en de wav file toont het ook in Audacity; dus het is bij het genereren van de wav file al ontstaan.

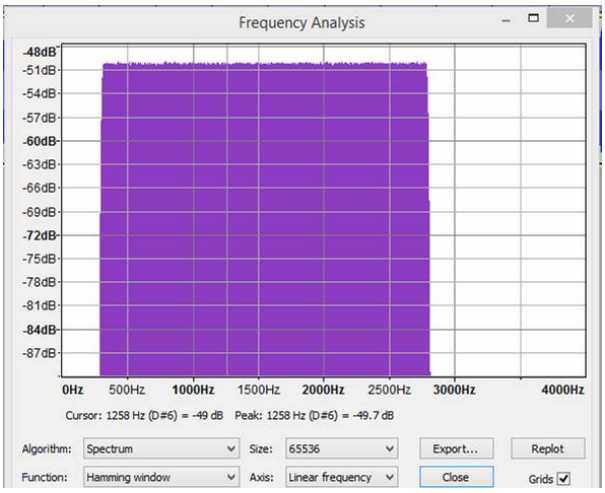
Dat vereist dus nader onderzoek, oplossing blijkt dat ik sin(2.pi.f.t) nam om het signaal te genereren, dat gaat uit van een vaste frequentie en een variabele oplopende tijd. Hier zijn echter zowel de frequentie als de tijd variabel zodat het argument van de sinus de integraal tussen grenzen 0 en t van f(t).dt moet zijn. Dat opgeknapt en nu klopt het wel.

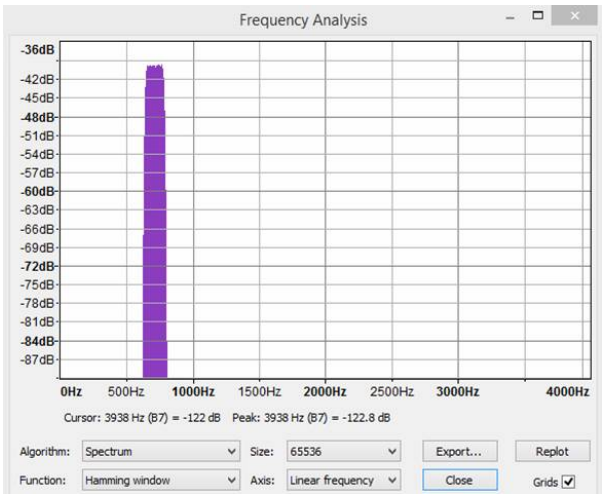
Invloed van het CW filter

De morse van 20 wpm, die is opgeborgen en beluisterbaar in bestand abcdef.wav, is door het CW filter heengestuurd, en opgenomen in abcdef2.wav; het eerste signaal in dat geluidsbestand is het ongefilterde originele signaal 20 wpm 712 Hz toonhoogte, en de wellicht schijnbare herhaling is het oorspronkelijke signaal door het smalle CWfilter heengehaald. Ik hoor net een ietsepietsie verschil bij de opflanken, dus het filter (150 Hz bandbreedte) is zoals te verwachten, niet te smal voor 12 wpm proefbestanden. Later is gewoon de totale energie van de 12 wpm input morse vergeleken met die van de output van het filter, je komt dan uit op 0,00 dB dus dat klopt. Het is ook nog getest met 40 wpm, dan blijkt er 0,04 dB demping op te treden.

Witte ruis in de filters.

Witte ruis gemaakt met random getallen is gemaakt en met zowel het CW filter van 150 Hz bandbreedte als met het SSB filter met bandbreedtegrenzen 300 en 2800 Hz bewerkt tot twee ruisbestanden in double format. Uitsluitend voor controle zijn die omgewerkt tot wav bestanden om met Audacity het ruisspectrum te kunnen bekijken. Dat ziet er perfect uit zoals hieronder moge blijken.





Wijziging van het plan van aanpak

Als ik een sinus van 712 Hz in haakse dits verdeel, krijg ik sleutelklik, tel ik dat signaal

op bij de pluk ruis van 150 Hz breed, dan blijven die clicks erin zitten omdat het spectrum van de CW wordt opgeteld en het breder is dan de ruisband. Dat is onnodig en een risico op meetfouten. Daarom is de wijziging nu dat alle signalen met de te testen S/N verhouding bij de SSB ruis worden opgeteld nadat ze zelf het SSB filter eerst gepasseerd zijn en de CW bij gebruik van het smalbandfilter door dat filter wordt gehaald. De demping voor Morse was 0 en die van de ruis kan zonder signaal worden gemeten door ingangsruisenergie en uitgangsruisenergie te bepalen van het CW-filter.

Toonhoogte

Aanvankelijk werd uitgegaan van 712 Hz omdat dat het midden was van de ruisband die van de websdr van TUtwente werd opgenomen. Nu ik echter zelf de filters construeer staat het mij vrij andere toonhoogten te nemen.

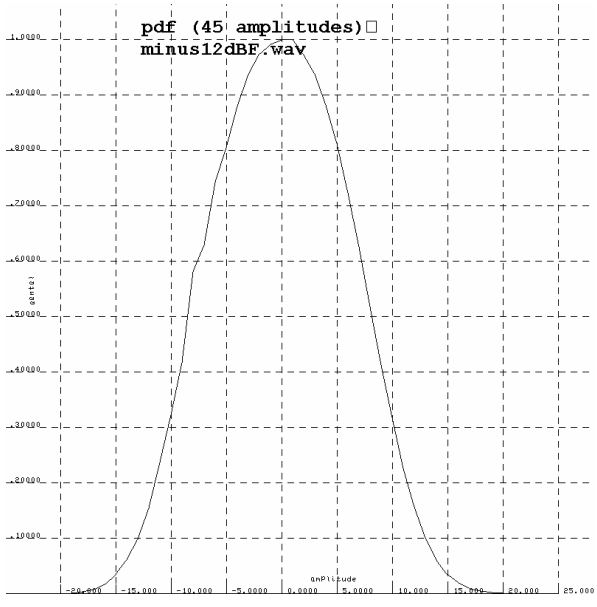
Ik heb een bestand gemaakt met een S/N van min 12 dB op 2,5 kHz brede ruisband, met een toonhoogte die start bij 350 Hz, 12 wpm en bij elke woordspatie gaat de toon automatisch 50 Hz hoger. Dat bestand is beschikbaar gezet op eham.net, waar ervaren telegrafisten zoals VK5EEE, GW3OQK en NI0C rapporteren dan de lagere frequenties betrouwbaarder zijn op te nemen. Dat kan aan hun leeftijdsgebonden achteruitgelopen gehoor liggen. Het bestand is bij dit artikel te beluisteren bij de wav bestanden onder de filenaam minus12f.wav, om te snel oplopen van de frequentie te vermijden zijn een aantal woorden gebundeld met het teken =. Het is zoals de naam als mnemonic reeds aangeeft 12 dB S/N verhouding. Wegens niet lineariteit van het gehoor, is het nuttig met het volume van het geluid te trachten optimale neembaarheid te verkrijgen. Minus10db.wav is een ander gebruikt bestand voor de eerste test

Zwerfbestand

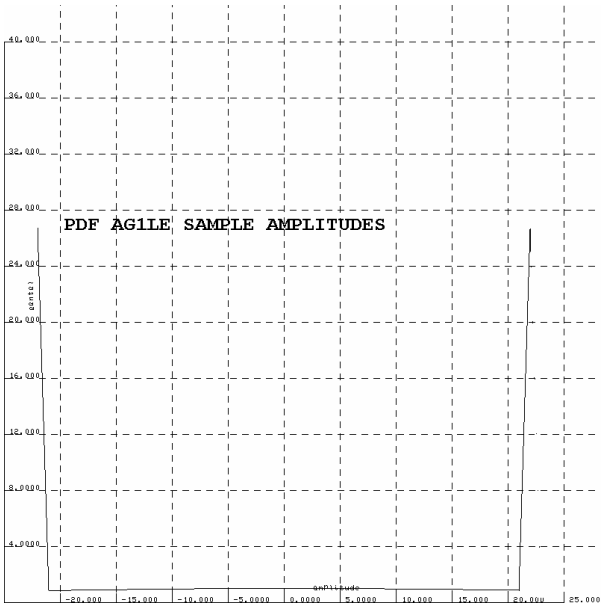
Bij presentatie van beschreven resultaten op eham.net blijkt er een bestand in omloop Rand-10db20wpm.wav en is gemaakt door AG1LE dat -10 dB signaalruisverhouding combineert met 20 wpm CW op 600 Hz toonhoogte volgens opgave. Enkele amateurs berichten dat zij dat beter kunnen nemen dan de door mij gefabriceerde CW op 712 Hz signaalruisverhouding -10 dB 12 wpm. Ruis zou natuurlijker klinken. Dat vergt dus nader onderzoek.

Die file via email ontvangen van Chuck NI0C. Het blijkt dat de samplerate slechts 4000 samples per seconde is, en dat hij overstuurd lijkt. Dat is inderdaad het geval. Blijkbaar is hij gemaakt met random getallen voor de monsters en dus van DC tot 2000 Hz lopend. Echter door oversturing vertoont de amplitudeverdeling enorme pieken door, ten gevolge van limiting, veel te vaak voorkomende maximum en minimum amplitude . Onderstaande figuren tonen de amplitude verdeling van de door mij gemaakte SSB ruis 300- 2800 Hz en de amplitudeverdeling van het bestand van AG1LE.

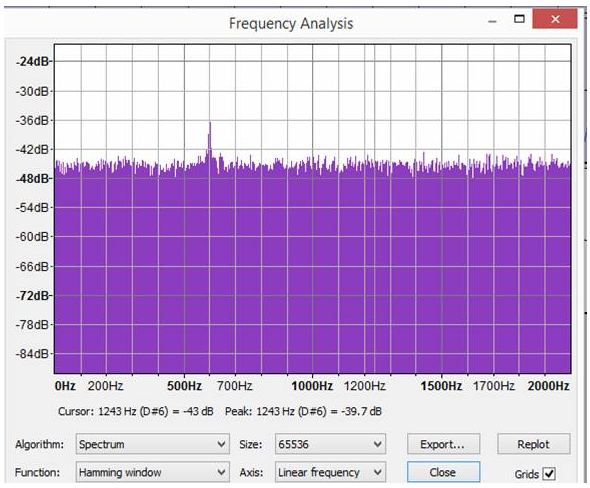
Oversturing leidt tot een blokgolf en die heeft



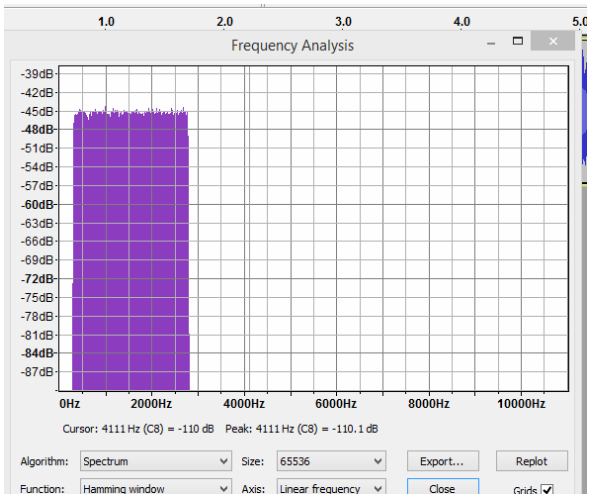
veel harmonischen zodat je met 4000 samples te laag bemonstert om die harmonischen mee te nemen , die vind je dus terug in aliasing (vouw-overspraak)

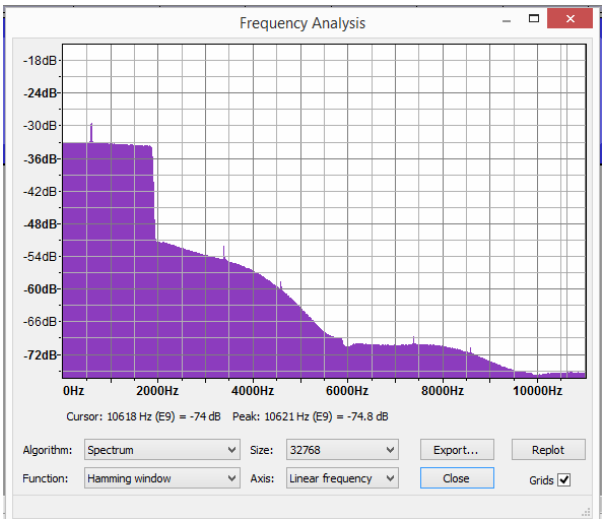


Dat blijkt ook als je de spectra bekijkt



Dit doet me afvragen wat resampling van het originele signaal met een hogere sampling rate voor resultaat zou hebben.





Daartoe zowel mijn eigen geproduceerde 300-2800 Hz als het bestand van AG1LE herbemonsterd op 11000 samples/s met Audacity.

Als resultaat blijkt dat het door mij geproduceerde signaal geen spectrumcomponenten boven 2800 Hz erbij krijgt maar dat van AG1LE wel hele lappen, Dat is in feite dus de uitgevouwen vouwoverspraak.

Dat bestand van AG2LE dus niet gebruiken anders dan voor oefening van opnemen van CW in ruis, waar NI0C het ook voor gebruikte.

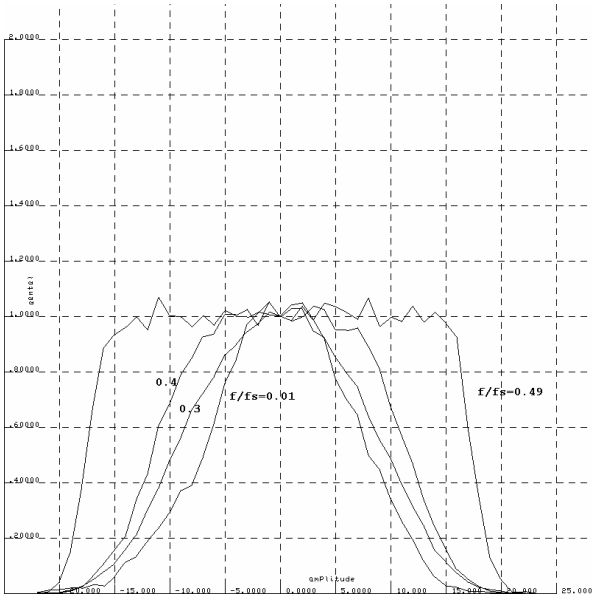
Gefilterde witte ruis.

We hebben eerder gezien dat ruis gevormd door random getallen een spectrum heeft dat vlak verloopt tussen 0 en de halve samplefrequentie. Ik verbaasde me toen over de Gausse verdeling van ruis op de band en zelf geproduceerde bandbegrensde ruis, met de sinussommatienethode.

Nu leek me het moment gekomen om dat nader te onderzoeken. Ik heb daartoe een laagdoorlatend digitaal filter geprogrammeerd met variabele grensfrequentie.

De volle bandbreedte van random-getallen witte ruis loopt van 0 tot 0,5 fs, de halve samplefrequentie. Het filter gaf ik een doorlaatband van 0,49 fs. De output blijkt dan nog

een vrijwel vlakke pdf te vertonen (alle amplitudes van monsters komen even vaak voor) Bij grensfrequentie fg=0,4 fs en lager treedt de gausse verdeling al duidelijk op, gevolg van het central limit theorema, alleen de breedte van de klokvorm krimpt wat bij lagere grensfrequenties. Een combinatiegrafiek voor fg tussen 0,49 fs en 0,01 fs is hieronder afgedrukt.



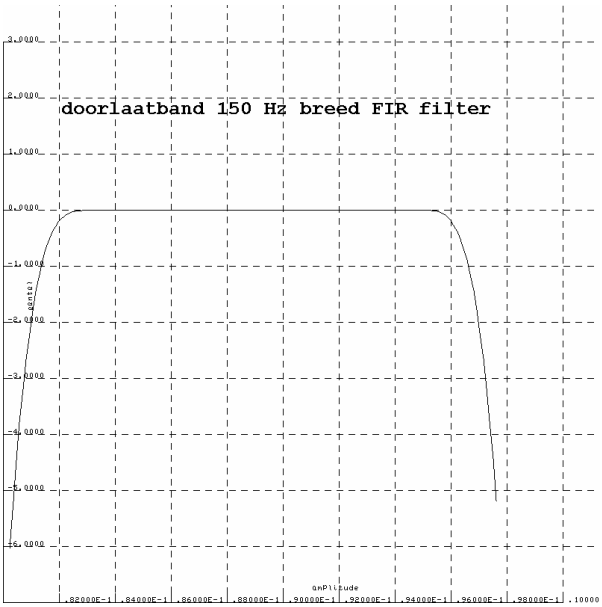
Gekozen methode voor CW filteren

Om CW in bandbeperkte ruis te maken lijkt het me achteraf verstandiger om niet de oorspronkelijke ruisvrije CW op te tellen bij een bandbeperkt ruissignaal, maar om CW in een SSB brede ruisband opgeteld, die vervolgens te filteren met een CW filter waar beide (2,5 kHz brede ruis en CW) gezamenlijk doorheengaan. Een en ander is ook van belang in verband met de transient response van het filter, die wijzigingen in het signaal geeft en om eventuele spectrumcomponenten van CW buiten de CW band gegarandeerd te elimineren..

De werkwijze is dan als volgt:

1. Bepaal in de doorlaatband de demping van het CW filter. Die is niet geheel vlak.

resultaat hiernaast. Niet noodzakelijk maar



wel interessant. Omdat het filter niet vlak is in het doorlaatgebied kunnen de zijbandfrequenties van het CW signaal verzwakken en aldus een wat andere stijgtijd vertonen.

2. Stuur uitsluitend SSBband brede ruis 300-2800 Hz. door het filter en bepaal de verhouding r van het ruisvermogen op de ingang vergeleken met het tot 150 Hz bandbeperkte ruisvermogen op de uitgang. We zagen eerder dat dit idealiter 12,2 dB zou moeten zijn, omdat het ruisvermogen evenredig is met de bandbreedte.

3. Bepaal zonder ruis de dempingsfactor c voor een aan het filter aangeboden CW signaal, door totale energie die het filter ingaat en uitkomt op elkaar te delen.

4. Voeg CW en ruis op de ingang van het filter samen in een zodanige verhouding dat met de kennis opgedaan in punt 3 en 2 de gewenste S/N verhouding op de uitgang van het filter wordt bereikt. Dat wil zeggen dat op de uitgang de S/N verhouding r/c maal de bekende S/N is op de ingang van het filter.

Een en ander vereenvoudigt omdat de demping voor 12 wpm Morsecode 0 blijkt te zijn, die neemt trouwens aanvankelijk niet snel toe want de gemeten demping bij 40 wpm is 0,05 dB.

De CW test

Er zijn 8 bestanden gemaakt a.wav t/m h.wav met elk 4 groepen van 5 random gekozen letters. Die bestanden staan beschikbaar op http://pa0wv.home.xs4all.nl/eham/a.wav etc. dus b.wav t/m h.wav

De bestanden zijn beurtelings CW in 2,5 kHz brede ruis, en vervolgens na het CW filter (met andere karakters), Het begint als eerste met CW in SSB-brede ruis met minus 10 dB signaalruisverhouding en loopt dan steeds 1 dB naar beneden per paar tot minus 13 dB S/N Altijd is de S/N na het filter 12,2 dB beter door het wegsnijden van het grootste deel van de ruis. Het ruissignaal is steeds hetzelfde bestand voor gebruikt.

Resultaten

Gemiddeld percentage fouten van de 4 deelnemers aan de test

-10 dB SSB filter errors	0,75%
na CW filter	2,50%
-11 dB SSB filter	12,5%
na CW filter	16,25 %
-12 dB SSB filter	32,5%
na CW filter	21,25 %
-13 dB SSB filter	65%
na CW filter	52.5 %

De test is 288 keer gedownload, maar er hebben slechts 4 personen resultaten ingezonden. Buiten enkele oude rotten die verklaarden alleen maar ruis te horen.

Conclusie CW metingen

Het CW filter knapt de S/N 12,2 dB op, - 10 dB wordt dus 2,2 dB, maar het foutenpercentage blijft ruwweg gelijk. We kunnen dus concluderen dat het brein van de ham die naar Morse luistert reeds als bandfilter werkt, dat op 712 Hz niet breder en mogelijk smaller is dan 150 Hz. Het Morsefilter levert dus geen betere ontvangst op in ruis, de foutkans is nagenoeg gelijk, maar uiteraard snijdt het wel hinderlijke QRM weg. Het vermeende voordeel in zendvermogenswinst waar ik van uitging van 12,2 dB tengevolge van het gebruik van een CW filter aan de ontvangstkant bestaat dus niet.

Zelf controleren met de beschikbaar staande bestanden kan u wellicht beter overtuigen.

Spraak

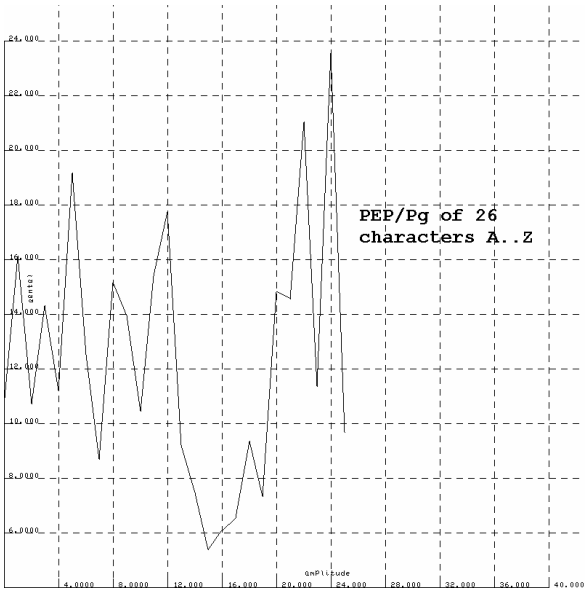
We willen SSB met CW vergelijken, dus nu is spraak aan de beurt om nauwkeuriger bekeken te worden.

Het NAVO-alfabet dat oorspronkelijk 7119 monsters per seconde bevatte omdat ik de ruis van de websdr geluidsopname wilde gebruiken, is herbemonsterd met 8000 monsters per seconde. Lastig want ik moest dan alle 52 woordgrenzen in het bestand weer bepalen in intikken.

Vervolgens is het bestand door het SSB filter gestuurd, en de uitgang van het filter in drijvende kommanotatie opgeslagen als bestand navodoub. . Ruis met SSB bandbreedte 300-2800 Hz was al voor de CW test aangemaakt als een doubles file met titel druis.bak.

Controle van druis.bak is gebeurd door die om te zetten in een wav file, en daarvan het spectrum te bekijken met Audacity. Perfect.

Ik had al eerder bemerkt dat het spectrum van mijn spraak niet beperkt is tot een bandbreedte van 2,5 kHz zodat dat het filter daar alsnog voor zorgt. De output van het filter is eveneens opgeslagen als doubles. Echter van elk woord dus 52 stuks is de verhouding PEP/Pg bepaald (zie grafiek hieronder) en in de representatie van elk woord in de doubles bestand een andere



A max= 32760.00 PEP/Pgs= 10.90

B max= 32760.00 PEP/Pgs= 11.87

C max= 32760.00 PEP/Pgs= 11.81

D max= 32760.00 PEP/Pgs= 12.47

E max= 32760.00 PEP/Pgs= 14.55

F max= 32760.00 PEP/Pgs= 21.44

G max= 32760.00 PEP/Pgs= 12.05

H max= 32760.00 PEP/Pgs= 9.06

I max= 32760.00 PEP/Pgs= 16.24

J max= 32760.00 PEP/Pgs= 14.43

K max= 32760.00 PEP/Pgs= 9.90

L max= 32760.00 PEP/Pgs= 15.59

M max= 32760.00 PEP/Pgs= 16.88

N max= 32760.00 PEP/Pgs= 10.30

O max= 32760.00 PEP/Pgs= 7.27

P max= 32760.00 PEP/Pgs= 6.16

Q max= 32760.00 PEP/Pgs= 6.11

R max= 32760.00 PEP/Pgs= 6.35

S max= 32760.00 PEP/Pgs= 10.51

T max= 32760.00 PEP/Pgs= 7.21

U max= 32760.00 PEP/Pgs= 14.84

V max= 32760.00 PEP/Pgs= 16.02

W max= 32760.00 PEP/Pgs= 16.72

X max= 32760.00 PEP/Pgs= 12.61

Y max= 32760.00 PEP/Pgs= 22.88

Z max= 32760.00 PEP/Pgs= 8.69

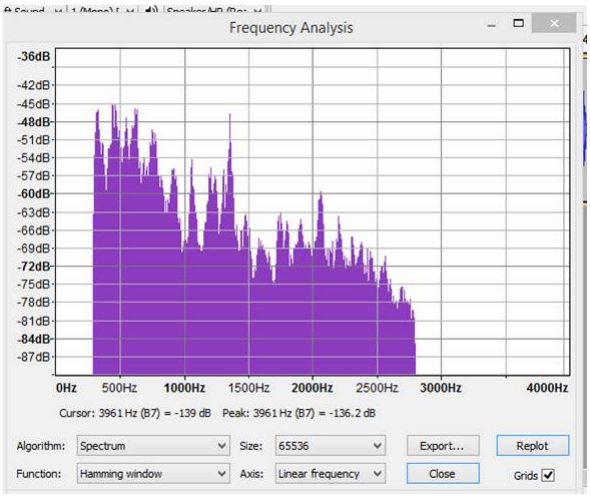
overall PEP/Pgs words only= 11.09

---

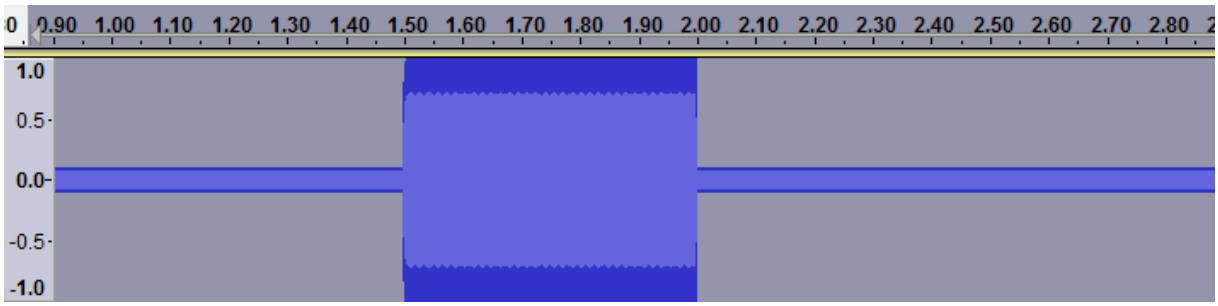
versterkingsfactor toegepast, zodat alle navo woorden die daarin te kiezen zijn, dezelfde PEP hebben.

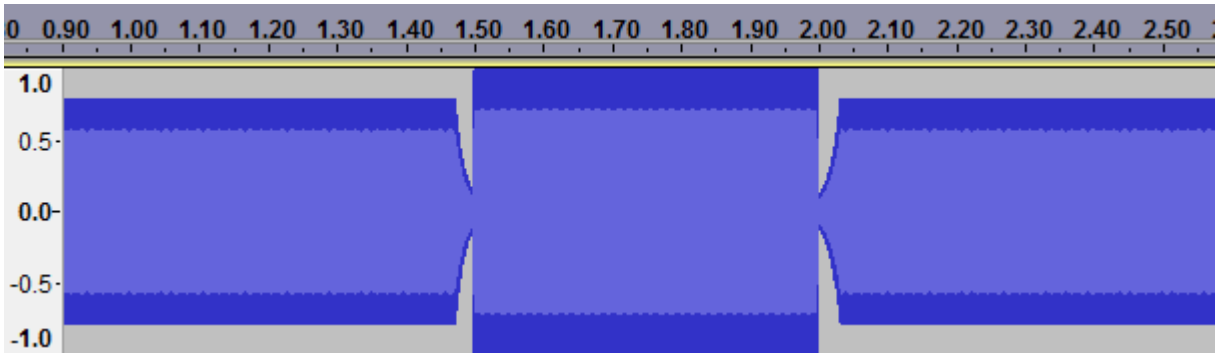
Compressie kan dan de verhouding PEP/Pg verbeteren, maar dat is nu nog niet aan de orde.

Als controle is het navodoub bestand getest door het van achter (zoeloe) naar voren (alfa) om te zetten in een wav bestand en dat af te spelen. Gaat ook perfect. Tevens is het spectrum van die spraakbestand bekeken en ook dat is keurig beperkt tussen 300 en 2800 Hz.



Samenvattend hebben we nu dus een bestand van navowoorden, elk woord afzonderlijk afspeelbaar, met elk dezelfde PEP, dat we gaan gebruiken om daar een random greep woorden in te doen en die tezamen op de erbij opgetelde ruis tot de gewenste signaalruis-





verhouding in een wav bestand te zetten.

Testbestanden bestaan uit 10 random letters uitgesproken in NAVO alfabet met alle letters dezelfde PEP en een signaalruisverhouding van 6 dB tot 0 dB

Gemiddelde foutkans van de respondenten in opnemen:

Ssb6.wav 15%

Ssb5.wav 20%

Ssb4,wav 25%

Ssb3.wav 25%

Ssb2.wav 35%

Ssb1.wav 35%

Ssb0.wav 70%

Opvalt dat de spreiding veel groter is dan bij CW, daar nam het aantal fouten zeer snel toe als de S/N verhouding slechter werd aangeboden.

Compressie van het NAVO-alfabet

Je pompt bij gelijke PEP meer energie de ether in als je verhouding PEP/Pg afneemt. Die verhouding drukken wordt compressie genoemd

Je kunt comprimeren door te klippen en dan met een filter de harmonischen die buiten de band vallen wegsnijden, dat geeft weer nieuwe pieken en dat proces kun je een paar keer herhalen want het convergeert enigszins. Niet fraai, want je krijgt ook inband componenten die er niet thuishoren gratis bij.

Andere methode is als je een sterker signaal ziet aankomen de versterking terug te draaien en als het sterke signaal voorbij is weer de versterking op te draaien. De achtergrondlawaai wil je niet versterken dus je kunt beneden een drempel dat effect niet toepassen.

Voordeel is data door dat langzaam volumevarieren je een vorm van amplitudemodulatie introduceert en die heeft geringere zijbandbreedte naarmate je langere attack en releasetijden van die amplituderegeling aanhoudt. Harmonischen in- en outband krijg je er denk

ik niet mee. Voorbeeld het oscillogram hierboven, het eerste zonder compressie en het tweede met/

Met Audacity is compressie uitgevoerd, dat kan aotomatisch met behoud van PEP

De gebruikte parameters zijn:

Threshold -26 dB

Noisefloor -65 dB

Ratio 10:1

Attacktime 0.01 s

releasetime 0,1 s

Op Internet is een handleiding te vinden die de werking verduidelijkt. Je kunt de werking ook zien of de geluidsdisplay van Audacity als je een sinusvormig signaal met volle amplitude genereer, voorafgegaan en gevolgd door een sinus met een tiende van de amplitude. Na de compressie kun je dan het spectrum bekijken, dat mag dan bijvoorbeeld geen harmonischen bevatten.

Hoewel zwakkere geluiden opgehaald worden en de omgevingsachtergrondgeluid tussen woorden in pauzes opvallend groeit, blijkt PEP/Pg van een geluidsbestandaanzienlijk te verGROTEN na compressie.

Programma, dat dat vaststelt nagelopen en getest met morse. De Pg/PEP van morse Paris paris bestand blijkt 0,434 te zijn in plaats van theoretisch 0,44. Dat geringe verwaarloosbare verschil ontstaat wellicht door de niet oneindig steile stijg- en daaltijden, die ten koste van de mark-duur gaan en dus Pg verlagen. Een test met PARIS met stijg en daaltijd 0 bevestigt dat vermoeden. Een probeersel met een morsetoonhoogte van 3,5 kHz levert hetzelfde resultaat als op 712 Hz in het SSB filter.

Een periode van een sinus van 1/20 Hz in een wav file gezet van 20 s lang, en die als proefexemplaar gebruikt. PEP /Pg blijkt inderdaad 1 te zijn (op 5 decimalen nauwkeurig)

Op eham.net een discussie gestart, gecomprimeerde files daar blijken een slechtere PEP/Pg te hebben dan mijn nato bestand, dus ik zie daar verder van af. Een uitnodiging aan het aan het forum deelnemende publiek om een wav file te leveren die met een bandbreedte van 300 tot 2800 Hz een betere PEP/Pg heeft dan de door mij gemaakte bestand leverde geen resultaat op.

Wel werden twee demo spraakbestanden aangeleverd door W6RZ gecomprimeerd (PEP/Pg 4.45) en zelfde bestand niet gecomprimeerd (PEP/Pg=47,47 die een bandbreedte hadden van 15 kHz. Bij bandbeperking tot 2,5 kHz werd het gecomprimeerde bestand echter een stuk slechter (PEP/Pg=16.0) dan mijn navo bestand dat gemiddeld 11.09 is. Een heilloos pad, dat ik verder niet ga aflopen.

W6RZ.PDF staat als samenvattung beschikbaar op die filenaam geplaatst op de inmiddels gebruikelijke URL.

Conclusie

1. Telegrafie in een 2,5 kHz brede ruisband is ongeveer 14 dB lagere S/N dan SSB betrouwbaarder te nemen dan met NAVO spellig uitgespelde SSB tekst. Bij dezelfde PEP aan zendzijde

2. Het gebruik van een smal CW filter geeft bij afwezigheid van QRM nauwelijks verbetering in de betrouwbaarheid van de opgenomen random tekst bij menselijke opname.

3. Degradation of receiving reliability is per dB S/N ratio less with SSB.