Tampereen teknillinen yliopisto. Signaalinkäsittelyn laitos.
Opetusmoniste 2014:1
Tampere University of Technology. Department of Signal Processing.
Lecture Notes 2014:1

Heikki Huttunen

Signaalinkäsittelyn perusteet

TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

Tampereen teknillinen yliopisto. Signaalinkäsittelyn laitos.
Opetusmoniste 2014:1
Tampere University of Technology. Department of Signal Processing.
Lecture Notes 2014:1

Heikki Huttunen

Signaalinkäsittelyn perusteet

Tampereen teknillinen yliopisto. Signaalinkäsittelyn laitos
Tampere 2014

ISBN 978-952-15-3221-4 (nid.)
ISBN 978-952-15-3222-1 (PDF)
ISSN 1459-4609

Esipuhe

Käsillä oleva moniste on tarkoitettu opetusmateriaaliksi Tampereen teknillisen yliopiston signaalinkäsittelyn laitoksen kurssille "SGN-11000 Signaalinkäsittelyn perusteet". Materiaali on kehittynyt nykyiseen muotoonsa luennoidessani aiheesta Jyväskylän yliopistossa lukuvuonna 1995–1996, Tampereen yliopistossa lukuvuonna 1999–2000 sekä Tampereen teknillisessä yliopistossa lukuvuosina 1999–2013. Tähän painokseen on yhdistetty materiaalia aiemmista monisteista "SGN-1201 Signaalinkäsittelyn menetelmät" (2005) ja "SGN-1251 Signaalinkäsittelyn sovellukset" (2013).

Luentomoniste en runko koostuu signaalinkäsittelyn teorian keskeisistä aiheista, kuten diskreetit signaalit, niiden ominaisuudet ja generointi Matlabilla, lineaariset järjestelmät, diskreetti Fourier-muunnos, FFT, z-muunnos, suodinsuunnittelu sekä näytteenottotaajuuden muuntelu. Tämän lisäksi kurssilla on tapana käsitellä myös tavallisimpia signaalinkäsittelyn sovelluksia, kuten puheenkäsittely, kuvankäsittely ja -koodaus, digitaalinen video (MPEG-standardit), lääketieteellinen signaalinkäsittely, ja hahmontunnistus. Vierailevat luennoitsijat laitokselta ja teollisuudesta hoitavat mahdollisuuksien mukaan osan sovellutusten esittelystä.

Kurssin tavoitteena on selvittää lineaaristen järjestelmien ja digitaalisen signaalinkäsittelyn peruskäsitteet sekä luoda kuva sovelluskohteista. Kurssin käytyään opiskelijan tulisi ymmärtää millaista signaalinkäsittelyn parissa työskentely on ja minkä tyyppisiin ongelmiin sitä voidaan soveltaa.

Lisäinformaatiota löytyy kurssin kotisivulta, jonka osoite on

`http://www.cs.tut.fi/kurssit/SGN-11000/`

Monisteen aiempia versioita on käytetty opetusmateriaalina vastaavilla kursseilla Tampereen AMK:ssa, Kemi-Tornion AMK:ssa, Jyväskylän AMK:ssa, Oulun AMK:ssa, Kuopion yliopistossa sekä TTY:n koordinoimassa DI-muuntokoulutuksessa Kuopiossa. Pyynnöstä materiaalin käytölle voidaan myöntää lupa myös muualla. Tällöin voin tarvittaessa toimittaa myös luentokalvot sekä harjoituksissa käytettäviä Matlab-skriptejä.

Tampereella, 3. tammikuuta 2014,
Heikki Huttunen
heikki.huttunen@tut.fi

Sisältö

Esipuhe **i**

1 Digitaalinen signaalinkäsittely **1**
1.1 Mitä signaalinkäsittelyllä tarkoitetaan 1
1.2 Näytteenottoteoreema 3
1.3 Digitaalisen signaalinkäsittelyn etuja ja haittoja jatkuva-aikaisiin suodattimiin nähden 5
1.4 Sovelluskohteita 6

2 Signaalit ja järjestelmät **11**
2.1 Eräitä signaaleita 12
2.1.1 Jonojen ominaisuuksia 14
2.1.2 Perusoperaatiot jonoille 14
2.2 Diskreettien järjestelmien ominaisuuksia 16
2.2.1 Muistittomuus 16
2.2.2 Lineaarisuus 16
2.2.3 Siirtoinvarianssi 17
2.2.4 Kausaalisuus 18
2.2.5 Stabiilisuus 19
2.3 Lineaariset siirtoinvariantit (LTI) järjestelmät 20
2.3.1 Impulssivaste ja konvoluutio 20
2.3.2 Konvoluution ominaisuuksia 22
2.3.3 FIR- ja IIR-suotimet 24

3 Fourier-muunnos **33**
3.1 Fourier-muunnos (ei-jaksollinen jatkuva-aikainen signaali) 34
3.2 Fourier-sarja (jaksollinen jatkuva-aikainen signaali) 35
3.3 Diskreettiaikainen Fourier-muunnos (diskreettiaikainen ei-jaksollinen signaali) 38
3.4 Diskreetti Fourier-muunnos (diskreettiaikainen jaksollinen signaali) 39
3.4.1 Diskreetin Fourier-muunnoksen ominaisuuksia 43
3.4.2 Nopea Fourier-muunnos (FFT) 45

4 *Z*-muunnos **53**
4.1 *Z*-muunnoksen määritelmä 53
4.2 Tavallisimpien jonojen *z*-muunnokset 55

4.3 Käänteinen z-muunnos . . . 57
4.4 Z-muunnoksen ominaisuuksia . . . 59
4.5 Siirtofunktio . . . 60
4.5.1 FIR-suotimen siirtofunktio . . . 60
4.5.2 Taajuusvaste, amplitudivaste ja vaihevaste . . . 61
4.5.3 Desibeliasteikko . . . 65
4.5.4 IIR-suotimen siirtofunktio . . . 66
4.6 Stabiilisuus . . . 69

5 Suotimen suunnittelu taajuustasossa 75
5.1 FIR-suodinten suunnittelu: suunnittelukriteerit . . . 76
5.1.1 Vaihevasteen vaatimukset . . . 76
5.1.2 Amplitudivasteen vaatimukset . . . 79
5.2 Suunnittelu ikkunamenetelmällä . . . 80
5.3 Yhteenveto ikkunamenetelmän käytöstä . . . 85

6 IIR-suodinten suunnittelu 95
6.1 Butterworth–suotimet . . . 95
6.2 Tyypin I Chebyshev-suotimet . . . 98
6.3 Tyypin II Chebyshev-suotimet . . . 101
6.4 Elliptiset suotimet eli Cauer-suotimet . . . 103
6.5 Suodintyyppien vertailua . . . 105

7 Äärellisen sananpituuden vaikutukset 111
7.1 AD-muunnoksen kvantisointivirhe . . . 112
7.2 Suodatuksen vaikutus kvantisointikohinaan . . . 114
7.3 Kertoimien pyöristämisen vaikutus . . . 117

8 Näytteenottotaajuuden muuntelu 123
8.1 Desimointi eli näytteenottotaajuuden alentaminen . . . 124
8.2 Näytteenottotaajuuden pienentäminen useassa vaiheessa . . . 127
8.3 Interpolointi eli näytteenottotaajuuden nostaminen . . . 128
8.4 Näytteenottotaajuuden muunnos rationaalikertoimella . . . 130
8.5 Interpoloinnin käyttö D/A-muunnoksessa . . . 131
8.5.1 Nollannen asteen pitopiiri . . . 131
8.5.2 Kohinanmuokkaus . . . 134

9 Digitaaliset signaaliprosessorit 143
9.1 Circular buffering . . . 144
9.2 Kiinteän vai liukuvan pilkun aritmetiikka? . . . 146
9.3 C:llä vai konekielellä? . . . 146

Luku 1

Digitaalinen signaalinkäsittely

Digitaalisesta signaalinkäsittelystä on tullut yksi nykytekniikan avainaloista, ja se tukee läheisesti ainakin tietoliikennetekniikkaa, mittaustekniikkaa ja tietotekniikkaa. Digitaalisen signaalinkäsittelyn (Digital signal processing, DSP) voidaan katsoa syntyneen 1960–1970–luvuilla, jolloin tietokoneet alkoivat olla riittävän yleisesti käytettävissä. Tämän jälkeen sitä on menestyksekkäästi sovellettu lukuisilla alueilla; lääketieteellisestä PET-kuvantamisesta CD-soittimeen ja GSM-puhelimeen.

Sovellukset ovat varsin lukuisat, joten kaikkea DSP:stä on mahdotonta hallita (eikä niitä ole mielekästä opettaa korkeakoulussa). Tärkeimmät perusmenetelmät ovat kuitenkin pysyneet vuosien varrella samoina. Jatkossa on tarkoitus käsitellä

- tärkeimmät peruskäsitteet,
- osa tärkeimmistä menetelmistä,
- esimerkkisovelluksia.

Kun lineaaristen järjestelmien perusasiat on käsitelty, perehdytään tyypilliseen signaalinkäsittelyn ongelmaan: kuinka poistaa tietyt taajuudet annetusta signaalista.

Tulevilla kursseilla perehdytään tarkemmin signaalinkäsittelyn menetelmiin sekä tarkastellaan sovelluskohteita lähemmin.

1.1 Mitä signaalinkäsittelyllä tarkoitetaan

Tyypillinen DSP-sovellus sisältää seuraavat vaiheet:

1. Niin sanottu A/D-muunnin (analog/digital) muuntaa vastaanotetun (jatkuva-aikaisen) analogisen signaalin digitaaliseksi ja diskreettiaikaiseksi.
2. Tämän jälkeen diskreettiaikaista digitaalista signaalia muokataan jollain järjestelmällä (esim. tietokoneella). Tätä vaihetta kutsutaan *suodattamiseksi*. Suodatuksen tavoite on muuntaa järjestelmään saapuva signaali sovellutuksen kannalta hyödyllisempään muotoon. Tämä saattaa tarkoittaa esimerkiksi:
    - Signaalissa olevan kohinan poistamista siten, että varsinainen signaali säilyy mahdollisimman hyvin.

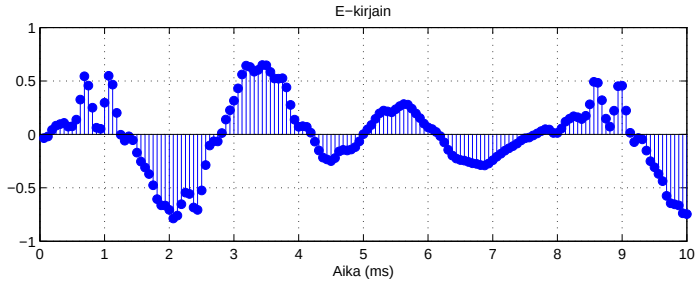
- Signaalissa olevien mielenkiintoisten piirteiden erottelua muun signaalin joukosta.

Perinteisesti suotimet ovat olleet lineaarisia niiden helpomman toteuttamisen ja analysoinnin vuoksi, mutta myös epälineaarisia suotimia on tutkittu.

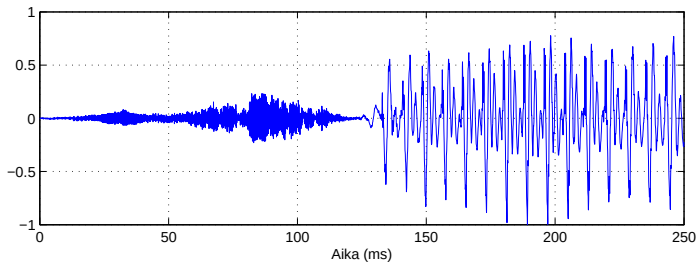
3. Suodatuksen jälkeen signaali muunnetaan takaisin analogiseksi D/A-muuntimella.

Jatkossa keskitytään pääasiassa vaiheeseen 2, suotimen suunnitteluun.

Käsiteltävä signaali voi esittää esimerkiksi ääntä, puhetta, pulssia, aivokäyrää, maanjäristystä, pörssikurseja tai mitä hyvänsä mitattavissa olevaa aikasarjaa. Alla olevassa kuvassa on viiden millisekunnin mittainen näyte puhesignaalista. Mikrofoni muuntaa mittamansa pienet ilmanpaineen vaihtelut sähköiseen muotoon, josta tietokone muuntaa ne edelleen digitaaliseen muotoon tallentamalla jännitteen hetkelliset lukuarvot 1/16000 sekunnin välein. Tässä tapauksessa *näytteenottotaajuus* on siis 16000 hertsiä.

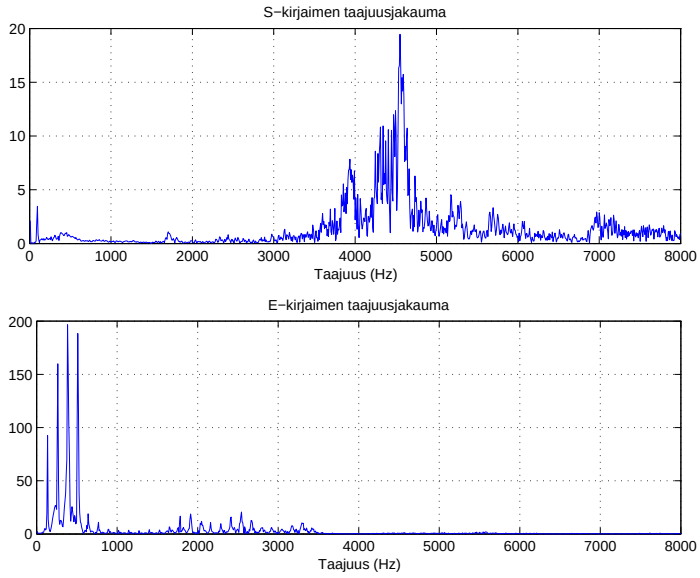


Seuraavassa kuvassa on pidempi näyte samasta signaalista. Kuvan signaali sisältää sanan "seitsemän" kaksi ensimmäistä kirjainta. S-kirjain sijaitsee alusta lukien noin 125 millisekunnin matkalla, ja seuraavassa 125 millisekunnissa on E-kirjain. Konsonantin ja vokaalin ero näkyy hyvin: soinnillisen vokaalin kohdalla on selkeä ylös-alas-värähtelykuvio ja konsonantin kohdalla lukuarvot ovat satunnaisempia.



Useille erityyppisille signaaleille on luontevaa ajatella niiden koostuvan yksittäisistä taajuuksista (yksittäisistä sinisignaaleista sopivassa suhteessa). Esimerkiksi äänisignaaleita on helpoin ymmärtää ja analysoida niiden taajuusjakauman kautta. Kuten kappaleessa 3

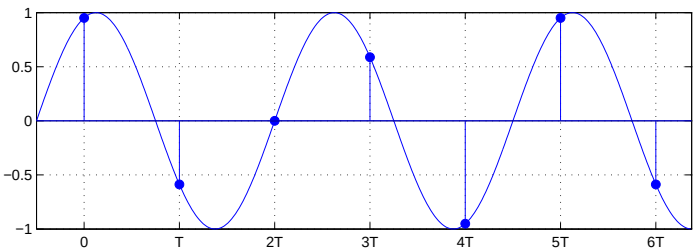
tullaan näkemään, taajuusjakauma voidaan laskea *Fourier-muunnoksen* avulla. Alla olevissa kuvissa on laskettu edellisen kuvan S- ja E-kirjainten taajuusjakaumat. Kuvista näkyy selvästi, että S-kirjaimen sisältämät taajuudet jakautuvat melko laajalle alueelle sekä melko suurillekin taajuuksille, kun E-kirjain sisältää vain yksittäisiä pieniä taajuuksia (korkeat piikit).



1.2 Näytteenottoteoreema

Edellisessä esimerkissä näytteenottotaajuus oli 16000 hertsiä eli 16 kHz. Mitä suurempi näytteenottotaajuus on, sitä pienemmät yksityiskohdat signaalista saadaan talteen. Suurempi näytteenottotaajuus vaatii kuitenkin enemmän tilaa, joten taajuutta ei kannata nostaa liian suureksi. Mistä siis tiedetään mikä on riittävä näytteenottotaajuus tietylle signaalille? Tähän kysymykseen vastaa *näytteenottoteoreema, engl. sampling theorem*.

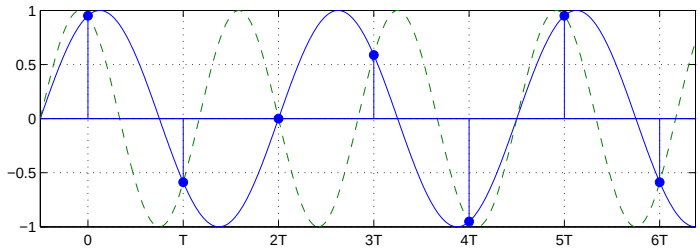
Jatkuva-aikaista signaalia näytteistettäessä siitä otetaan näytteitä ajanhetkillä 0, T, $2T$, $3T, \ldots$ ja vain signaalin näillä hetkillä saamat arvot talletetaan (mustat ympyrät alla olevassa kuvassa). Jos siis jatkuvasta signaalista käytetään merkintää $x_c(t)$, missä $t \in \mathbf{R}$, niin näytteistyksen tuloksena saadaan lukujono $x(n)$, jolle on voimassa ehto $x(n) = x_c(nT)$ $(n = 0, 1, 2, \ldots)$.



Vakio T ilmoittaa siis kuinka monta sekuntia on kahden peräkkäisen näytteen väli. Useimmiten sama asia ilmaistaan sanomalla montako kertaa sekunnissa näytteitä otetaan.

Tämän suureen nimi on *näytteenottotaajuus* (sampling frequency) ja se on vakion T käänteisluku, $F_s = \frac{1}{T}$. Jos näytteenottotaajuus on liian pieni (ja siis näytteiden väli T liian suuri), tapahtuu *laskostumista* eli *alinäytteistymistä* (aliasing).

Laskostuminen tulee ilmi alla olevasta kuvasta. Kuvan kahdella sinisignaalilla on samat näytearvot, koska näytteenottotaajuus on liian pieni suurempitaajuuksiselle signaalille (katkoviiva). Muunnettaessa näin näytteistettyä signaalia takaisin analogiseksi lopputuloksena on pienempitaajuinen sinisignaali (yhtenäinen viiva). Sanotaan, että suurempi taajuus laskostuu pienemmän taajuuden päälle.

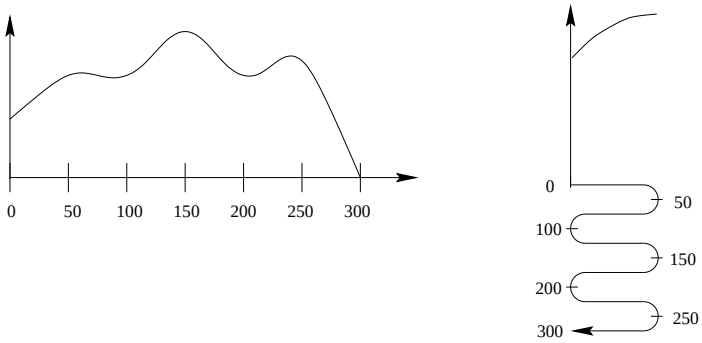


Pohdittaessa riittävää näytetaajuutta edellisen esimerkin sinisignaalille tuntuu, että kaksi näytettä jaksoa kohti saattaisi riittää. Tällöin nimittäin tallennettaisiin signaalin suurin ja pienin arvo ja näiden avulla voitaisiin interpoloida muut lukuarvot huippuarvojen välille. Luonnollisesti mikä tahansa tätä suurempi taajuus käy yhtä hyvin. Näytteenottoteoreema kertoo, että tämä arvaus pitää paikkansa.

> *Jatkuva-aikainen signaali voidaan muodostaa uudelleen näytearvoistaan, jos näytteenottotaajuus* F_s *on vähintään kaksi kertaa niin suuri kuin signaalin sisältämä suurin taajuuskomponentti.*

Jos edellinen ehto ei ole voimassa, täytyy taajuutta $F_s/2$ suuremmat taajuudet leikata pois jollain analogisella järjestelmällä laskostumisen estämiseksi. Taajuudesta $F_s/2$ käytetään nimitystä Nyquistin taajuus (Nyquist frequency, Nyquist rate).

Näin ollen enintään 300 kHz taajuuksia sisältävä signaali vaatii järjestelmän, jonka näytteenottotaajuus on vähintään 600 kHz. Jos näin ei ole käy kuten alla olevassa kuvassa. Kuvat esittävät eräiden signaalien spektrejä, jotka kertovat, kuinka paljon kutakin taajuutta on mukana signaalissa. Kuvan alkuperäisessä jatkuvassa signaalissa on taajuuksia 300 kHz asti (vasemmalla), jolloin 100 kHz taajuudella näytteistettäessä yli 50 kHz taajuudet summautuvat alemmille taajuuksille laskostuneen x-akselin mukaisesti (oikealla).



Lopputuloksessa esimerkiksi taajuudella 25 kHz on usean taajuuskomponentin summa: 25 kHz, 75 kHz, 125 kHz, 175 kHz, 225 kHz ja 275 kHz. Toisaalta 10 kHz taajuuden kohdalla ovat seuraavat taajuudet summautuneena: 10 kHz, 90 kHz, 110 kHz, 190 kHz, 210 kHz ja 290 kHz. Kyseisten kuuden komponentin summasta ei voida palauttaa enää alkuperäistä signaalia.

Paras tulos esimerkin 100 kilohertsin näytteenottotaajuudella saadaan poistamalla yli 50 kilohertsin taajuudet ennen näytteenottoa. Vaikka taajuudet 50 kHz – 300 kHz menetetäänkin, saadaan edes taajuudet 0 kHz – 50 kHz tallennettua alkuperäisessä muodossaan.

1.3 Digitaalisen signaalinkäsittelyn etuja ja haittoja jatkuva-aikaisiin suodattimiin nähden

Analogisia suotimia on elektroniikan alalla tutkittu jo kauan. Nämä kootaan elektronisista komponenteista ja ne poimivat tyypillisesti tietyt taajuudet signaalista ja poistavat muut. Herää kysymys, miksi sama pitäisi tehdä digitaalisesti?

Digitaalisista suotimista on helppo tehdä tarkkoja, ja niiden ominaisuudet pysyvät samoina koko käyttöajan. Digitaalisten suodinten ominaisuudet määräytyvät niiden kertoimien kautta, eivätkä tietokoneohjelman kertoimet muutu esimerkiksi ajan myötä tai lämpötilan vaihdellessa. Tarkkuus saadaan myös helposti paremmaksi lisäämällä laskentatehoa ja laskentatarkkuutta. Analogisistakin järjestelmistä voidaan toki tehdä yhtä tarkkoja ja ominaisuutensa säilyttäviä, mutta tällöin on käytettävä kalliimpia ja laadukkaampia komponentteja. Usein sanotaankin, että digitaalinen CD-soitin toi HIFI-laadun tavallisen kuluttajan ulottuville, kun analogisilla laitteistoilla se oli ainoastaan varakkaiden saatavilla.

Digitaalisilla suotimilla on useita teorian kannalta hyviä ominaisuuksia. Niiden avulla voidaan esimerkiksi toteuttaa täysin *lineaarivaiheinen* suodin, mikä on mahdotonta analogisen suotimen avulla. Lineaarivaiheisuus tarkoittaa sitä, että kaikki signaalin sisältämät taajuudet viivästyvät yhtä paljon. Tähän käsitteeseen tutustutaan lähemmin kappaleessa 5.1.1. Lisäksi digitaaliset suotimet toimivat kuin tietokoneohjelmat, joten niihin voidaan lisätä monimutkaisiakin rakenteita, joita analogisilla järjestelmillä on mahdoton toteuttaa.

Tärkein syy digitaalisten suodinten käyttöön analogisten komponenttien sijaan on kuitenkin raha: samaa signaaliprosessoria voidaan käyttää useisiin eri sovelluksiin, jolloin sitä voidaan tuottaa suuremmissa erissä ja suuret tuotantoerät painavat prosessorien hintoja alas. Prosessoreja käyttävät yritykset puolestaan toteuttavat oman tuotteensa ohjelmistona fyysisten laitteiden sijaan. Tällöin tuotteen monistaminen on helppoa, ja sama tuote voidaan myydä useaan kertaan—aivan kuten tietokoneohjelmistotkin.

On kuitenkin tiettyjä tapauksia, joissa toimivaa analogista järjestelmää ei kannata tai ei voi korvata digitaalisella. A/D-muunninten näytteenottotaajuuden yläraja on nykyisin luokkaa 10–100 MHz, joten hyvin suuria taajuuksia sisältäviä signaaleja ei voida käsitellä diskreetin järjestelmän avulla. Radioiden ja televisioiden suurtaajuusosat toteutetaan jatkuva-aikaisten suodattimien avulla.

Toisaalta hyvin yksinkertaiset järjestelmät, jotka eivät tarvitse suurta tarkkuutta, on helpointa toteuttaa analogisilla komponenteilla. Digitaalinen järjestelmä tarvitsee aina A/D ja D/A-muuntimet sekä prosessorin. Jos tavoitteena on vain jakaa autostereoiden kaiutinsignaali kahteen eri taajuuskaistaan, ei tätä varten kannata rakentaa digitaalista järjestelmää.

1.4 Sovelluskohteita

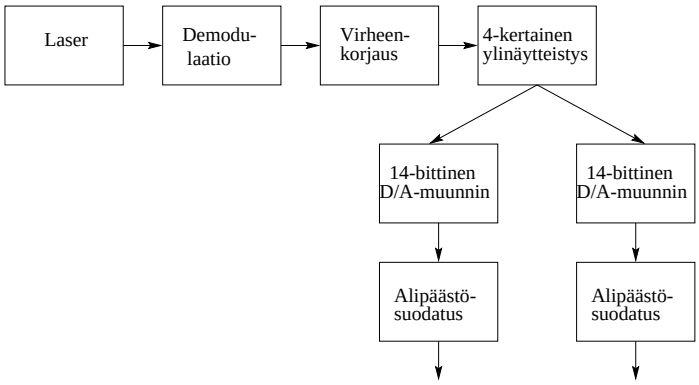
Seuraavaksi tarkastellaan muutamia tyypillisiä sovelluskohteita melko pintapuolisesti.

Telekommunikaatio: multipleksaus (multiplexing): Maailmassa arvioidaan olevan miljardi lankapuhelinta. 1960-luvulle saakka yhtä puhelinyhteyttä kohti tarvittiin yksi puhelinjohto. Digitaalisen signaalinkäsittelyn avulla johtojen määrää voidaan vähentää lähettämällä useita signaaleja yhdessä johdossa. Niin sanottu *T-carrier*-järjestelmä välittää 24 signaalia yhdessä johdossa. Jokainen signaali muunnetaan digitaaliseksi (8000 näytettä/s) käyttäen 8-bittistä esitystä. Kukin linja kuluttaa siis 64000 bittiä sekunnissa, ja kaikki 24 kanavaa kuluttavat 1.536 megabittiä sekunnissa. Tämä määrä voidaan helposti välittää puhelinkeskusten välisillä linjoilla.

Kompressio: Edellisen esimerkin bittimäärää voidaan edelleen vähentää käyttämällä pakkausta (kompressiota). Kompressioalgoritmit pyrkivät poistamaan näytteiden välillä olevaa *redundanssia*, ja saamaan näin tarvittavien bittien määrää vähennetyksi. Erilaisia pakkausmenetelmiä on lukuisia. Menetelmä saattaa olla suunniteltu pakkaamaan tehokkaasti esimerkiksi äänisignaalia, digitaalista kuvaa tai videota tai vaikkapa lääketieteessä tavattavia kolmiulotteisia tomografiakuvia. Menetelmät käyttävät hyväkseen kunkin sovelluksen ominaispiirteitä. Esimerkiksi standardisointijärjestö ISO:n pakkausstandardit MPEG-1, MPEG-2, MPEG-4 ja MPEG-7 ovat olleet runsaan huomion kohteena viime aikoina.

Kaiunkumous: Koska puhelinliikenteen signaalit kulkevat kahteen suuntaan, ongelmaksi saattaa muodostua käyttäjälle ärsyttävä kaiku. Jokainen puhuttu sana palaa pienellä viiveellä linjaa pitkin takaisin ja saa aikaan kaikua vastaavan efektin. Erityisenä ongelmana tämä on kaiutinpuhelinta käytettäessä, mutta myös tavallisissa puhelimissa. Ongelma voidaan poistaa riittävän tehokkaasti adaptiivisen signaalinkäsittelyn menetelmin.

CD-soitin: Viime aikoina musiikkiteollisuus on siirtynyt käyttämään digitaalista CD-levyä perinteisten vinyylilevyjen asemesta. Alla oleva yksinkertaistettu kaavio tyypillisestä CD-soittimesta sisältää lukuisia kohteita, joissa käytetään DSP:tä.



Puheentunnistus: Puheentunnistukselle voidaan helposti keksiä lukuisia sovelluksia, mutta valitettavasti nykyinen teknologia ei kykene tehtävään täysin aukottomasti. Yleisimmät lähestymistavat jakavat ongelman kahteen osaan: piirteiden erotteluun (feature extraction) ja piirteiden vertailuun (feature matching). Erotteluvaiheessa puhe

jaetaan erillisiin sanoihin (tai vaikkapa äänteisiin) ja verrataan näitä aiemmin kuultuihin sanoihin. Vertailuvaiheessa saattaa tulla ongelmia jos puhuja ei ole sama kuin opetusvaiheessa.

Tietokonetomografia: Röntgenkuvissa tavallinen ongelma on, että kaksiulotteisesta kuvista ei voida tunnistaa päällekkäisiä elimiä. Lisäksi röntgenkuvista käy ilmi potilaan anatomia, ei fysiologia. Toisin sanoen, ainoastaan kehon rakenne on nähtävissä, ei sen toimintaa. Näin ollen kuolleen ihmisen röntgenkuva vastaa miltei täysin elävän ihmisen röntgenkuvaa. Näitä ongelmia on ratkaistu 1970-luvulta alkaen tietokonetomografian keinoin. Perinteisessä tietokonetomografiassa otetaan röntgenkuvia useista suunnista ja saadaan kolmiulotteinen malli. Röntgensäteiden vaarallisuudesta johtuen myöhemmin on kehitelty muita menetelmiä, kuten PET (positron emission tomography) sekä MRI (magnetic resonance imaging). Näistä edellinen käyttää positronisädettä kuvan muodostamiseen ja jälkimmäinen muodostaa kuvan voimakkaiden sähkömagneettien avulla. Signaalinkäsittelyä käytetään pääasiassa kuvan kokoamisessa yksittäisten säteiden muodostamasta datasta.

Harjoitustehtäviä

1.1. *(Matlab)* Tutustutaan Matlabin käyttöön piirtämällä samanlaiset kuvaajat kuin kappaleessa 2.1.

(a) Sijoita yksikkönäyte vektoriin `delta` sivulla 12 annetulla komennolla. Koska kuvan vaaka-akseli sisältää pisteet $-7, -6, -5, \ldots, 7$, luodaan myös vektori `n`, joka sisältää nämä pisteet: `n = -7:7;`. Piirrä tämän jälkeen yksikkönäytteen kuvaaja komennolla `stem(n, delta);`.

(b) Piirrä yksikköaskel $u(n)$ (s. 12) samalla tavalla.

(c) Piirrä ramppisignaali $nu(n)$ (s. 13) samalla tavalla.

1.2. *(Matlab)*

(a) Luo vektori **t**, joka sisältää arvot $1.00, 1.01, 1.02, 1.03, \ldots, 1.98, 1.99, 2.00$. *Vihje:* `help colon`

(b) Luo vektori **x**, joka sisältää funktion $x(t) = \cos(2\pi t) + 2\sin(4\pi t)$ evaluoituna vektorin **t** määräämissä pisteissä.

(c) Tulosta edellisen kohdan funktion mukainen käyrä välillä $[1, 2]$. *Vihje:* `help plot`

(d) Tulosta edellisen kohdan funktion itseisarvon mukainen käyrä välillä $[1, 2]$. *Vihje:* `help abs`

(e) Tulosta molemmat käyrät samaan kuvaan. *Vihje:* `help subplot`. Nimeä kuva. *Vihje:* `help title`.

1.3. *(Matlab)*

(a) Luo 10×10 matriisi:

$$A = \begin{pmatrix} 1 & 2 & \cdots & 10 \\ 11 & 12 & \cdots & 20 \\ \vdots & \vdots & \ddots & \vdots \\ 91 & 92 & \cdots & 100 \end{pmatrix}$$

Vihje: `help reshape, help transpose`

(b) Korota matriisin A jokainen alkio kolmanteen potenssiin *Vihje:* `help power`

(c) Laske matriisin A kolmas potenssi A^3 *Vihje:* `help mpower`

(d) Luo satunnaisluvuista koostuva 10×10-matriisi ja sijoita se muuttujaan B. *Vihje:* `help rand`

(e) Laske edellisen kohdan matriisin käänteismatriisi ja sijoita se muuttujaan C. Laske vielä matriisien B ja C matriisitulo. *Vihje:* `help inv`

1.4. *(Matlab)*

(a) Luo 50-alkioinen normaalisti jakautunut satunnaisvektori (`help randn`) ja tulosta sen itseisarvojen (`help abs`) kuvaaja 'diskreetisti' komennolla `stem`.

(b) Luo funktio nimeltä *summaus* (tiedostoon nimeltä `summaus.m`), joka saa syötteenä vektorin **x** ja palauttaa **x**:n alkioiden summan ja niiden neliösumman *Vihje:* `help function, help sum`.

(c) Testaa funktiota (a)-kohdan vektorilla.

1.5. Analoginen signaali koostuu yksittäisestä siniaallosta, jonka taajuus on 1000 Hz. Signaalista otetaan näytteitä 0.0006 sekunnin välein.

(a) Tapahtuuko laskostumista?

(b) Jos vastauksesi on myönteinen, miksi taajuudeksi em. sinisignaali tulkitaan, ts. mille taajuudelle se laskostuu?

(c) Mikä olisi riittävä näytteenottotaajuus laskostumisen estämiseksi?

1.6. Analoginen (jatkuva-aikainen) signaali on muotoa $x(t) = 2\cos(120\pi t) - 0.7\sin(30\pi t) + \cos(200\pi t) + \cos(180\pi t)$, missä t on aikaa kuvaava muuttuja (sekunteina). Kuinka usein signaalista pitää ottaa näytteitä, ettei laskostumista tapahdu? *Vihje: tutki kuinka monta kertaa kukin signaalin termi värähtää ylös-alas sekunnin aikana. Näistä saat signaalin suurimman taajuuden ja edelleen näytteenottotaajuuden.*

1.7. Tarkkaan ottaen Nyquistin rajataajuus ei riitä laskostumisen välttämiseksi. Tarkastellaan tällaista tilannetta tässä tehtävässä.

(a) Signaalista $x(t) = \sin(20\pi t)$ otetaan näytteitä 0.05:n sekunnin välein alkaen hetkestä $t = 0$ s. Määritä viiden ensimmäisen näytteen arvo. Voidaanko alkuperäinen signaali rekonstruoida näistä näytearvoista?

(b) Millaiset näytteet saadaan jos näytteenotto aloitetaan hetkellä $t = 0.025$ s? Mitkä näytearvot tällöin saadaan? Voidaanko alkuperäinen signaali rekonstruoida näistä näytearvoista, vai voisivatko nämä näytteet esittää jotain muutakin samantaajuista signaalia?

1.8. *(Matlab)* Generoi yhden sekunnin mittainen signaali, jonka taajuus on 1000 Hz kun näytteenottotaajuus on 8192 Hz. Yleisesti signaali saadaan kaavasta

$$x(n) = \sin\left(\frac{2\pi n f}{F_s}\right),$$

missä f on haluttu taajuus Hertseinä ja F_s on näytteenottotaajuus Hertseinä. Muuttuja n on Matlabissa vektori, joka sisältää halutut pisteet ajassa, t.s., $(1, 2, 3, \ldots, 8192)$. Generoi myös signaalit, joiden taajuudet ovat 2000 Hz ja 3000 Hz, ja kuuntele kaikki tulokset komennolla `soundsc`. Mitä tapahtuu, kun ylität Nyquistin rajan, eli generoit signaaleja, joiden taajuudet ovat 6000, 7000 ja 8000 Hz?

1.9. *(Matlab)* Simuloidaan Matlabilla laskostumista.

Nouda tiedosto `http://www.cs.tut.fi/kurssit/SGN-1200/seiska.mat` ja lataa se Matlabiin komennolla `load`, joka lataa sen automaattisesti muuttujaan `x`. Kuuntele alkuperäinen signaali komennolla `soundsc(x, F)`, missä `F` on näytteenottotaajuus. Pudota näytteenottotaajuus 16384 Hz ensin puoleen alkuperäisestä komennolla `y=x(1:2:length(x));` Tällöin jäljelle jää vain joka toinen näyte, eli sama tulos oltaisiin saatu otettaessa alun perin näytteitä taajuudella 8192 Hz. Koska signaali todellisuudessa sisältää taajuuksia aina 8192 Hertsiin asti, laskostuvat suuret taajuudet pienten päälle. Kuuntele tulos komennolla `soundsc`. Vertaa kuulemaasi tulokseen, joka saadaan oikeaoppisesti poistamalla liian suuret taajuudet (yli 4096 Hz) ennen näytteenottotaajuuden pienentämistä (komento `decimate`).

1.10. *(Matlab)* Edellisessä tehtävässä laskostuminen ei ollut vielä kovin merkittävää, koska miesääni ei sisällä suuria taajuuksia juurikaan muualla kuin joidenkin konsonanttien kohdalla (erityisesti s-kirjaimen). Tee edellisen tehtävän testi signaalilla

`http://www.cs.tut.fi/kurssit/SGN-1200/lintu1.mat`

Näytteenottotaajuus on nyt 8192 Hz. Kuuntele alkuperäinen näyte, joka on muuttujassa `x`. Pudota näytteenottotaajuus puoleen komennolla `z=x(1:2:length(x));` ja kuuntele tulos komennolla `soundsc` (huomaa, että näytteenottotaajuus on nyt enää 4096 Hz; tämän voi antaa `soundsc`-komennolle optiona). Kiinnitä huomiota loppuosaan: alkuperäisessä näytteessä on laskeva ääni, laskostuneessa nouseva. Tulosta ruudulle molempien signaalien spektrogrammit (komento `spectrogram`).

1.11. *(Matlab)* Lataa testisignaali `seiska.wav` kurssin kotisivulta tai suoraan osoitteesta

`http://www.cs.tut.fi/kurssit/SGN-1201/seiska.wav`

Lataa se Matlabiin komennolla `wavread`, ja piirrä signaalin spektrogrammi komennolla `specgram`, ja kuuntele se komennolla `soundsc`.

Suunnittele seuraavaksi ns. ylipäästösuodin, joka poistaa signaalista matalat taajuudet. Tämä tapahtuu komennolla[1]

[1] Kappaleessa 5 selviää mitä komento itse asiassa tekee.

```
h = fir1(30, 0.3, 'high');
```

Suodata lataamasi testisignaali suunnittelemallasi suotimella. Komento on

```
y = filter(h, 1, x);
```

missä `x` on muuttuja johon latasit testisignaalin.

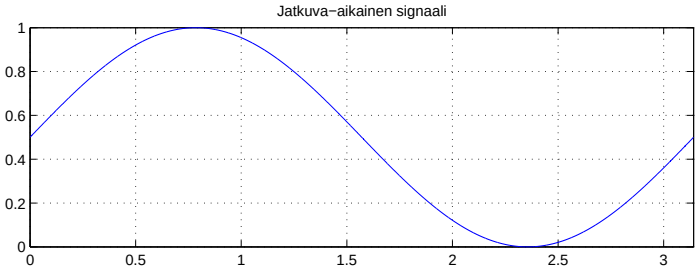
Kuuntele signaali `y` ja piirrä sen spektrogrammi ruudulle. Poistuivatko matalat taajuudet?

Luku 2

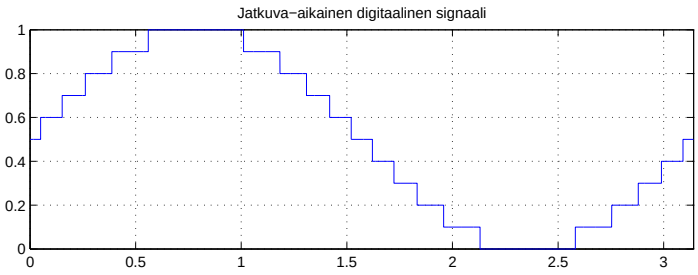
Signaalit ja järjestelmät

Edellisessä kappaleessa näimme jo esimerkkejä digitaalisista signaaleista. Määritellään nyt käytettävät käsitteet täsmällisesti.

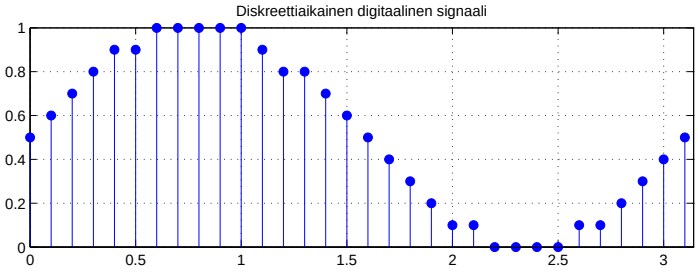
Analoginen signaali on määritelty jokaisella ajanhetkellä ja se voi saada äärettömän määrän eri arvoja (esim. väliltä $[0, 1]$, kuten alla).



Digitaalinen signaali saa vain äärellisen määrän eri arvoja.



Diskreettiaikainen signaali saa arvoja vain tietyillä ajanhetkillä.



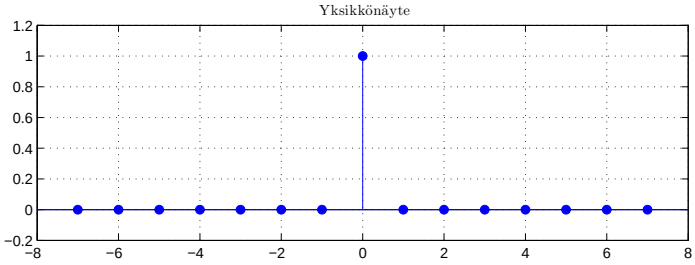
Jos esim. näytteitä otetaan taajuudella 100 Hz, signaali on määritelty ajanhetkillä $\frac{1}{100}$s, $\frac{2}{100}$s, $\frac{3}{100}$s,.... Esimerkiksi CD-soittimen signaali on 16-bittinen, eli se voi saada 2^{16} erilaista arvoa. Näytteenottotaajuus on 44100 Hz, jolloin $T = \frac{1}{44100}$ s $\approx 2.27 \cdot 10^{-5}$ s.

Matemaattisen käsittelyn helpottamiseksi lukujonoja käytetään usein diskreettiaikaisen signaalin mallina. Toisin kuin reaalimaailman signaali, lukujono on äärettömän pitkä, mutta tämä ei ole mallinnuksen kannalta ongelma.

2.1 Eräitä signaaleita

Yksikkönäyte (unit sample) eli *impulssi* $\delta(n)$ määritellään seuraavasti:

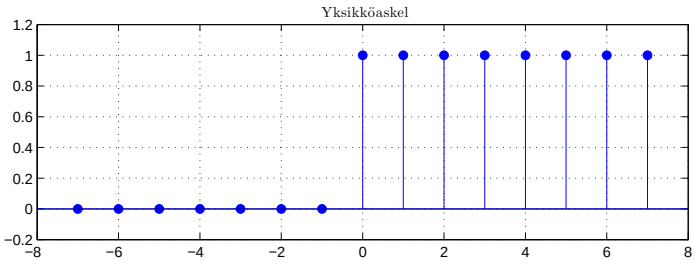
$$\delta(n) = \begin{cases} 1, & \text{kun } n = 0, \\ 0, & \text{kun } n \neq 0. \end{cases}$$



Matlab: `delta = [zeros(1,7),1,zeros(1,7)];`

Yksikköaskel (unit step) $u(n)$ määritellään seuraavasti:

$$u(n) = \begin{cases} 1, & \text{kun } n \geq 0, \\ 0, & \text{kun } n < 0. \end{cases}$$



Matlab: `u = [zeros(1,7),ones(1,8)];`

Nämä jonot voidaan esittää toistensa avulla seuraavasti:

$$\delta(n) = u(n) - u(n-1)$$

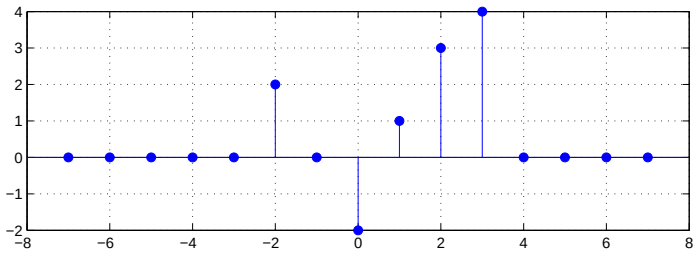
ja

$$u(n) = \sum_{k=-\infty}^{n} \delta(k).$$

Mikä tahansa jono voidaan esittää siirrettyjen ja painotettujen yksikkönäytteiden avulla seuraavasti:

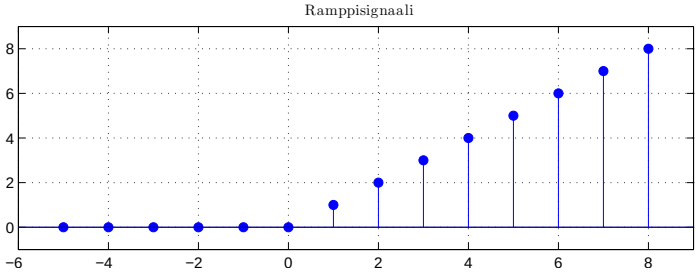
$$x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k).$$

Esimerkiksi alla olevan kuvan signaali voidaan esittää muodossa $x(n) = 2\delta(n+2) - 2\delta(n) + \delta(n-1) + 3\delta(n-2) + 4\delta(n-3)$. Kyseinen esitysmuoto on erikoistapaus *konvoluutiosta*, johon tutustutaan lähemmin kappaleessa 2.3.1.



Ramppisignaali (ramp signal) määritellään seuraavasti:

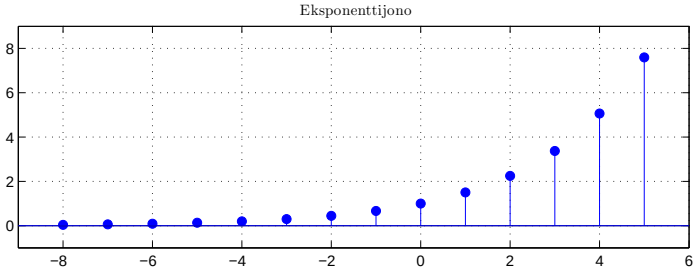
$$r(n) = nu(n) = \begin{cases} n, & \text{kun } n \geq 0, \\ 0, & \text{kun } n < 0. \end{cases}$$



Matlab: `r = [zeros(1,7),0:7];`

Eksponenttijono määritellään seuraavasti:

$$x(n) = A\alpha^n.$$



Matlab: `x = -7:7; e = 1.5.^x;`

Erikoistapaus eksponenttijonosta saadaan kun α on ei-reaalinen. Silloin $x(n)$ voidaan esittää muodossa (yksinkertaisuuden vuoksi $A = 1$)[1].

$$\begin{aligned} x(n) = \alpha^n &= |\alpha|^n e^{i\omega n} \\ &= |\alpha|^n \cos(\omega n) \\ &+ |\alpha|^n i \sin(\omega n). \end{aligned}$$

Viimeisimmästä muodosta nähdään, että eksponenttijono on näin ollen kosinijonon ja sinijonon summa. Tällä on merkitystä jatkossa, koska eksponenttijonojen algebralliset operaatiot (esim. kertolasku) ovat yksinkertaisia, mutta sini- ja kosinijonojen operaatiot ovat usein hankalia.

2.1.1 Jonojen ominaisuuksia

Diskreettiaikainen signaali on *jaksollinen*, jos on olemassa sellainen $N \in \mathbf{N}$, että

$$x(n) = x(n + N),$$

kaikilla indeksin n arvoilla. Lukua N sanotaan *jakson pituudeksi*.

Esimerkiksi kompleksinen eksponenttijono on jaksollinen, jos $|\alpha| = 1$. Samoin sinijono $x(n) = \sin(\omega n)$ on jaksollinen. Molempien jakson pituus on

$$N = \frac{2\pi}{\omega}.$$

Jonot ovat tosin määritelmän tiukassa mielessä jaksollisia vain jos näin saatu N on kokonaisluku.

Jos $\mathcal{S}$ on kaikkien signaalien joukko, niin kuvausta $\mathcal{F} : \mathcal{S} \mapsto \mathcal{S}$ sanotaan diskreetiksi järjestelmäksi tai suotimeksi. Kuvauksen $\mathcal{F}$ argumenttia kutsutaan *sisäänmenojonoksi* tai *herätteeksi* ja sen palauttamaa jonoa kutsutaan *ulostulojonoksi* tai *vasteeksi*.

Esimerkiksi yhtälö $y(n) = x(n - 10)$ määrittelee suotimen, joka viivästää signaalia 10 askelta. Toinen esimerkki laskee keskiarvon:

$$y(n) = \frac{1}{5}(x(n) + x(n-1) + x(n-2) + x(n-3) + x(n-4))$$

tai yleisemmin:

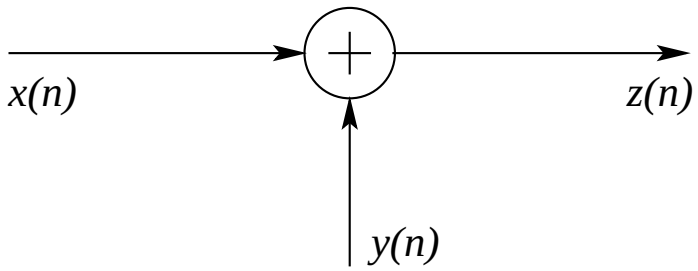
$$y(n) = \frac{1}{2K+1} \sum_{k=0}^{2K} x(n-k).$$

2.1.2 Perusoperaatiot jonoille

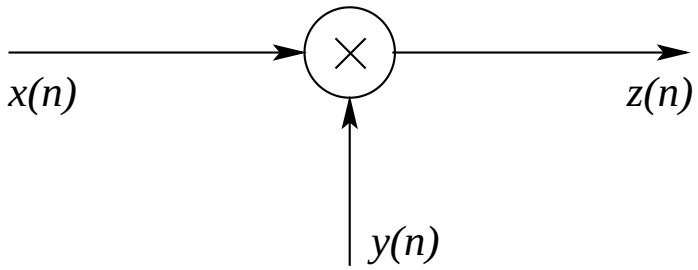
Perusoperaatiot lukujonoille (signaaleille) on esitetty seuraavassa. Mukana ovat myös operaatioille lohkokaavioissa käytettävät symbolit.

[1] Imaginääriyksiköstä käytetään merkintää $i = \sqrt{-1}$.

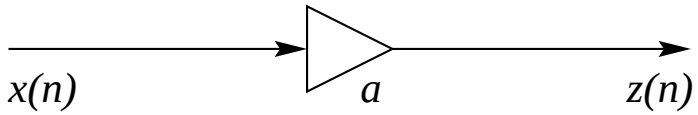
Yhteenlasku: $z(n) = x(n) + y(n)$



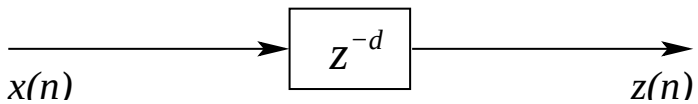
Kertolasku: $z(n) = x(n) \cdot y(n)$



Vakiolla kertominen: $z(n) = ax(n)$



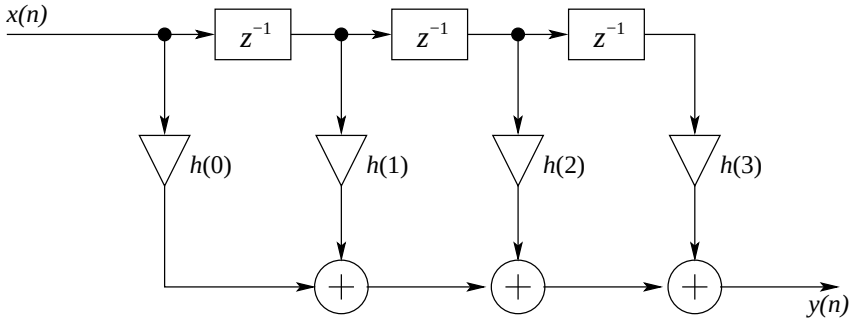
Viive: $z(n) = x(n - d)$



Edellä olleita operaatioita voidaan yhdistellä ja muodostaa näin monimutkaisempia järjestelmiä. Yksi esimerkki on järjestelmä

$$y(n) = \sum_{k=0}^{3} h(k)x(n-k),$$

jonka lohkokaavio on alla.



2.2 Diskreettien järjestelmien ominaisuuksia

Edellä olleen määritelmän mukaan kaikki operaatiot signaalien joukolla ovat suotimia. Suotimia on siis erittäin paljon, ja suurin osa niistä on käytännön kannalta tarpeettomia. Seuraavassa käydään läpi ominaisuuksia, joita suotimilla voi olla. Näin suotimia voidaan luokitella käyttötarpeen mukaan.

2.2.1 Muistittomuus

Järjestelmä on *muistiton*, jos sen ulostulo $y(n)$ riippuu vain samanaikaisesti sisääntulevasta näytteestä $x(n)$.

Esimerkiksi $y(n) = e^{x(n)}$ ja $y(n) = x(n)^2$ ovat muistittomia, mutta $y(n) = x(n-10)$ ei ole.

2.2.2 Lineaarisuus

Järjestelmä $\mathcal{F}(\cdot)$ on *lineaarinen*, jos

$$\mathcal{F}[ax_1(n) + bx_2(n)] = a\mathcal{F}[x_1(n)] + b\mathcal{F}[x_2(n)], \tag{2.1}$$

kaikilla signaaleilla $x_1(n), x_2(n)$ ja kaikilla kertoimilla a, b.

Sanallisesti sama asia voidaan ilmaista esimerkiksi seuraavasti:

- Vasteiden summa on summan vaste.
- Vaste herätteellä $ax(n)$ saadaan kertomalla herätteen $x(n)$ vaste $\mathcal{F}[x(n)]$ skalaarilla a.

Toisaalta ehto tarkoittaa myös, että:

- Skalaarilla kertominen voidaan suorittaa ennen tai jälkeen suodatuksen.
- Yhteenlasku voidaan suorittaa ennen tai jälkeen suodatuksen.

Järjestelmän lineaarisuus osoitetaan näyttämällä, että yhtälön (2.1) vasen ja oikea puoli ovat yhtäsuuret riippumatta kertoimista a ja b ja signaaleista $x_1(n)$ ja $x_2(n)$. Epälineaarisuuden osoittaminen on yleensä helpompaa: riittää löytää sellaiset kertoimet ja signaalit, että vasen ja oikea puoli ovat erisuuret edes jollakin indeksin n arvolla.

Esimerkiksi järjestelmä

$$\mathcal{F}[x(n)] = \frac{1}{2K+1} \sum_{k=0}^{2K} x(n-k).$$

on lineaarinen. Se voidaan osoittaa seuraavasti.

Olkoot a ja b mitkä tahansa reaaliluvut ja $x_1(n)$ sekä $x_2(n)$ mitkä tahansa kaksi lukujonoa. Tällöin

$$\begin{aligned} & \mathcal{F}[ax_1(n) + bx_2(n)] \\ = \; & \frac{1}{2K+1} \sum_{k=0}^{2K} (ax_1(n-k) + bx_2(n-k)) \\ = \; & a\frac{1}{2K+1} \sum_{k=0}^{2K} x_1(n-k) + b\frac{1}{2K+1} \sum_{k=0}^{2K} x_2(n-k) \\ = \; & a\mathcal{F}[x_1(n)] + b\mathcal{F}[x_2(n)]. \end{aligned}$$

Näin ollen järjestelmä on lineaarinen.

Toisaalta esimerkiksi järjestelmä

$$\mathcal{F}[x(n)] = x(n)^2$$

ei ole lineaarinen, koska yhtälö (2.1) ei ole voimassa esimerkiksi seuraavilla arvoilla: olkoon $x_1(n) = u(n)$ ja $x_2(n)$ nollasignaali. Asetetaan vielä $a = 2$ sekä $b = 0$. Valitaan vielä $n = 0$, jolloin

$$\begin{aligned} \text{vasen puoli} &= \mathcal{F}[ax_1(n) + bx_2(n)] = \mathcal{F}[2 \cdot 1] = 2^2 = 4, \\ \text{oikea puoli} &= a\mathcal{F}[x_1(n)] + b\mathcal{F}[x_2(n)] = 2 \cdot 1^2 = 2. \end{aligned}$$

Koska nämä kaksi lauseketta saavat erisuuret arvot, järjestelmä ei ole lineaarinen.

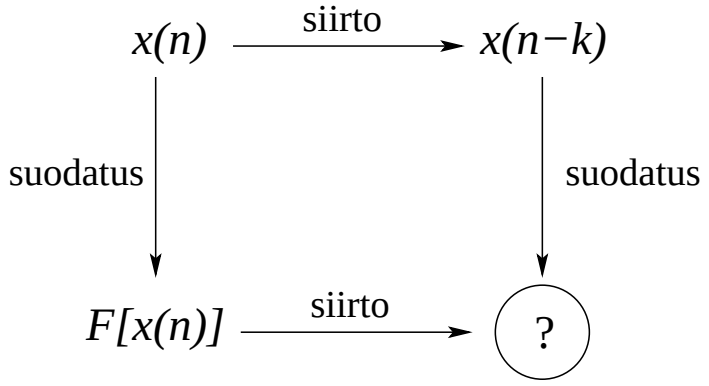
2.2.3 Siirtoinvarianssi

Merkitään $y(n) = \mathcal{F}[x(n)]$. Järjestelmä $\mathcal{F}(\cdot)$ on tällöin *siirtoinvariantti* (shift-invariant) eli *aikainvariantti* (time-invariant), jos

$$y(n-k) = \mathcal{F}[x(n-k)]$$

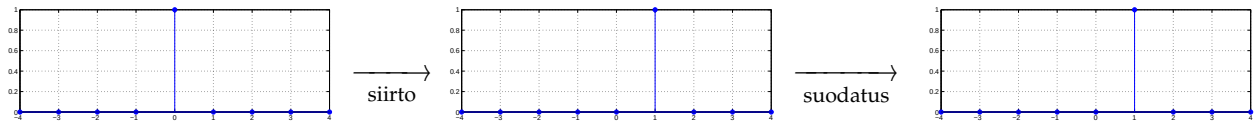
aina, kun $k \in \mathbf{Z}$. Tämä tarkoittaa toisin sanoen sitä, ettei järjestelmä ole riippuvainen ajasta, vaan siirto voidaan tehdä suodatusta ennen tai sen jälkeen.

Siirtoinvarianssia voidaan tarkastella myös seuraavan kaavion avulla:

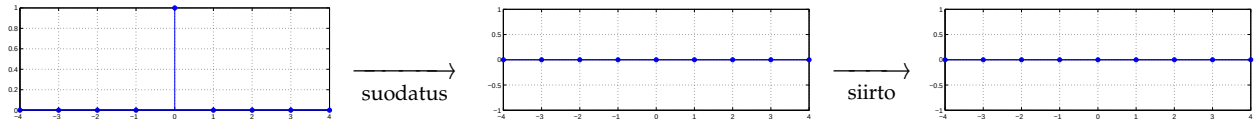


Jos tulos on sama molempia kaavion reittejä pitkin, on kyseinen järjestelmä siirtoinvariantti. Jos sen sijaan saadaan eri signaalit, niin järjestelmä ei ole siirtoinvariantti.

Esimerkiksi järjestelmä $\mathcal{F}[x(n)] = nx(n)$ ei ole siirtoinvariantti. Tämä voidaan osoittaa keksimällä sellainen signaali $x(n)$, että siirron ja suodatuksen tulos on eri kuin suodatuksen ja siirron. Valitaan $x(n) = \delta(n)$. Jos tätä siirretään yksi askel oikealle, saadaan signaali $\delta(n-1)$. Suodatettaessa tämä signaali kerrotaan kussakin pisteessä indeksillä n, eli pisteessä $n = 1$ luvulla yksi. Lopputuloksena on siis $\delta(n-1)$, kuten alla olevassa kuvassa näkyy.



Jos operaatiot tehdään toisessa järjestyksessä, tulos on toinen. Suodatettaessa signaali $\delta(n)$ kerrotaan kukin arvo indeksillä n. Pisteessä $n = 0$ oleva arvo kerrotaan siis nollalla, joten tuloksena on nollasignaali. Nollasignaalin siirto ei muuta sitä mitenkään, joten tuloksena on nollasignaali, ks. alla oleva kuva.



Koska "siirto+suodatus" tuottaa signaalin $y(n) = \delta(n-1)$ ja "suodatus+siirto" signaalin $y(n) = 0$, ei järjestelmä ole siirtoinvariantti.

Toisena esimerkkinä olkoon järjestelmä $\mathcal{F}[x(n)] = e^{x(n)}$. Osoitetaan, että $\mathcal{F}$ on siirtoinvariantti. Olkoon $y(n) = \mathcal{F}[x(n)]$ jokaisella indeksillä $n \in \mathbf{Z}$ ja olkoon $k \in \mathbf{Z}$ mielivaltainen. Määritellään lisäksi $x_1(n) = x(n-k)$. Tällöin $y(n-k) = e^{x(n-k)}$, joka on sama kuin $\mathcal{F}[x(n-k)] = \mathcal{F}[x_1(n)] = e^{x_1(n)} = e^{x(n-k)}$. Näin ollen $\mathcal{F}$ siis on siirtoinvariantti.

Siirtoinvarianssi on tärkeä lähinnä teorian kannalta. Käytännössä kaikki järkevät järjestelmät ovat siirtoinvariantteja. Ei-siirtoinvariantti järjestelmähän toimisi eri tavalla eri ajanhetkillä.

2.2.4 Kausaalisuus

Järjestelmän $\mathcal{F}(\cdot)$ sanotaan olevan *kausaalinen*, jos vaste $y(n)$ riippuu pelkästään herätteen arvoista $x(n), x(n-1), x(n-2), \ldots$ eikä arvoista $x(n+1), x(n+2), x(n+3), \ldots$ Kausaalisuus siis tarkoittaa sitä, ettei järjestelmän tarvitse tietää etukäteen, mitä arvoja heräte tulee jatkossa saamaan.

Tarkastellaan järjestelmää

$$\mathcal{F}[x(n)] = \frac{1}{5}(x(n) + x(n-1) + x(n-2) + x(n-3) + x(n-4)).$$

Vasteen laskemiseen kohdassa n tarvitaan selvästi ainoastaan herätteen arvoja $x(n), x(n-1), x(n-2), x(n-3)$ ja $x(n-4)$, joten järjestelmä on kausaalinen.

Jos sen sijaan tarkasteltavana on järjestelmä

$$\mathcal{F}[x(n)] = \frac{1}{5}(x(n+2) + x(n+1) + x(n) + x(n-1) + x(n-2)),$$

niin vasteen laskemiseen kohdassa n tarvitaan herätteen arvoja $x(n+2)$ ja $x(n+1)$, joten järjestelmä ei ole kausaalinen.

2.2.5 Stabiilisuus

Lukujonon $x(n)$ sanotaan olevan *rajoitettu*, jos on olemassa sellainen yläraja $M \in \mathbf{R}$, että

$$|x(n)| \leq M,$$

jokaisella indeksillä n. Diskreetti järjestelmä $\mathcal{F}(\cdot)$ on *stabiili*, jos jokainen rajoitettu heräte aiheuttaa rajoitetun ulostulon. Toisin sanoen, ehdosta

$$\exists M_1 \in \mathbf{R} : \forall n \in \mathbf{Z} : |x(n)| \leq M_1$$

seuraa ehto

$$\exists M_2 \in \mathbf{R} : \forall n \in \mathbf{Z} : |y(n)| \leq M_2,$$

missä $y(n) = \mathcal{F}[x(n)]$.

Tarkastellaan järjestelmää

$$y(n) = \mathcal{F}[x(n)] = \frac{1}{5}(x(n) + x(n-1) + x(n-2) + x(n-3) + x(n-4)),$$

ja osoitetaan sen olevan stabiili. Oletetaan, että $x(n)$ on rajoitettu. Silloin on olemassa sellainen reaaliluku M_1, että $|x(n)| \leq M_1$ jokaisella indeksillä n. Stabiilisuuden osoittamiseksi on näytettävä, että on olemassa sellainen $M_2 \in \mathbf{R}$, että $|y(n)| \leq M_2$ aina, kun $n \in \mathbf{Z}$. Voidaan helposti todistaa, että viiden sisäänmenonäytteen keskiarvo on aina näytteistä suurimman ja pienimmän välillä. Siis

$$-M_1 \leq \min_{n-4 \leq k \leq n} x(k) \leq y(n) \leq \max_{n-4 \leq k \leq n} x(k) \leq M_1.$$

Näin ollen vaste $y(n)$ on rajoitettu aina, kun $n \in \mathbf{Z}$, joten järjestelmä $\mathcal{F}$ on stabiili.

Sen sijaan järjestelmä $y(n) = \mathcal{F}[x(n)] = nx(n)$ on epästabiili, sillä esimerkiksi yksikköaskeljonolla $u(n)$ (joka on rajoitettu) saadaan vasteeksi jono

$$y(n) = \begin{cases} 0, & \text{kun } n < 0, \\ n, & \text{kun } n \geq 0, \end{cases}$$

joka ei selvästikään ole rajoitettu.

Myöskään järjestelmä

$$y(n) = 1.1y(n-1) + x(n)$$

ei ole stabiili, sillä syötteellä $u(n)$ vaste kasvaa rajatta. Seuraavissa kappaleissa nähdään, että tämän tyyppisten järjestelmien stabiilisuustarkastelu voidaan tehdä yksinkertaisesti tarkastelemalla järjestelmästä laskettavaa ns. napa-nollakuviota.

2.3 Lineaariset siirtoinvariantit (LTI) järjestelmät

Tässä kappaleessa tutustutaan järjestelmiin, jotka ovat lineaarisia ja siirtoinvariantteja (linear time-invariant; LTI). Tulemme havaitsemaan, että tällaisten järjestelmien suunnittelu ja toteutus on suoraviivaista ja niiden avulla voidaan toteuttaa taajuuksia muokkaavia suotimia. Tutustutaan ensin näiden järjestelmien perusominaisuuksiin.

Lineaarisuusominaisuus tarkoitti, että

1. *kertolasku voidaan suorittaa ennen suodatusta tai sen jälkeen*
2. *yhteenlasku voidaan suorittaa ennen suodatusta tai sen jälkeen*

Siirtoinvarianssi puolestaan tarkoitti, että

3. *siirto voidaan tehdä ennen suodatusta tai sen jälkeen*

Siispä LTI-järjestelmän tapauksessa on yhdentekevää, suoritetaanko kertolasku, yhteenlasku tai siirto ennen suodatusta vai sen jälkeen.

2.3.1 Impulssivaste ja konvoluutio

Mainittujen kolmen ominaisuuden ollessa voimassa järjestelmälle saadaan kätevä esitysmuoto: *konvoluutio*, jossa kaikki järjestelmän ominaisuudet määräytyvat *impulssivasteen* kautta. Impulssivaste määritellään seuraavasti.

Olkoon $\mathcal{F}(\cdot)$ lineaarinen siirtoinvariantti järjestelmä. Silloin sen *impulssivaste* (eli järjestelmän vaste impulssille) on $\mathcal{F}[\delta(n)]$ ja siitä käytetään merkintää $h(n) = \mathcal{F}[\delta(n)]$.

Näimme aiemmin, että jokainen signaali voidaan esittää siirrettyjen ja skaalattujen impulssien avulla:

$$x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k).$$

Tätä muotoa käyttäen voidaan johtaa kaikille LTI-järjestelmille yhteinen konvoluutioesitys. Jos järjestelmä $\mathcal{F}(\cdot)$ on lineaarinen ja siirtoinvariantti, niin silloin sen vaste herätteellä $x(n)$ voidaan esittää muodossa

$$\begin{aligned}
y(n) = \mathcal{F}[x(n)] = &\quad \mathcal{F}[\sum_{k=-\infty}^{\infty} x(k)\delta(n-k)] \\
= &\sum_{k=-\infty}^{\infty} \mathcal{F}[x(k)\delta(n-k)] && \text{(Lineaarisuus)} \\
= &\sum_{k=-\infty}^{\infty} x(k)\mathcal{F}[\delta(n-k)] && \text{(Lineaarisuus)} \\
= &\sum_{k=-\infty}^{\infty} x(k)h(n-k) && \text{(Siirtoinvarianssi)}
\end{aligned}$$

LTI-järjestelmän vaste mille tahansa herätteelle voidaan siis esittää yksikäsitteisesti siirrettyjen yksikkönäytteiden $\delta(n-k)$ vasteiden

$$h(n-k) = \mathcal{F}[\delta(n-k)]$$

avulla. Kyseistä esitysmuotoa kutsutaan jonojen $x(n)$ ja $h(n)$ *konvoluutioksi* tai *konvoluutiosummaksi*, ja konvoluution käsitteeseen tutustutaan seuraavassa tarkemmin.

Tarkastellaan kahta lukujonoa $x(n)$ ja $h(n)$. Lukujonojen $x(n)$ ja $h(n)$ *konvoluutiosta* käytetään merkintää $x(n) * h(n)$ ja se määritellään seuraavasti:

$$y(n) = x(n) * h(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k).$$

Seuraavassa kappaleessa osoitetaan, että konvoluutio on kommutatiivinen. Näin ollen yllä olevan määritelmän sijasta voidaan käyttää seuraavaa yhtäpitävää muotoa:

$$y(n) = h(n) * x(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k).$$

Jälkimmäinen muoto on useimmissa tapauksissa havainnollisempi.

Esimerkki: Olkoon lineaarisen järjestelmän $\mathcal{F}(\cdot)$ impulssivaste $h(n) = \delta(n) + 2\delta(n-1) + \delta(n-2)$, eli

$$h(n) = \begin{cases} 2, & \text{kun } n = 1, \\ 1, & \text{kun } n = 0, 2, \\ 0 & \text{muulloin.} \end{cases}$$

Määritetään vaste $y(n) = \mathcal{F}[x(n)]$, kun $x(n) = \delta(n) + \delta(n-1) + \delta(n-2)$ eli

$$x(n) = \begin{cases} 1, & \text{kun } n = 0, 1, 2, \\ 0 & \text{muulloin.} \end{cases}.$$

Nyt

$$y(n) = h(n) * x(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k).$$

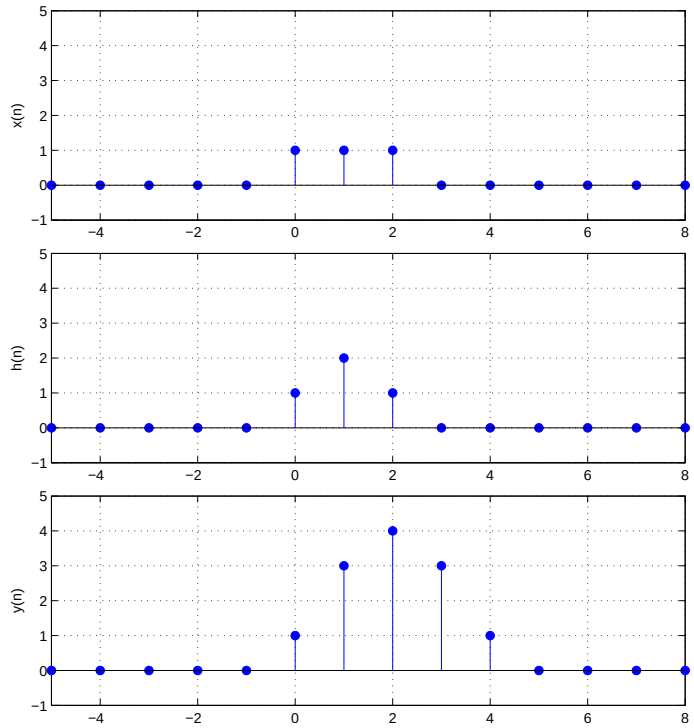
Koska $h(k) \neq 0$ vain, kun $k = 0, 1, 2$, niin $y(n)$ saa muodon

$$\begin{aligned} y(n) &= \sum_{k=0}^{2} h(k)x(n-k) \\ &= h(0)x(n) + h(1)x(n-1) + h(2)x(n-2) \\ &= x(n) + 2x(n-1) + x(n-2). \end{aligned}$$

Siis

$$\begin{aligned} y(0) &= x(0) = 1, \\ y(1) &= x(1) + 2x(0) = 1 + 2 = 3, \\ y(2) &= x(2) + 2x(1) + x(0) = 1 + 2 + 1 = 4, \\ y(3) &= 2x(2) + x(1) = 2 + 1 = 3 \text{ ja} \\ y(4) &= x(2) = 1. \end{aligned}$$

Muulloin $y(n) = 0$. Alla ensimmäisessä kuvassa on signaali $x(n)$, toisessa signaali $h(n)$ ja kolmannessa signaali $y(n)$.



Toinen esimerkki konvoluutiosta on kappaleessa 5 sivulla 76. Tässä esimerkissä tulee esille myös konvoluution vaikutus taajuusjakaumaan.

2.3.2 Konvoluution ominaisuuksia

Seuraavassa esitetään lyhyesti tärkeimpiä konvoluution ja siten myös LTI-järjestelmien ominaisuuksia.

Kommutatiivisuus (vaihdantalaki): $x(n) * h(n) = h(n) * x(n)$.

Distributiivisuus (osittelulaki): $x(n) * (h_1(n) + h_2(n)) = x(n) * h_1(n) + x(n) * h_2(n)$.

Kaskadi: Jos sellaiset järjestelmät $\mathcal{F}_1(\cdot)$ ja $\mathcal{F}_2(\cdot)$, joiden impulssivasteet ovat $h_1(n)$ ja $h_2(n)$, kytketään sarjaan, niin kokonaisuuden eli järjestelmän $\mathcal{F}_1(\mathcal{F}_2(\cdot))$ impulssivaste on $h_1(n) * h_2(n)$. Järjestelmää $\mathcal{F}_1(\mathcal{F}_2(\cdot))$ sanotaan siis järjestelmien $\mathcal{F}_1(\cdot)$ ja $\mathcal{F}_2(\cdot)$ *kaskadiksi* (cascade) ja siitä käytetään merkintää $\mathcal{F}_1 \circ \mathcal{F}_2(\cdot) = \mathcal{F}_1(\mathcal{F}_2(\cdot))$.

Kausaalisuus: Järjestelmä on kausaalinen tarkalleen silloin, kun sen impulssivaste $h(n)$ toteuttaa ehdon:

$$n < 0 \Rightarrow h(n) = 0.$$

Stabiilisuus: Järjestelmä on stabiili tarkalleen silloin, kun sen impulssivaste $h(n)$ toteuttaa ehdon

$$\sum_{k=-\infty}^{\infty} |h(k)| < \infty,$$

eli jonon $h(k)$ kaikkien alkioiden itseisarvojen summa on äärellinen.

Osoitetaan seuraavassa muutamia edellisistä kohdista.

Vaihdantalaki ($x(n) * h(n) = h(n) * x(n)$):

$$\begin{aligned} x(n) * h(n) &= \sum_{k=-\infty}^{\infty} x(k)h(n-k) \\ &\overset{k=n-t}{=} \sum_{t=-\infty}^{\infty} x(n-t)h(t) \\ &= \sum_{t=-\infty}^{\infty} h(t)x(n-t) \\ &= h(n) * x(n). \end{aligned}$$

Stabiilisuusehto: Oletetaan, että järjestelmä, jonka impulssivaste on $h(n)$, on stabiili. Silloin esimerkiksi rajoitetulla herätteellä

$$x(n) = \operatorname{sgn}[h(-n)] = \begin{cases} -1, & \text{kun } h(-n) < 0 \\ 1, & \text{kun } h(-n) \geq 0 \end{cases}$$

saadaan rajoitettu vaste, joten erityisesti $\mathcal{F}[x(n)]\Big|_{n=0}$ on äärellinen. Tarkastellaan nyt kyseistä lauseketta lähemmin.

$$\begin{aligned} \mathcal{F}[x(n)]\Big|_{n=0} &= \sum_{k=-\infty}^{\infty} h(k)x(0-k) \\ &= \sum_{k=-\infty}^{\infty} h(k)x(-k) \\ &= \sum_{k=-\infty}^{\infty} h(k)\operatorname{sgn}[h(-(-k))] \\ &= \sum_{k=-\infty}^{\infty} h(k)\operatorname{sgn}[h(k)] \\ &= \sum_{k=-\infty}^{\infty} |h(k)|. \end{aligned}$$

Näin ollen viimeinen summa on äärellinen.

Oletetaan seuraavaksi, että

$$\sum_{k=-\infty}^{\infty} |h(k)| = M_1 < \infty,$$

ja olkoon $x(n)$ mielivaltaisesti valittu rajoitettu lukujono (merkitään $|x(n)| \leq M_2$). Silloin järjestelmän vasteen itseisarvo herätteellä $x(n)$ on

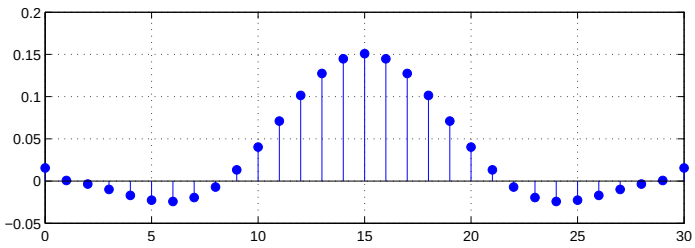
$$\begin{aligned}
|h(n) * x(n)| &= \left| \sum_{k=-\infty}^{\infty} h(k)x(n-k) \right| \\
&\overset{\Delta-ey}{\leq} \sum_{k=-\infty}^{\infty} |h(k)x(n-k)| \\
&= \sum_{k=-\infty}^{\infty} |h(k)|\,|x(n-k)| \\
&\leq \sum_{k=-\infty}^{\infty} |h(k)|\,M_2 \\
&= M_2 \sum_{k=-\infty}^{\infty} |h(k)| \\
&= M_2 M_1.
\end{aligned}$$

Näin ollen jokainen rajoitettu heräte tuottaa kyseisellä järjestelmällä rajoitetun vasteen, ja järjestelmä on stabiili.

2.3.3 FIR- ja IIR-suotimet

Konvoluution avulla voidaan siis esittää kaikki lineaariset järjestelmät. Konvoluutio käsittelee sisääntulevan signaalin $x(n)$ lukujonon $h(n)$ kertoimien määrittämällä tavalla. Lineaariset järjestelmät jaetaan kahteen luokkaan impulssivasteen pituuden mukaan.

FIR-suodin: Niin sanotun FIR-suotimen (finite impulse response) impulssivaste on äärellisen mittainen[2] (eli tietyn rajan jälkeen molempiin suuntiin impulssivaste on aina nolla). Alla oleva kuva esittää Matlabilla suunnitellun FIR-suotimen impulssivastetta. Konvoluutio kyseisen impulssivasteen kanssa poistaa signaalista tiettyä rajataajuutta suuremmat taajuudet. Impulssivasteen pituus on nyt 31. Kaikki muut impulssivasteen arvot ovat nollia.



Kun lasketaan vastetta hetkellä n, kerrotaan herätteen x uusin näyte $x(n)$ impulssivasteen nollannella termillä $h(0)$, toiseksi uusin näyte $x(n-1)$ ensimmäisellä termillä

[2]Huomaa, että tämä on eri asia kuin lause: "impulssivaste on äärellinen".

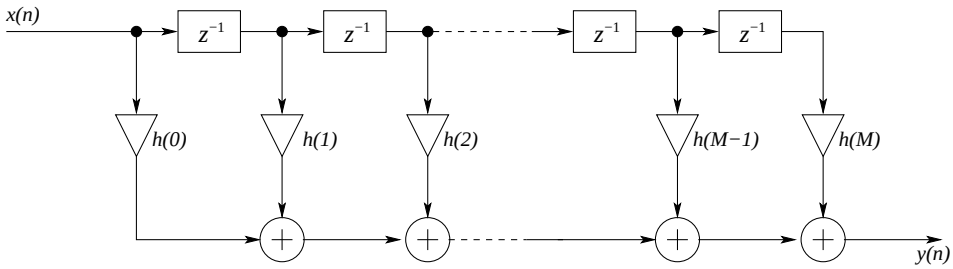
$h(1)$, näyte $x(n-2)$ termillä $h(2)$, jne. Lopuksi näin saadut tulot lasketaan yhteen. Kaavana tämä ilmaistaan muodossa

$$y(n) = \sum_{k=0}^{30} h(k)x(n-k).$$

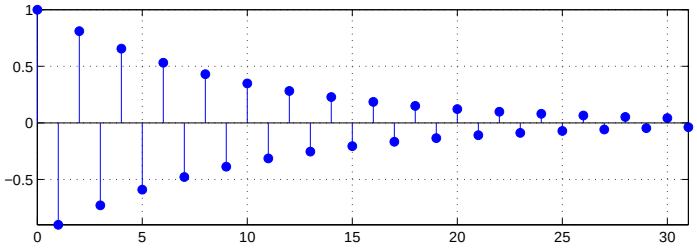
Kausaalinen FIR-suodin määritellään yleisessä muodossaan kaavalla

$$y(n) = \sum_{k=0}^{M} h(k)x(n-k).$$

Suotimen aste M määräytyy suurimman viiveen mukaan. Tämän suotimen aste on M, mutta siinä on $N = M + 1$ kerrointa. Alla on FIR-suotimen lohkokaavio.



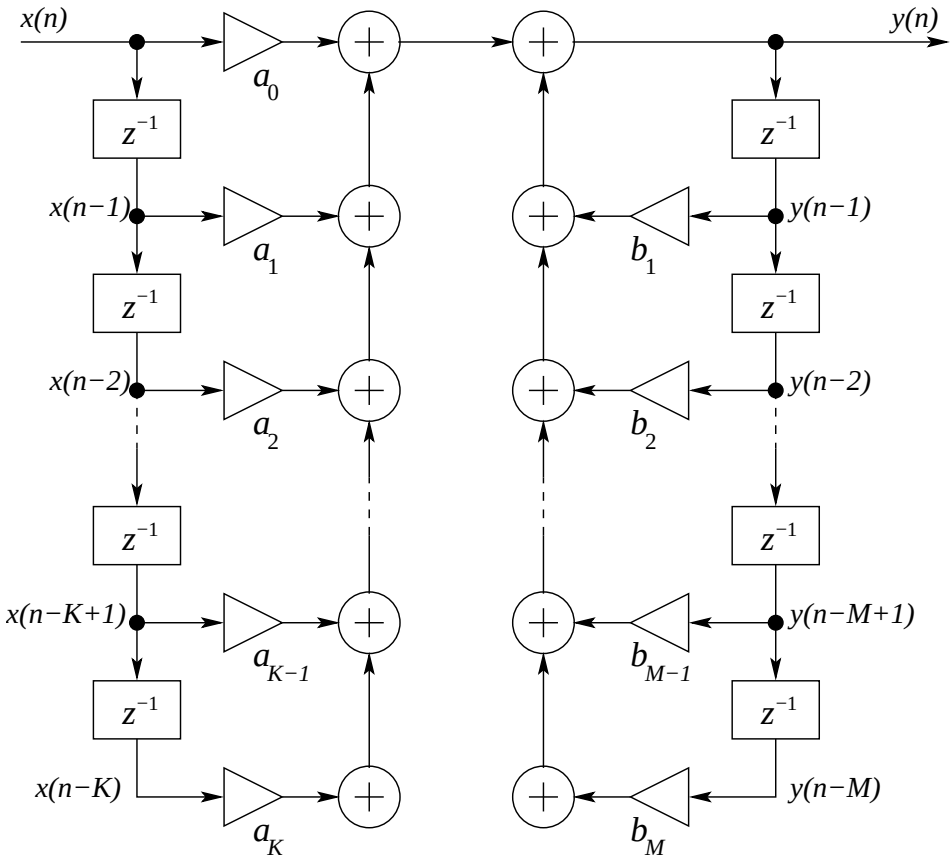
IIR-suodin: Niin sanotun IIR-suotimen (infinite impulse response) impulssivaste on äärettömän pitkä (eli ei ole rajaa, jonka jälkeen impulssivaste olisi aina nolla). Esimerkiksi impulssivaste: $h(n) = u(n)(-0.9)^n$. Kuvassa on vasteen 32 ensimmäistä termiä. Impulssivasteessa on kuitenkin äärettömän paljon nollasta poikkeavia arvoja.



IIR-suotimet voidaan toteuttaa differenssiyhtälöiden avulla. Äärettömän mittainen impulssivaste saadaan aikaiseksi takaisinkytkennällä, eli käyttämällä aiemmin laskettuja vastearvoja. Yleisessä muodossaan IIR-suodatin on operaatio, joka toteuttaa seuraavan differenssiyhtälön ulostulojonon $y(n)$ ja sisäänmenojonon $x(n)$ välillä:

$$y(n) = \sum_{k=0}^{K} a_k x(n-k) + \sum_{m=1}^{M} b_m y(n-m).$$

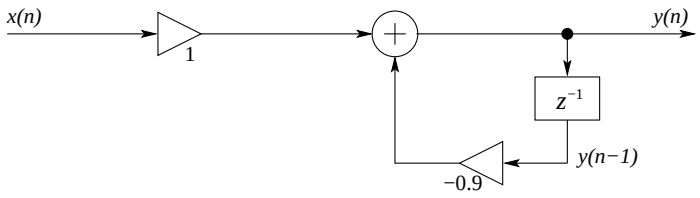
Suodin määritellään kertoimien $a_k, k = 0, 1, \ldots, K$ ja $b_m, m = 1, 2, \ldots, M$ avulla. Seuraavassa on tämä differenssiyhtälö kaaviomuodossa.



Esimerkiksi aiemmin ollut impulssivaste $h(n) = u(n)(-0.9)^n$ saadaan differenssiyhtälöstä

$$y(n) = -0.9y(n-1) + x(n),$$

kun herätteenä on impulssi $\delta(n)$. Kuva järjestelmästä on alla.



Esimerkki: Lasketaan differenssiyhtälön

$$y(n) = -0.9y(n-1) + x(n)$$

määräämän järjestelmän impulssivaste. Olkoon siis heräte $x(n)$ yksikkönäyte. Tällaisessa tehtävässä oletetaan, että $y(n) = 0$, kun $n < 0$ eli että vaste on kausaalinen. Aletaan laskea vastetta indeksistä $n = 0$ alkaen:

$$y(0) = -0.9\underbrace{y(0-1)}_{=0} + \underbrace{x(0)}_{=1} = 1.$$

Seuraavaksi asetetaan indeksiksi $n = 1$, sitten $n = 2$ ja niin edelleen:

$$\begin{aligned}
y(1) &= -0.9\underbrace{y(1-1)}_{=1} + \underbrace{x(1)}_{=0} = -0.9 \\
y(2) &= -0.9\underbrace{y(2-1)}_{=-0.9} + \underbrace{x(2)}_{=0} = (-0.9)^2 \\
y(3) &= -0.9\underbrace{y(3-1)}_{=(-0.9)^2} + \underbrace{x(3)}_{=0} = (-0.9)^3 \\
y(4) &= -0.9\underbrace{y(4-1)}_{=(-0.9)^3} + \underbrace{x(4)}_{=0} = (-0.9)^4 \\
&\vdots
\end{aligned}$$

Tässä vaiheessa havaitaan[3], että vasteen yleinen muoto on

$$y(n) = (-0.9)^n, \text{ kun } n \geq 0$$

eli

$$y(n) = u(n)(-0.9)^n,$$

joka on tietenkin sama kuin edellisen esimerkin $h(n)$.

Eri lähteissä kertoimien rooli saattaa olla toinen kuin yllä. Kyseessä on makuasia, joka saattaa kuitenkin joissain tapauksissa aiheuttaa sekaannuksia. Esimerkiksi Matlabia käytettäessä kertoimet annetaan vektoreina `b` ja `a`, missä `b` sisältää herätteen $x(n)$ kertoimet $(b_0, b_1, b_2, \ldots, b_M)$ ja vektori `a` sisältää vasteen $y(n)$ aikaisemmin laskettujen termien kertoimet $(a_0, a_1, \ldots, a_K)$; siis juuri päinvastoin kuin monisteessa (esim. s. 25). Yllä olevat kertoimet toteuttavat Matlabissa järjestelmän

$$\sum_{k=0}^{K} b_k x(n-k) = \sum_{m=0}^{M} a_m y(n-m),$$

eli

$$y(n) = \frac{1}{a_0}\left(\sum_{k=0}^{K} b_k x(n-k) - \sum_{m=1}^{M} a_m y(n-m)\right).$$

Jos $a_0 = 1$ (kuten kaikissa Matlabin suunnittelemissa suotimissa on), saadaan lauseke vastaavaan muotoon kuin sivulla 25 oleva määritelmä:

$$y(n) = \sum_{k=0}^{K} b_k x(n-k) - \sum_{m=1}^{M} a_m y(n-m).$$

Ainoana erona nyt on, että vasteen kertoimet a_m ($m = 1, 2, \ldots, M$) ovat vastakkaismerkkisiä (ks. myös tehtävä 2.19).

[3]Haluttaessa perustella täsmällisesti lausekkeen $y(n)$ yleinen muoto on käytettävä matemaattista induktiota luvun n suhteen: väitteenä on $y(n) = (-0.9)^n$. Induktion perusaskel ($y(1) = -0.9$) on voimassa. Olkoon $n \geq 1$. Tehdään induktio-oletus: $y(n) = (-0.9)^n$. Induktioväite $y(n+1) = (-0.9)^{n+1}$ nähdään seuraavasti: $y(n+1) = (-0.9)y(n) + x(n) = (-0.9) \cdot (-0.9)^n + 0 = (-0.9)^{n+1}$. Induktioperiaatteen nojalla on $y(n) = (-0.9)^n$ aina, kun $n \geq 1$. Lisäksi tapaus $n = 0$ on jo käsitelty, joten

$$\forall n \in \mathbf{N} : y(n) = (-0.9)^n.$$

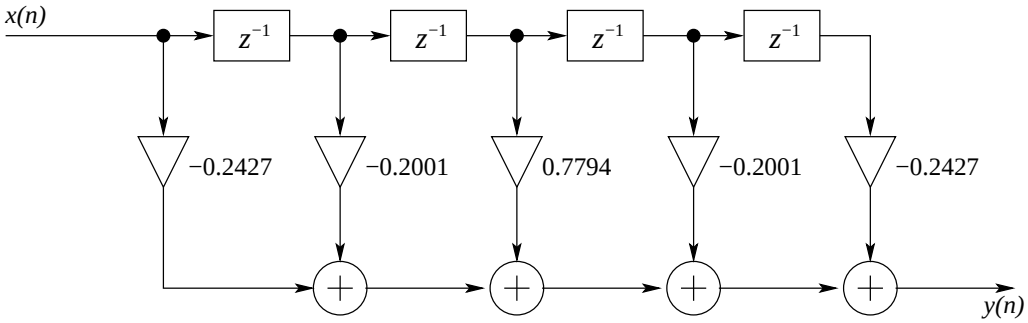
Harjoitustehtäviä

2.1. Osoita yksikköimpulssin ja yksikköaskeleen määritelmiä käyttäen luvussa 2.1 todetut yhteydet

$$\delta(n) = u(n) - u(n-1) \quad \text{ja} \quad u(n) = \sum_{k=-\infty}^{n} \delta(k)$$

Voiko jälkimmäisessä yhtälössä summan alaraja olla muukin kuin $-\infty$?

2.2. (a) Mikä differenssiyhtälö on herätteen $x(n)$ ja vasteen $y(n)$ välillä kun järjestelmän lohkokaavio on alla olevan kuvan mukainen?



(b) Piirrä lohkokaavio, kun

$$y(n) = 0.11x(n) + 0.28x(n-1) - 0.01x(n-2) - 0.52x(n-3).$$

2.3. *(Matlab)*

(a) Lataa Matlabin äänitiedosto `gong.mat` muuttujaan `y` komennolla `load gong`. Tutustu konvoluution toteuttavaan komentoon (`help conv`) ja laske signaalin `y` ja tehtävän 2.2 (a)-kohdan kertoimista muodostetun vektorin konvoluutio. Sijoita tulos muuttujaan `z`. Kuuntele alkuperäinen signaali ja vertaa sitä suodatustulokseen. Kertoimet oli valittu siten, että konvoluutiossa osa taajuuksista poistuu. Arvioi mitkä taajuudet poistettiin?

(b) Impulssivaste voi olla ihmisen suunnittelema (kuten a-kohdassa) tai mitattu esim. jossain kaiullisessa tilassa. Jälkimmäisessä tapauksessahan se mallintaa tilan akustiikkaa. Lataa kurssin sivulta tiedosto `hall.wav`, ja lataa se Matlabiin komennolla `wavread`. Tiedosto esittää ison salin impulssivastetta (ja kuulostaa siis siltä kuin käsiä olisi läpsäytetty kirkossa). Lataa myös tiedosto `seiska.mat`, ja laske sen ja lataamasi impulssivasteen konvoluutio. Kuuntele tulos ennen ja jälkeen suodatusta. Tuloksen pitäisi kuulostaa siltä kuin puhe olisi siirretty kaiulliseen tilaan.

2.4. Osoita, ettei järjestelmä

$$\mathcal{F}[x(n)] = e^{x(n)}$$

ole lineaarinen.

2.5. Onko järjestelmä

$$\mathcal{F}[x(n)] = \text{toiseksi suurin luvuista } x(n), x(n-1) \text{ ja } x(n-2)$$

lineaarinen? Miksi / miksi ei?

2.6. Osoita, että järjestelmä

$$\mathcal{F}[x(n)] = x(n) + \frac{1}{2}x(n-1) + \frac{1}{4}x(n-2)$$

on siirtoinvariantti.

2.7. Osoita, että järjestelmä

$$\mathcal{F}[x(n)] = n^2 x(n)$$

ei ole siirtoinvariantti.

2.8. Osoita, että järjestelmä

$$\mathcal{F}[x(n)] = \frac{1}{x(n)}$$

ei ole stabiili.

2.9. Osoita, että järjestelmä

$$\mathcal{F}[x(n)] = \sin(x(n))$$

on stabiili.

2.10. Osoita, että järjestelmä

$$y(n) = x(n) - 0.9y(n-1)$$

on stabiili.

2.11. *(Matlab)* Tekstissä mainitaan, ettei järjestelmä $y(n) = 1.1y(n-1) + x(n)$ ole stabiili koska syötteellä $u(n)$ vaste kasvaa rajatta. Kokeile tätä Matlabilla seuraavasti. Luo ensin pätkä signaalista $u(n)$ kuten tehtävässä 1.1 (tee kuitenkin ykkösten osuudesta hieman pidempi). Tutustu komentoon `filter` ja suodata $u(n)$ mainitulla järjestelmällä. Tulosta vaste ruudulle komennolla `stem`.

2.12. Määritellään signaali $x(n)$ ja impulssivaste $h(n)$ seuraavasti:

$$\begin{aligned} x(n) &= \delta(n) + 2\delta(n-1) - \delta(n-3) \\ h(n) &= 2\delta(n+1) + 2\delta(n-1) \end{aligned}$$

Piirrä signaalit $x(n)$ ja $h(n)$. Laske signaali $h(n) * x(n)$ ja piirrä myös se (kynällä ja paperilla).

2.13. *(Matlab)* Olkoot jonot $x(n)$ ja $h(n)$ kuten edellä. Laske konvoluutiot $h(n) * x(n)$ sekä $h(n) * h(n) * x(n)$ Matlabilla ja tulosta ne ruudulle (*komentoja:* `conv, stem`).

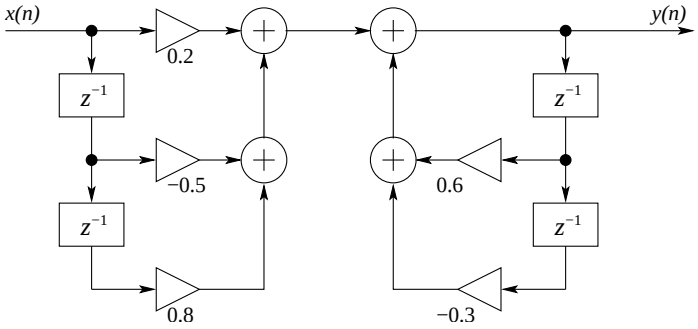
2.14. (a) Laske konvoluutio $h(n) * x(n)$, missä

$$\begin{aligned} h(n) &= \left(\frac{1}{3}\right)^n u(n), \\ x(n) &= u(n-1). \end{aligned}$$

Vihje: tarvitset geometrisen sarjan summakaavaa.

(b) Onko impulssivasteen $h(n)$ määräämä järjestelmä kausaalinen? Entä stabiili?

2.15. (a) Mitä differenssiyhtälöä alla oleva järjestelmä kuvaa?



(b) Piirrä lohkokaavio järjestelmästä

$$y(n) = x(n) + \frac{1}{2}x(n-1) - \frac{1}{4}x(n-2) + \frac{1}{6}y(n-1) + \frac{1}{8}y(n-2).$$

2.16. LTI-järjestelmän $\mathcal{F}[x(n)]$ impulssivaste on

$$h(n) = \begin{cases} 2, & \text{kun } n = 1, \\ 1, & \text{kun } n = 0, 2, \\ 0 & \text{muulloin.} \end{cases}$$

Mikä on kaskadin $\mathcal{F} \circ \mathcal{F}$ impulssivaste?

2.17. LTI-järjestelmän herätteen $x(n)$ ja vasteen $y(n)$ välillä on voimassa yhtälö

$$y(n) = 0.75y(n-1) + 0.5x(n) + 0.5x(n-1).$$

Määritä järjestelmän impulssivaste.

2.18. Merkitään LTI-järjestelmän vastetta yksikköaskeleelle seuraavasti: $z(n) = \mathcal{F}[u(n)]$. Määritä impulssivaste $h(n)$, kun askelvaste (step response) $z(n)$ tiedetään.[4]

2.19. *(Matlab)* Matlab suodattaa vektorissa `x` olevan signaalin komennolla

```
y=filter(b,a,x);
```

[4]Näennäisestä esoteerisuudestaan huolimatta tällä lauseella on joskus käyttöä. Generoimalla herätteeksi yksikköaskel saadaan vastetta tarkastelemalla selville myös impulssivaste. Yksikköaskel on lisäksi usein helpompi generoida kuin impulssi.

missä vektori `b` sisältää herätteen $x(n)$ kertoimet ja vektori `a` sisältää vasteen $y(n)$ aikaisemmin laskettujen termien kertoimet. Huomaa sivun 27 kommentti kertoimien esitystavasta Matlabissa.

(a) Muodosta impulssisignaali $\delta(n)$ seuraavasti: `delta=[1,zeros(1,127)];` Suodata tämä suotimella, jonka differenssiyhtälö on

$$y(n) = -0.9y(n-1) + x(n)$$

ja tulosta tulossignaali ruudulle.

(b) Suodata impulssi myös tehtävän 2.15 a-kohdan järjestelmällä ja tulosta tulossignaali ruudulle.

(c) Suodata impulssi järjestelmällä

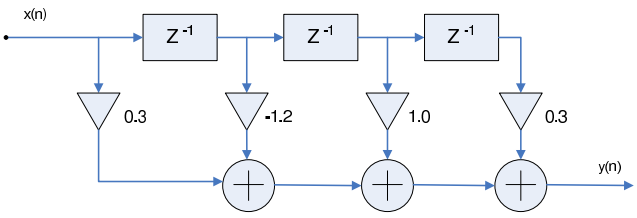
$$y(n) = x(n) + 0.5x(n-1) + 0.25x(n-2) + 0.5y(n-1) + 0.6y(n-2)$$

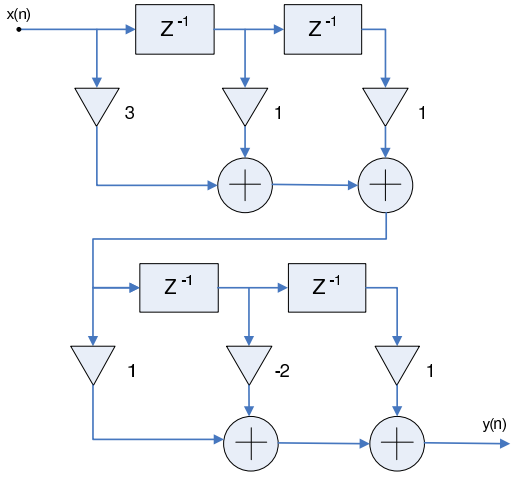
ja tulosta tulossignaali ruudulle.

Yksi edellämainituista järjestelmistä ei ole stabiili. Osaatko sanoa impulssivasteiden perusteella mikä?

2.20. *(Matlab)* Yritä saada edellisessä tehtävässä saamasi impulssivasteet Matlabin valmiilla `impz`-komennolla. Parametreina annetaan vektorit `a` ja `b`.

2.21. Mitä järjestelmiä alla olevien kuvien lohkokaaviot esittävät? Vastaus yhtälönä $x(n)$:n ja $y(n)$:n välillä. Oikeanpuoleisen järjestelmän hahmottamisessa voi auttaa luentomonisteen sivu 22.

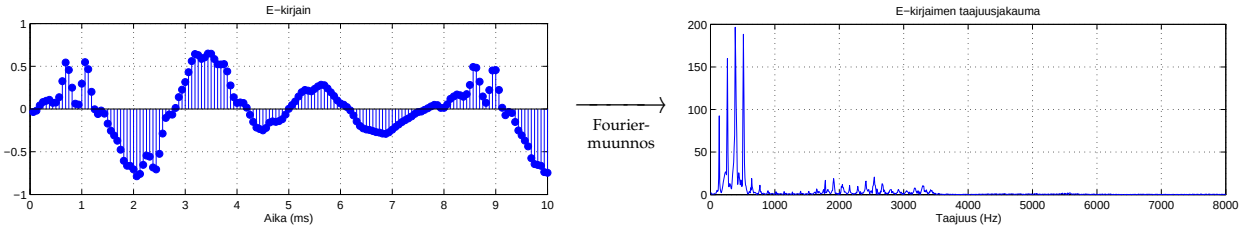




Luku 3

Fourier-muunnos

Aiemmissa kappaleissa on jo sivuttu taajuuden käsitettä. Fourier-muunnos liittää signaalin näytearvot sen sisältämiin taajuuksiin ja vastaa kysymykseen: "kuinka paljon signaalissa on kutakin taajuutta". Muunnoksen tuloksena saadaan signaalin taajuusjakauma. Alla olevassa kuvassa on vasemmalla eräs testisignaali ja oikealla sen Fourier-muunnos, josta näkyy selvästi mikä on äänisignaalin hallitseva taajuus.



Fourier-muunnos voidaan tehdä jatkuvalle tai diskreetille signaalille. Riippuen signaalin jaksollisuudesta tuloksena on vektori, diskreetti signaali tai jatkuva-aikainen signaali, jotka kuvaavat signaalin sisältämiä taajuuksia. Alla oleva taulukko esittää neljää eri muunnostyyppiä.

Muunnettavan signaalin tyyppi	*Jatkuva-aikainen*	*Diskreettiaikainen*
Ei-jaksollinen	Tuloksena on ei-jaksollinen jatkuva-aikainen signaali. Käytetään nimitystä **Fourier-muunnos**. (jatkuva $\rightarrow$ jatkuva)	Tuloksena on jaksollinen jatkuva-aikainen signaali. Käytetään nimitystä **diskreetti-aikainen Fourier-muunnos (DTFT)**. (diskreetti $\rightarrow$ jatkuva)
Jaksollinen	Tuloksena on ei-jaksollinen diskreettiaikainen signaali. Käytetään nimitystä **Fourier-sarja**. (jatkuva $\rightarrow$ diskreetti)	Tuloksena on jaksollinen diskreettiaikainen signaali. Käytetään nimitystä **diskreetti Fourier-muunnos (DFT)**. (diskreetti $\rightarrow$ diskreetti)

Fourier-muunnos esittää muunnettavan signaalin kompleksisten eksponenttifunktioiden $(\ldots, e^{-3it}, e^{-2it}, e^{-it}, e^{0}, e^{it}, e^{2it}, e^{3it}, \ldots)$ avulla. Eulerin kaavan mukaan kukin näistä

funktioista on itse asiassa sinin ja kosinin summa:

$$e^{ikt} = \cos(kt) + i\sin(kt).$$

Näin ollen kompleksinen eksponenttifunktio koostuu sinin ja kosinin värähtelyistä, missä k määrää taajuuden. Kumpi tahansa muoto on käyttökelpoinen, mutta käytännössä eksponettifunktion kanssa on paljon helpompi työskennellä (esim. kahden kompleksisen eksponettifunktion tulo: $e^{ikt}e^{imt}$ on helpompi sieventää kuin $(\cos(kt)+i\sin(kt))(\cos(mt)+i\sin(mt))$).

Fourier-analyysin ideana on selvittää voidaanko tietty signaali (funktio, lukujono) esittää kompleksisten eksponenttifunktioiden painotettuna summana (eli lineaarikombinaationa). Jos tämä on mahdollista, täytyy vielä ratkaista kunkin eksponenttifunktion painokerroin. Tästä painokertoimesta voidaan sitten päätellä kuinka paljon signaalissa on kyseistä taajuutta. Lisäksi siitä käy ilmi missä vaiheessa kyseinen taajuuskomponentti on.

Digitaalisen signaalinkäsittelyn sovelluksissa käytetään pääasiassa diskreettiä Fourier-muunnosta signaalin taajuussisältöä analysoitaessa. Syynä on se, että laskenta on tällöin puettavissa matriisikertolaskun muotoon ja laskut ovat näin ollen äärellisiä ja helppoja suorittaa tietokoneella. Muut muunnostyypit ovat teoreettisia työkaluja joita käytetään esimerkiksi suodinsuunnittelussa. Näiden laskukaavat sisältävät äärettömiä summia sekä integraaleja, jotka ovat tietokoneella laskettaessa hankalia käsiteltäviä. Näin ollen ne soveltuvat huonommin taajuuksien automaattiseen analyysiin. Sen sijaan ne ovat korvaamaton työkalu suunnitteluvaiheessa.

Sovellusten kannalta tärkein muunnostyyppi on diskreetti Fourier-muunnos ja sen toteuttava nopea algoritmi FFT. Muun muassa ikkunamenetelmän yhteydessä tarvitaan kuitenkin diskreettiaikaista Fourier-muunnosta, joten siihenkin on syytä tutustua lyhyesti. Täydellisyyden vuoksi mainitaan myös jatkuva-aikaisten signaalien muunnokset, vaikka ei vielä tässä monisteessa tarvitakaan. Ne saattavat tulla kuitenkin myöhemmin vastaan jatkuvia signaaleja käsiteltäessä, esimerkiksi tietoliikennetekniikassa.

Jonon Fourier-muunnoksesta on tapana käyttää vastaavaa isoa kirjainta; siis esim. jonon x Fourier-muunnos on X ja jonon y Fourier-muunnos on Y. Vastaava merkintätapa on käytössä myös z-muunnoksen yhteydessä kappaleessa 4.

3.1 Fourier-muunnos (ei-jaksollinen jatkuva-aikainen signaali)

Ensimmäinen muunnostyyppi muuntaa jatkuva-aikaisen signaalin jatkuva-aikaiseksi signaaliksi, eli selvittää kuinka jatkuva funktio voidaan esittää kompleksisten eksponentiaalien painotettuna integraalina[1]. Jos muunnettavana on ei-jaksollinen jatkuva-aikainen signaali $x(t)$ ($t \in \mathbf{R}$), määritellään sen Fourier-muunnos integraalina

$$X(e^{i\omega}) = \int_{-\infty}^{\infty} x(t)e^{-i\omega t}\,dt.$$

[1]Painotettu integraali tarkoittaa integraalia, jossa integroitava funktio kerrotaan painofunktiolla ennen integrointia. Vastaava kuin painotettu summa, mutta summan tilalla integraali.

Tuloksena saadaan muuttujan $\omega \in \mathbf{R}$ funktio[2]. Fourier-muunnoksesta päästään takaisin alkuperäiseen signaaliin *käänteisellä Fourier-muunnoksella*:

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(e^{i\omega}) e^{i\omega t} \, d\omega.$$

Merkintä $X(e^{i\omega})$ saattaa näyttää hankalalta, mutta kuvastaa myöhemmin tarkasteltavaa Fourier-muunnoksen ja z-muunnoksen yhteyttä. Funktio $X(e^{i\omega})$ on nyt siis reaalimuuttujan ω funktio. Periaatteessa voitaisiin siis käyttää merkintää $X(\omega)$, mutta se saattaisi aiheuttaa sekaannuksia jatkossa.

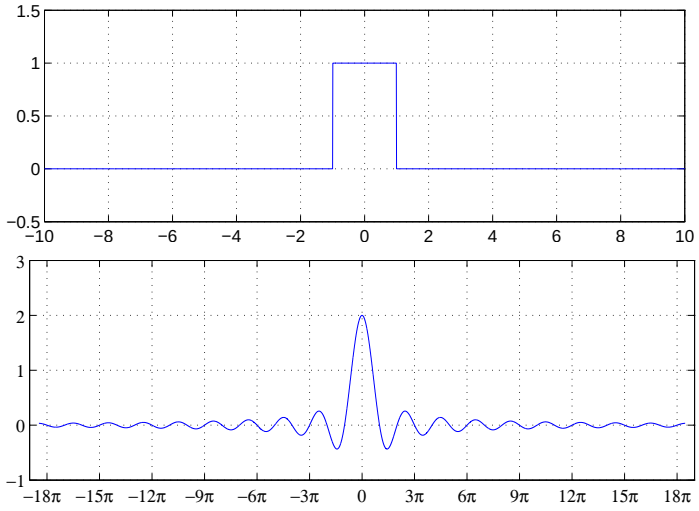
Esimerkiksi signaalin

$$x(t) = \begin{cases} 1, & \text{kun } -1 \leq t \leq 1 \\ 0, & \text{muulloin} \end{cases}$$

Fourier-muunnos on

$$X(e^{i\omega}) = \begin{cases} \frac{2\sin(\omega)}{\omega}, & \text{kun } \omega \neq 0 \\ 2, & \text{kun } \omega = 0. \end{cases}$$

Kuvaajat ovat alla olevissa kuvissa; ylemmässä on $x(t)$ ja alemmassa $X(e^{i\omega})$. Yleensä Fourier-muunnos on kompleksiarvoinen funktio, mutta tässä esimerkissä tulosfunktio sattumoisin oli reaalinen. Näin käy aina kun muunnettava signaali on symmetrinen y-akselin suhteen (ks. harjoitus 3.13). Useimmiten lopputuloksessa on kuitenkin myös kompleksiosa. Tällöin muunnos esitetään kahtena kuvaajana: ensimmäisessä on funktion itseisarvo ja toisessa sen vaihekulma.



3.2 Fourier-sarja (jaksollinen jatkuva-aikainen signaali)

Toinen muunnostyyppi muuntaa jatkuva-aikaisen signaalin diskreettiaikaiseksi, eli selvittää kuinka jaksollinen funktio on esitettävissä kompleksisten eksponentiaalien painotettuna summana. Jaksollisen signaalin tapauksessahan ei voida käyttää edellisen kappaleen

[2]Taajuusmuuttujasta on tapana käyttää kreikkalaista omega-kirjainta ω. Muuttujasta ω käytetään nimeä *kulmataajuus*.

Fourier-muunnosta, koska jaksollisen signaalin tapauksessa Fourier-muunnoksen määräävä integraali ei suppene[3]. Jaksollisen signaalin tapauksessa informaation määrä on siinä mielessä pienempi, että yksi jakso signaalista määrää sen käyttäytymisen täysin. Osoittautuukin, että jaksollisen signaalin esittämiseen riittää diskreetti määrä taajuuksia. Tuloksena on siis kokonaislukuarvoilla määritelty lukujono.

Olkoon $x(t)$ 2π-jaksollinen jatkuva-aikainen signaali (siis $x(t) = x(t+2\pi)$). Sen Fourier-sarja on diskreettiaikainen signaali

$$X(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} x(t)e^{-int}\,dt,$$

missä $n \in \mathbf{Z}$.

Jos tiedetään Fourier-sarja $X(n)$, päästään takaisin alkuperäiseen signaaliin kaavalla

$$x(t) = \sum_{n=-\infty}^{\infty} X(n)e^{int}.$$

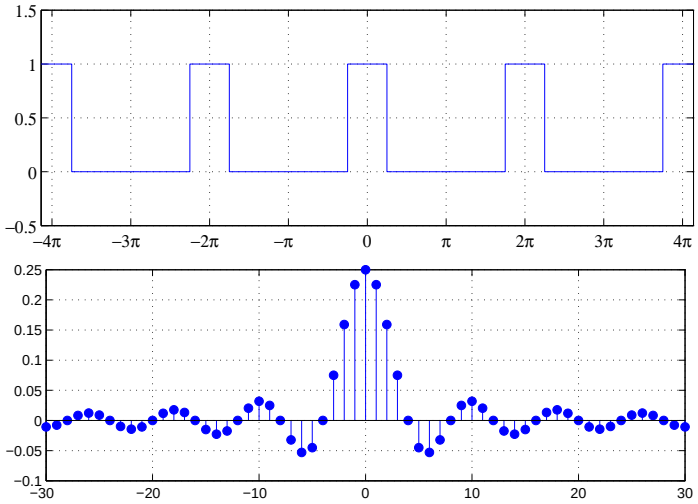
Tarkastellaan esimerkkiä jatkuvan jaksollisen signaalin Fourier-sarjaesityksestä. Määritellään funktio $x(t)$ seuraavasti:

$$x(t) = \begin{cases} 1, & \text{kun } -\frac{\pi}{4} \le t \le \frac{\pi}{4}, \\ 0, & \text{muissa välin } [-\pi,\pi] \text{ pisteissä.} \end{cases}$$

Funktio $x(t)$ on nimeltään *kanttiaalto*. Välin $[-\pi,\pi]$ ulkopuolella f on periodinen: $x(t) = x(t+2\pi)$. Tämän funktion Fourier-sarjaesitys on

$$X(n) = \begin{cases} \frac{1}{4}\frac{\sin(n\pi/4)}{n\pi/4}, & \text{kun } n \neq 0, \\ \frac{1}{4}, & \text{kun } n = 0. \end{cases}$$

Kuvaajat ovat alla; ensin on signaalin $x(t)$ ja sitten sen Fourier-sarjakehitelmän kertoimien kuvaaja. Tässäkin tapauksessa tulos on reaalinen, koska muunnettava signaali on symmetrinen y-akselin suhteen.



[3] Ainoa poikkeus on nollasignaali $x(t) \equiv 0$.

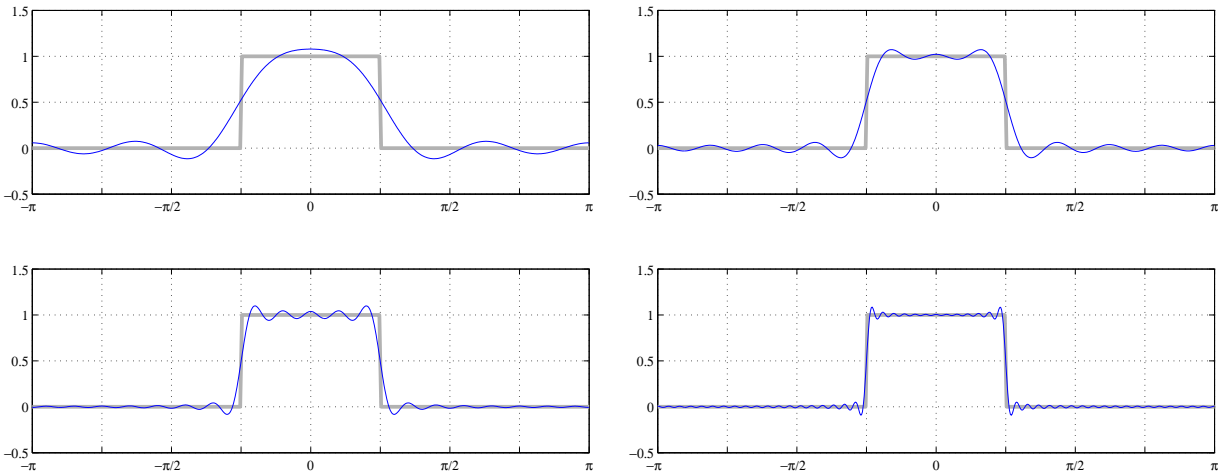
Kukin kerroin ilmoittaa nyt kuinka paljon alkuperäisessä signaalissa on kutakin taajuutta: termi $X(n)$ ilmaisee millä kertoimella signaali e^{int} on mukana muodostamassa signaalia $x(t)$. Toisin sanoen, kuinka alkuperäinen signaali voidaan esittää eritaajuisten sinien ja kosinien summana. Funktion jako eri taajuuksiin onkin Fourier-analyysin tärkeimpiä käyttökohteita.

Fourier-sarjaa voidaan käyttää hyväksi mm. generoitaessa sinien ja kosinien avulla monimutkaisempia aaltomuotoja. Sähköteknisissä laitteissa eritaajuuksiset siniaallot ovat generoitavissa melko luontevasti, mutta esimerkiksi edellämainittu jaksollinen kanttiaalto täytyy muodostaa sinien ja kosinien avulla. Tämä onnistuu käyttäen pientä osaa edellä lasketun Fourier-sarjan kertoimista. Seuraavassa vasemmalla ylhäällä oleva kuva esittää funktiota $x(t)$ ja sen Fourier-sarjaa, jossa on käytetty kertoimia $X(-5), X(-4), \ldots, X(5)$ (kaikkiaan 11 kpl). Näin saadaan approksimaatio kanttiaallolle:

$$\begin{aligned}\hat{x}(t) &= X(-5)e^{-5it} + X(-4)e^{-4it} + X(-3)e^{-3it} + \cdots + X(4)e^{4it} + X(5)e^{5it} \\ &= -0.0450e^{-5it} + 0.0750e^{-3it} + 0.1592e^{-2it} + 0.2251e^{-it} + 0.2500 \\ &+ 0.2251e^{it} + 0.1592e^{2it} + 0.0750e^{3it} - 0.0450e^{5it}.\end{aligned}$$

Valitsemalla kertoimet symmetrisesti nollan molemmin puolin, supistuvat kaikki lausekkeessa olevat imaginaariosat pois, ja lopputulos on reaalinen funktio.

Yhdentoista kertoimen avulla saatava approksimaatio ei ole vielä kovin tarkka, mutta kertoimia lisäämällä saadaan funktio lähemmäs todellista kanttiaaltoa. Yläoikealla on käytetty 21 kerrointa, alhaalla vasemmalla 41 kerrointa ja vihdoin alaoikealla 101 kerrointa. Kuvista havaitaan, että approksimaatio muuttuu tarkemmaksi kun kertoimien määrää lisätään. Pystyakselille jää kuitenkin aina noin kymmenen prosentin heitto reunan molemmille puolille. On havaittu, ettei tätä maksimipoikkeamaa voida pienentää kertoimien määrää kasvattamalla. Kertoimien lisääminen kyllä kaventaa korkeinta huippua, mutta sen korkeus pysyy suunnilleen vakiona. Tästä Fourier-sarjalle tyypillisestä käyttäytymisestä käytetään nimeä *Gibbs-ilmiö* (engl. *Gibbs phenomenon*).



3.3 Diskreettiaikainen Fourier-muunnos (diskreettiaikainen ei-jaksollinen signaali)

Kolmannessa muunnostyypissä ei-jaksollinen lukujono esitetään kompleksisten eksponentiaalien painotettuna integraalina. Toisin kuin ensimmäisessä muunnostyypissä, nyt riittää integroida välillä $[-\pi, \pi]$. Diskreettiaikaisen ei-jaksollisen signaalin $x(n)$ Fourier-muunnos on signaali

$$X(e^{i\omega}) = \sum_{n=-\infty}^{\infty} x(n)e^{-i\omega n}.$$

Tuloksena on siis reaalimuuttujan ω funktio X. Jos tämä funktio tiedetään, päästään takaisin alkuperäiseen signaaliin kaavalla

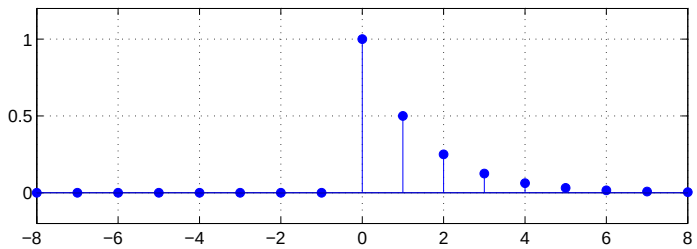
$$x(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{i\omega})e^{i\omega n}\,d\omega.$$

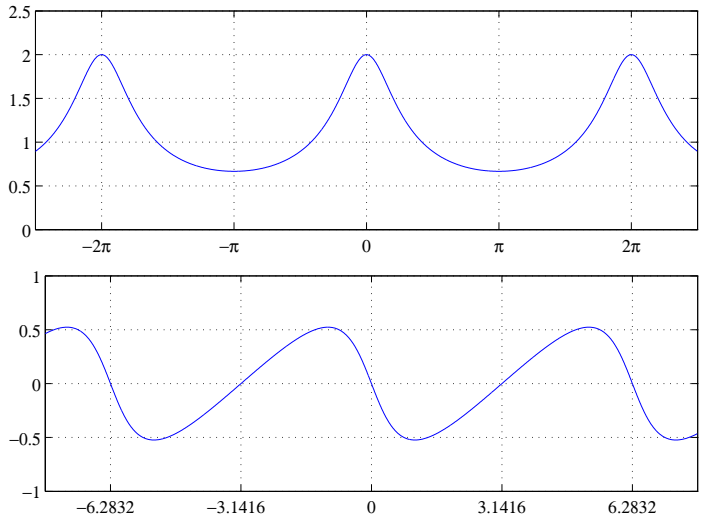
Kaavoista havaitaan, että itse asiassa tämä muunnos määritellään melkein samoilla kaavoilla kuin edellisen kappaleen Fourier-sarja.

Esimerkki: Määrätään signaalin $x(n) = 0.5^n u(n)$ Fourier-muunnos. Nyt

$$\begin{aligned}
X(e^{i\omega}) &= \sum_{n=-\infty}^{\infty} x(n)e^{-i\omega n} \\
&= \sum_{n=-\infty}^{\infty} 0.5^n e^{-i\omega n}u(n) \\
&= \sum_{n=0}^{\infty} 0.5^n e^{-i\omega n} \\
&= \sum_{n=0}^{\infty} (0.5e^{-i\omega})^n \qquad \text{(Geom. sarja, } |0.5e^{-i\omega}| = 0.5 < 1) \\
&= \frac{1}{1-0.5e^{-i\omega}}.
\end{aligned}$$

Alla ovat alkuperäisen signaalin ja sen Fourier-muunnoksen kuvaajat; ensimmäisenä on alkuperäisen signaalin $x(n)$ kuvaaja, toisena sen Fourier-muunnoksen itseisarvon $|X(e^{i\omega})|$ kuvaaja ja kolmantena Fourier-muunnoksen vaihekulman $\arg(X(e^{i\omega}))$ kuvaaja kulmataajuuden ω funktiona.





Diskreettiaikainen Fourier-muunnos $X(e^{i\omega})$ ilmaisee millä kertoimella eri kompleksiset eksponenttifunktiot tulee ottaa mukaan muodostettaessa tarkasteltavan signaalin esitystä taajuuksien avulla. Jos halutaan tietää kuinka suurella kertoimella taajuus f Hertseinä on mukana annetussa lukujonossa, jaetaan f ensin näytteenottotaajuudella F_s ja näin saatu normalisoitu taajuus $\frac{f}{F_s}$ muunnetaan kulmataajuudeksi ω kertomalla luvulla 2π. Oletetaan esimerkiksi, että edellisen esimerkin signaali $x(n) = 0.5^n u(n)$ on saatu aikaan järjestelmällä, jonka näytteenottotaajuus on 8000 Hz. Nyt 1000 Hertsin taajuus (eli signaali $e^{in\cdot(2\pi\cdot 1000/8000)}$) on signaalissa mukana kertoimella

$$X(e^{i\cdot 2\pi\frac{f}{Fs}}) = X(e^{i\cdot 2\pi\frac{1000}{8000}}) = X(e^{\frac{\pi i}{4}}) = \frac{1}{1-0.5e^{-\frac{\pi i}{4}}} = 1.19 - 0.65i.$$

Vastaavasti 2000 Hertsin taajuus (eli signaali $e^{in\cdot(2\pi\cdot 2000/8000)}$) on signaalissa $x(n)$ mukana kertoimella

$$X(e^{i\cdot 2\pi\frac{f}{Fs}}) = X(e^{i\cdot 2\pi\frac{2000}{8000}}) = X(e^{\frac{\pi i}{2}}) = \frac{1}{1-0.5e^{-\frac{\pi i}{2}}} = 0.8 - 0.4i.$$

Koska tuhannen Hertsin kerroin on itseisarvoltaan 1.36 ja kahdentuhannen Hertsin kerroin 0.89, on signaalissa enemmän tuhannen Hertsin taajuutta kuin kahdentuhannen Hertsin taajuutta. Kaikkien signaalin sisältämien taajuuksien jakauma voidaan lukea yllä olevasta funktion $|X(e^{i\omega})|$ kuvaajasta. Nollataajuus on luonnollisesti nollan kohdalla ja signaalin sisältämä suurin taajuus (eli Nyquistin rajataajuus) on kulmataajuuden $\omega = 2\pi \cdot (0.5F_s)/F_s = \pi$ kohdalla. Eniten signaalissa on nollataajuutta (kertoimen itseisarvo on 2) ja vähiten Nyquistin taajuutta (kertoimen itseisarvo on $\frac{2}{3}$).

Vaikka kertoimet ja eksponenttifunktiot ovatkin kompleksisia, integroinnin tuloksena on tässä tapauksessa reaalinen lukujono, koska negatiiviset taajuudet kumoavat kompleksitermit.

3.4 Diskreetti Fourier-muunnos (diskreettiaikainen jaksollinen signaali)

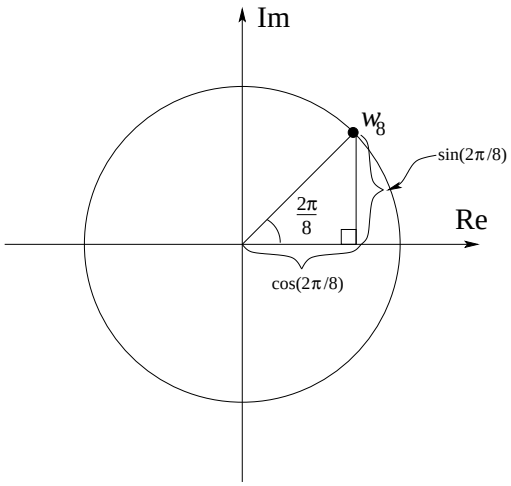
Sovellusten kannalta tärkein mainituista neljästä muunnostyypistä on diskreetti Fourier-muunnos joka muuntaa diskreettiaikaisen jaksollisen signaalin diskreettiaikaiseksi jaksol-

liseksi signaaliksi. Muunnoksen lopputulos sisältää tiedon alkuperäisen signaalin sisältämistä taajuuksista, joita on äärellinen määrä. Jakson pituutta kasvatettaessa kohti ääretöntä on lopputuloksena diskreettiaikainen Fourier-muunnos, eli taajuusakseli muuttuu tällöin jatkuvaksi. Koska jaksollisesta signaalista tarvitsee tietää vain yksi jakso, voidaan koko operaatio esittää vektoreiden avulla. Tällöin itse operaatio on matriisikertolasku. Tähän muotoon palataan määritelmien jälkeen.

Diskreetin Fourier-muunnoksen määritelmässä on kertoimena luku $w_N = e^{2\pi i/N}$, joka on nimeltään *ykkösen* N*:s juuri* (N'th root of unity). Nimensä mukaisesti tämän luvun N:s potenssi on yksi:

$$w_N^N = (e^{2\pi i/N})^N = e^{2\pi i} = \cos(2\pi) + i\sin(2\pi) = 1.$$

Luku w_N sijaitsee kompleksitason yksikköympyrällä kulmassa $2\pi/N$. Kulma on $\frac{1}{N}$ koko ympyrästä, ks. kuva.



Diskreettiaikaisen jaksollisen signaalin $x(n)$ (jakso N) Fourier-muunnos määritellään kaavalla

$$X(n) = \sum_{k=0}^{N-1} x(k) w_N^{-kn}.$$

Signaalin $X(n)$ käänteinen diskreetti Fourier-muunnos on

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) w_N^{kn}.$$

Diskreetti Fourier-muunnos (DFT) ja käänteinen diskreetti Fourier-muunnos (IDFT) on mahdollista esittää myös matriisimuodossa. Tämä onkin yleensä kaikkein kätevin esitysmuoto. Olkoon $x(n)$ muunnettava signaali, ja vektori

$$\mathbf{x} = \begin{pmatrix} x(0) \\ x(1) \\ x(2) \\ \vdots \\ x(N-1) \end{pmatrix}$$

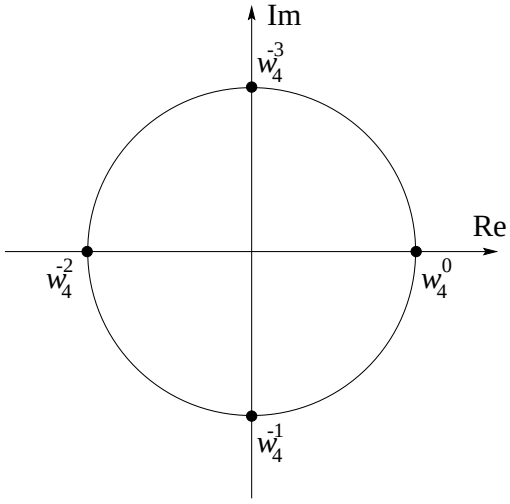
sen yksi jakso. Tällöin sen diskreetti Fourier-muunnos saadaan kertomalla vektori **x** matriisilla

$$\begin{pmatrix} w_N^0 & w_N^0 & w_N^0 & \dots & w_N^0 \\ w_N^0 & w_N^{-1} & w_N^{-2} & \dots & w_N^{-(N-1)} \\ w_N^0 & w_N^{-2} & w_N^{-4} & \dots & w_N^{-2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_N^0 & w_N^{-(N-1)} & w_N^{-2(N-1)} & \dots & w_N^{-(N-1)^2} \end{pmatrix}.$$

Esimerkiksi perusjaksolla $N = 4$ DFT saadaan laskettua kertomalla vektori matriisilla

$$\begin{pmatrix} w_4^0 & w_4^0 & w_4^0 & w_4^0 \\ w_4^0 & w_4^{-1} & w_4^{-2} & w_4^{-3} \\ w_4^0 & w_4^{-2} & w_4^{-4} & w_4^{-6} \\ w_4^0 & w_4^{-3} & w_4^{-6} & w_4^{-9} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix}.$$

Matriisi on generoitavissa kompleksitason yksikköympyrän avulla. Tapauksessa $N = 4$ kaikki termit ovat luvun w_4 ei-positiivisia potensseja. Luku w_4 sijaitsee yksikköympyrällä kulmassa $\frac{2\pi}{4}$, joka on neljäsosa koko ympyrästä. Negatiiviset potenssit saadaan kuten positiivisetkin, paitsi että kiertosuunta vaihtuu.



Muunnosmatriisin ylimmällä rivillä eksponentti on aina nolla, seuraavalla rivillä se pienenee aina yhdellä, sitä seuraavilla kahdella, kolmella, jne. Esimerkiksi tapauksessa $N = 6$ muunnosmatriisi on seuraava.

$$\begin{pmatrix} w_6^0 & w_6^0 & w_6^0 & w_6^0 & w_6^0 & w_6^0 \\ w_6^0 & w_6^{-1} & w_6^{-2} & w_6^{-3} & w_6^{-4} & w_6^{-5} \\ w_6^0 & w_6^{-2} & w_6^{-4} & w_6^{-6} & w_6^{-8} & w_6^{-10} \\ w_6^0 & w_6^{-3} & w_6^{-6} & w_6^{-9} & w_6^{-12} & w_6^{-15} \\ w_6^0 & w_6^{-4} & w_6^{-8} & w_6^{-12} & w_6^{-16} & w_6^{-20} \\ w_6^0 & w_6^{-5} & w_6^{-10} & w_6^{-15} & w_6^{-20} & w_6^{-25} \end{pmatrix} \begin{matrix} \leftarrow \text{eksponentin muutos on } 0 \\ \leftarrow \text{eksponentin muutos on } -1 \\ \leftarrow \text{eksponentin muutos on } -2 \\ \leftarrow \text{eksponentin muutos on } -3 \\ \leftarrow \text{eksponentin muutos on } -4 \\ \leftarrow \text{eksponentin muutos on } -5 \end{matrix}$$

Muunnosmatriisi on helppo tehdä piirtämällä yksikköympyrälle luvun w_N ei-positiiviset potenssit ja hyppimällä näin saatuja pisteitä myötäpäivään nollan, yhden, kahden, jne. askeleen välein.

Esimerkki: Lasketaan lukujonon $x(n)$: $(0,1,2,3)^T$ diskreetti Fourier-muunnos. Muunnos tapahtuu matriisikertolaskulla:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 6 \\ -2+2i \\ -2 \\ -2-2i \end{pmatrix}$$

Perusjakson $N = 4$ tapauksessa muunnosmatriisi on yksinkertainen. Tilanne saattaa olla toinen muilla perusjaksoilla.

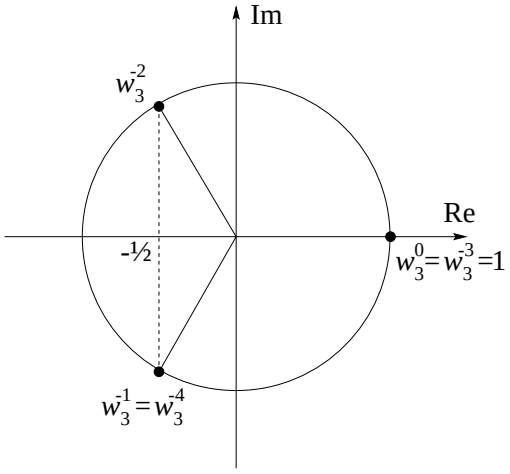
Esimerkki: Lasketaan lukujonon $x(n)$: $(0,1,2)^T$ diskreetti Fourier-muunnos. Nyt perusjakso on $N = 3$, jolloin tarvittava muunnosmatriisi on

$$\begin{pmatrix} w_3^0 & w_3^0 & w_3^0 \\ w_3^0 & w_3^{-1} & w_3^{-2} \\ w_3^0 & w_3^{-2} & w_3^{-4} \end{pmatrix}.$$

Nyt luvut $w_3^{-k}, k = 0,1,2,4$, saadaan laskettua Eulerin kaavan avulla, jolloin muunnosmatriisiksi tulee Nyt luvut $w_3^{-k}, k = 0,1,2,4$, saadaan laskettua Eulerin kaavan avulla, jolloin muunnosmatriisiksi tulee

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & -\frac{1}{2} - i\frac{\sqrt{3}}{2} & -\frac{1}{2} + i\frac{\sqrt{3}}{2} \\ 1 & -\frac{1}{2} + i\frac{\sqrt{3}}{2} & -\frac{1}{2} - i\frac{\sqrt{3}}{2} \end{pmatrix}$$

Muunnosmatriisin laskemista voidaan havainnollistaa yksikköympyrän avulla:



Haluttu diskreetti Fourier-muunnos on nyt siis

$$\mathbf{X} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -\frac{1}{2} - i\frac{\sqrt{3}}{2} & -\frac{1}{2} + i\frac{\sqrt{3}}{2} \\ 1 & -\frac{1}{2} + i\frac{\sqrt{3}}{2} & -\frac{1}{2} - i\frac{\sqrt{3}}{2} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 3 \\ -\frac{3}{2} + i\frac{\sqrt{3}}{2} \\ -\frac{3}{2} - i\frac{\sqrt{3}}{2} \end{pmatrix}.$$

Vektorin $\mathbf{X}$ käänteismuunnos saadaan kertomalla vektori matriisilla

$$\frac{1}{N}\begin{pmatrix} w_N^0 & w_N^0 & w_N^0 & \dots & w_N^0 \\ w_N^0 & w_N & w_N^2 & \dots & w_N^{N-1} \\ w_N^0 & w_N^2 & w_N^4 & \dots & w_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_N^0 & w_N^{N-1} & w_N^{2(N-1)} & \dots & w_N^{(N-1)^2} \end{pmatrix},$$

eli aiemmin olleen matriisin käänteismatriisilla. Matriisi eroaa aikaisemmasta siten, että eksponentit ovat vastakkaismerkkisiä ja matriisilla on kerroin $\frac{1}{N}$. Kertoimet nähdään tässäkin tapauksessa yksikköympyrältä, mutta nyt vastapäivään kiertämällä. Tapauksessa $N = 4$ matriisi saa muodon

$$\frac{1}{4}\begin{pmatrix} w_4^0 & w_4^0 & w_4^0 & w_4^0 \\ w_4^0 & w_4 & w_4^2 & w_4^3 \\ w_4^0 & w_4^2 & w_4^4 & w_4^6 \\ w_4^0 & w_4^3 & w_4^6 & w_4^9 \end{pmatrix} = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}.$$

Esimerkki: Olkoon erään signaalin diskreetti Fourier-muunnos $X(n)$: $(-3, \frac{3i+9\sqrt{3}}{3i+\sqrt{3}}, \frac{6i+3\sqrt{3}}{\sqrt{3}})^T$. Määrätään alkuperäinen signaali $x(n)$ eli signaalin $X(n)$ käänteinen Fourier-muunnos. Yllä on jo laskettu perusjaksoa $N = 3$ vastaava muunnosmatriisi. Vastaavasti saadaan käänteismuunnoksen matriisi:

$$\frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & -\frac{1}{2}+i\frac{\sqrt{3}}{2} & -\frac{1}{2}-i\frac{\sqrt{3}}{2} \\ 1 & -\frac{1}{2}-i\frac{\sqrt{3}}{2} & -\frac{1}{2}+i\frac{\sqrt{3}}{2} \end{pmatrix}.$$

Nyt

$$\frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & -\frac{1}{2}+i\frac{\sqrt{3}}{2} & -\frac{1}{2}-i\frac{\sqrt{3}}{2} \\ 1 & -\frac{1}{2}-i\frac{\sqrt{3}}{2} & -\frac{1}{2}+i\frac{\sqrt{3}}{2} \end{pmatrix}\begin{pmatrix} -3 \\ \frac{3i+9\sqrt{3}}{3i+\sqrt{3}} \\ \frac{6i+3\sqrt{3}}{\sqrt{3}} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -4 \end{pmatrix},$$

joten $x(n)$: $(1, 0, -4)^T$.

3.4.1 Diskreetin Fourier-muunnoksen ominaisuuksia

Seuraavassa tarkastellaan diskreetin Fourier-muunnoksen tärkeimpiä ominaisuuksia.

Lukujono	*Diskreetti Fourier-muunnos*
$x(n)$	$X(n)$
$y(n)$	$Y(n)$
$ax(n) + by(n)$	$aX(n) + bY(n)$
$x(n+m)$	$w_N^{nm}X(n)$
$w_N^{-nm}x(n)$	$X(n+m)$
$x(n) * y(n)$ (konvoluutio)	$X(n)Y(n)$
$x(n)y(n)$	$\frac{1}{N}X(n) * Y(n)$
$\overline{x(n)}$ (kompleksikonjugaatti)	$\overline{X(-n)}$

Lisäksi seuraavat ominaisuudet pitävät paikkansa jos lukujono $x(n)$ on reaalinen.

- $X(n) = \overline{X(-n)}$
- Re $[X(n)]$ =Re $[X(-n)]$
- Im $[X(n)] = -$Im $[X(-n)]$
- $|X(n)| = |X(-n)|$ (DFT on siis symmetrinen origon suhteen.)
- $\arg[X(n)] = -\arg[X(-n)]$.

Osoitetaan seuraavaksi, että jonon $y(n) = x(n+m)$ diskreetti Fourier-muunnos on $w_N^{nm}X(n)$, kun jonon $x(n)$ diskreetti Fourier-muunnos on $X(n)$. Tarkastellaan jonon $y(n)$ diskreettiä Fourier-muunnosta:

$$\begin{aligned}
& \sum_{k=0}^{N-1} y(k) w_N^{-nk} \\
= \; & w_N^{nm} \sum_{k=0}^{N-1} y(k) w_N^{-nk} w_N^{-nm} \\
= \; & w_N^{nm} \sum_{k=0}^{N-1} y(k) w_N^{-n(k+m)} \\
= \; & w_N^{nm} \sum_{k=0}^{N-1} x(k+m) w_N^{-n(k+m)}.
\end{aligned}$$

Koska jono $x(n)$ on periodinen jaksolla N, niin voidaan viimeisessä summassa vaihtaa muuttujaa $r = k + m$. Silloin saadaan lauseke

$$w_N^{nm} \sum_{r=0}^{N-1} x(r) w_N^{-nr}.$$

Tutkitaan toisena esimerkkinä vielä erittäin tärkeää konvoluution diskreettiä Fourier-muunnosta[4]. Olkoot $x(n)$ ja $y(n)$ jaksollisia lukujonoja (jakso $= N$) ja $X(n)$ ja $Y(n)$ niiden

[4]Tässä yhteydessä kyseessä on ns. jaksollinen konvoluutio (engl. circular convolution), joka saadaan lausekkeesta

$$x(n) * y(n) = \sum_{k=0}^{N-1} x(k) y(n-k)$$

eikä tavallinen konvoluutio, joka saadaan lausekkeesta

$$x(n) * y(n) = \sum_{k=-\infty}^{\infty} x(k) y(n-k).$$

Tavallinen konvoluutiohan ei jaksollisten jonojen tapauksessa edes suppene, ja diskreetin Fourier-muunnoksen yhteydessä jonot oletetaan jaksollisiksi. Jaksollisesta konvoluutiosta käytetään joissain yhteyksissä merkintää

$$x(n) \circledast y(n).$$

Fourier-muunnokset. Tarkastellaan jonon $X(n)Y(n)$ käänteistä Fourier-muunnosta:

$$\begin{aligned}
& \frac{1}{N}\sum_{k=0}^{N-1}[X(k)Y(k)]w_N^{nk} \\
= \; & \frac{1}{N}\sum_{k=0}^{N-1}\left[\sum_{r=0}^{N-1}x(r)w_N^{-kr}\right]\left[\sum_{m=0}^{N-1}y(m)w_N^{-km}\right]w_N^{nk} \\
= \; & \frac{1}{N}\sum_{r=0}^{N-1}\sum_{m=0}^{N-1}x(r)y(m)\sum_{k=0}^{N-1}w_N^{-kr}w_N^{-km}w_N^{nk} \\
= \; & \frac{1}{N}\sum_{r=0}^{N-1}\sum_{m=0}^{N-1}x(r)y(m)\underbrace{\sum_{k=0}^{N-1}w_N^{k(n-r-m)}}_{(*)}.
\end{aligned}$$

Nyt lauseke $(*)$ saa arvon N silloin kun $n-r-m=0$, eli $m=n-r$ ja arvon 0 muulloin. Näin ollen koko summalauseke saa muodon

$$\frac{1}{N}\sum_{r=0}^{N-1}x(r)y(n-r)\cdot N = x(n)*y(n).$$

Fourier-muunnos muuntaa siis konvoluution kertolaskuksi. Ominaisuus on voimassa muillakin muunnostyypeillä sekä seuraavan kappaleen z-muunnoksella. Kertolaskun yksinkertaisuus ja havainnollisuus konvoluutioon verrattuna tekee tästä ominaisuudesta tärkeän työkalun jatkossa. Tämä myös antaa konvoluutiolle taajuustulkinnan: konvoluutiossa taajuusjakaumat kerrotaan keskenään. Tämä mahdollistaa sellaisten jonojen suunnittelun, jotka konvoluutiossa poistavat tietyt taajuudet kokonaan—riittää, että taajuusvaste on poistettavalla alueella nolla.

3.4.2 Nopea Fourier-muunnos (FFT)

Vektoreiden Fourier-muunnosten laskeminen suoraan määritelmän perusteella on käytännössä melko vaivalloista. N-ulotteisen vektorin kertominen $N \times N$-matriisilla vaatii N^2 kertolaskua ja $N(N-1)$ yhteenlaskua. N-ulotteisen Fourier-muunnoksen ajantarve on siis suoraan verrannollinen dimension neliöön. Kuitenkin niin sanotulla *nopealla Fourier-muunnoksella* (engl. Fast Fourier Transform; FFT) voidaan Fourier-muunnos toteuttaa kompleksisuudella $O(N \log N)$, eli huomattavasti nopeammin etenkin kun N on suuri. Seuraavassa tarkastellaan nopeaa Fourier-muunnosta.

Vuonna 1965 julkaistussa artikkelissaan Cooley ja Tukey esittelivät nopeamman tavan suorittaa diskreetti Fourier-muunnos, ja tämä tulos antoikin Fourier-analyysille aivan uusia ulottuvuuksia ja käytännön sovelluksia. Vaikka nopean Fourier-muunnoksen periaate olikin jo Gaussin keksimä, vasta Cooley ja Tukey havaitsivat sen käyttökelpoisuuden tietokoneilla suoritettavassa laskennassa.

Tämä Cooleyn ja Tukeyn nopea Fourier-muunnos perustuu niin sanottuun *hajoita ja hallitse* -algoritmien suunnittelumenetelmään, jossa annettu tehtävä muunnetaan ensin pienemmiksi osaongelmiksi, jotka sitten voidaan ratkaista huomattavasti helpommin kuin

alkuperäinen ongelma. FFT-algoritmin tapauksessa muunnettava vektori jaetaan kahteen osaan ja näin saadut vektorit muunnetaan rekursiivisesti samalla algoritmilla. Rekursio loppuu kun muunnettavan vektorin dimensio on yksi, koska yksiulotteisen vektorin Fourier-muunnos on määritelmän perusteella se itse. Koska joka vaiheessa vektori jaetaan kahtia, on sen alkuperäinen dimensio oltava muotoa 2^k, $k \in \mathbf{N}$.

Itse muunnoksen johtamisessa tarvitaan seuraavaa huomiota luvuista w_k ja w_{2k}. Näillä luvuilla on nimittäin voimassa sääntö

$$w_{2k}^2 = e^{\frac{2\cdot 2\pi i}{2k}} = e^{\frac{2\pi i}{k}} = w_k. \tag{3.1}$$

Säännön mukaan ylä- ja alaindeksi voidaan "supistaa" esimerkiksi näin: $w_4^2 = w_2$ ja $w_{10}^2 = w_5$. Yhtälöstä (3.1) nähdään helposti, että sääntö on voimassa myös muille yhteisille tekijöille kuin kakkoselle. Seuraavassa Fourier-muunnoksen algoritmin johtamisessa tarvitaan kuitenkin vain kaavan (3.1) tapausta.

Olkoon $x(n)$ jaksollinen lukujono, jonka jakso on $N = 2^k$ ($k \in \mathbf{N}$). Sen diskreetti Fourier-muunnos on määritelmän mukaan

$$X(n) = \sum_{k=0}^{N-1} x(k) w_N^{-nk}.$$

Tarkastellaan erikseen parillis- ja paritonindeksisiä komponentteja:

$$X(n) = \sum_{k=0}^{\frac{N}{2}-1} x(2k) w_N^{-n(2k)} + \sum_{k=0}^{\frac{N}{2}-1} x(2k+1) w_N^{-n(2k+1)}.$$

Koska kaavan (3.1) mukaan $w_N^2 = w_{N/2}$, niin

$$X(n) = \sum_{k=0}^{\frac{N}{2}-1} x(2k) w_{N/2}^{-nk} + w_N^{-n} \sum_{k=0}^{\frac{N}{2}-1} x(2k+1) w_{N/2}^{-nk}.$$

Jos merkitään $x_0(k) = x(2k)$ ja $x_1(k) = x(2k+1)$ (ja Fourier-muunnoksia vastaavasti $X_0(n)$ ja $X_1(n)$), niin saadaan yhtälö

$$X(n) = X_0(n) + w_N^{-n} X_1(n), \tag{3.2}$$

kun $n = 0, 1, 2, \ldots, N/2 - 1$ ja

$$X(n) = X_0(n - N/2) + w_N^{-n} X_1(n - N/2), \tag{3.3}$$

kun $n = N/2, N/2 + 1, \ldots, N - 1$.

Jakojäännöksen avulla nämä voidaan esittää myös yhtenä kaavana:

$$X(n) = X_0\left(\operatorname{mod}(n, N/2)\right) + w_N^{-n} X_1\left(\operatorname{mod}(n, N/2)\right), \tag{3.4}$$

missä $n = 0, 1, \ldots, N - 1$ ja $\operatorname{mod}(a, b)$ on jakojäännös jaettaessa luku a luvulla b.

Näin ollen Fourier-muunnos periodiselle jonolle, jonka jakso on N saadaan laskemalla kaksi Fourier-muunnosta, joiden pituudet ovat $N/2$. Nämä kaksi edelleen saadaan rekursiivisesti laskemalla yhteensä neljä Fourier-muunnosta, joiden pituudet ovat $N/4$. Näin saatiin nopean Fourier-muunnoksen algoritmi.

Nopean Fourier-muunnoksen algoritmi

1. Jos muunnettavan vektorin dimensio on $N = 1$, palauta se sellaisenaan.
2. Jaetaan muunnettava lukujono $x(n)$ kahdeksi lukujonoksi $x_0(n)$ ja $x_1(n)$, joista $x_0(n)$ koostuu parillisindeksisistä ja $x_1(n)$ paritonindeksisistä komponenteista.
3. Lasketaan lukujonojen $x_0(n)$ ja $x_1(n)$ Fourier-muunnokset rekursiivisesti tällä algoritmilla.
4. Lopputulos saadaan yhdistämällä jonot $X_0(n)$ ja $X_1(n)$ kaavojen (3.2) ja (3.3) tai (3.4) mukaisesti.

Esimerkki: Tarkastellaan jaksollista lukujonoa $(0, 1, 2, 3)^T$, jonka jakso on 4 ja suoritetaan sille nopea Fourier-muunnos. Ensin muodostetaan lukujonot $x_0(n)$: $(0, 2)^T$ ja $x_1(n)$: $(1, 3)^T$, joille suoritetaan Fourier-muunnokset. Jonojen Fourier-muunnokset ovat $X_0(n)$: $(2, -2)^T$ ja $X_1(n)$: $(4, -2)^T$. Kaavoista (3.2) ja (3.3) saadaan

$$\begin{aligned}
X(0) &= \underbrace{X_0(0)}_{=2} + \underbrace{w_4^0}_{=1}\underbrace{X_1(0)}_{=4} = 6 \\
X(1) &= \underbrace{X_0(1)}_{=-2} + \underbrace{w_4^{-1}}_{=-i}\underbrace{X_1(1)}_{=-2} = -2 + 2i \\
X(2) &= \underbrace{X_0(2-2)}_{=2} + \underbrace{w_4^{-2}}_{=-1}\underbrace{X_1(2-2)}_{=4} = -2 \\
X(3) &= \underbrace{X_0(3-2)}_{=-2} + \underbrace{w_4^{-3}}_{=i}\underbrace{X_1(3-2)}_{=-2} = -2 - 2i
\end{aligned}$$

Vastaavasti voidaan alkuperäinen Fourier-muunnoksen määräävä summa jakaa neljään eri osaan, jolloin saadaan niin sanottu *radix-4* FFT-algoritmi. Kahdeksaan osaan jakaminen tuottaa vastaavasti *radix-8*-algoritmin, jne. Tällöin lukujonojen pituuden pitää olla joku neljän tai kahdeksan potenssi.

Seuraavassa on Fourier-muunnoksen toteuttava C++ -kielinen ohjelmakoodi. Ohjelmasta puuttuu pääohjelma, joka kutsuu funktiota FFT. Kääntyvä ohjelma, jossa myös pääohjelma on mukana, löytyy osoitteesta

`http://www.cs.tut.fi/kurssit/SGN-11000/FFT.cpp`

```
// Määritellään oma kompleksivektorityyppi luettavuuden
// parantamiseksi.

typedef vector < complex <double> > ComplexVector;

ComplexVector FFT (ComplexVector data)
{
   int N = data.size();

   ComplexVector even;
   ComplexVector odd;
   ComplexVector evenResult;
   ComplexVector oddResult;
   ComplexVector finalResult (N);
   int k;

   // Määritellään pii, imaginääriyksikkö ja ykkösen N:s juuri.

   double pi = 3.14159265358979;
   complex<double> I (0, 1);
   complex<double> w_N = exp(2 * pi * I / (double) N);

   // Jos muunnettavan vektorin dimensio on 1, rekursio päättyy.

   if (N == 1)
   {
      return data;
   }

   // Jaetaan syöte paritonindeksisiin (odd) ja
   // parillisindeksisiin (even) alkioihin.

   for (k = 0; k < N; k += 2)
   {
      even.push_back (data [k]);
      odd.push_back  (data [k+1]);
   }

   // Lasketaan näin saatujen N/2-ulotteisten
   // vektoreiden (even ja odd) Fourier-muunnokset.

   evenResult = FFT (even);
   oddResult  = FFT (odd);

   // Lasketaan lopullinen tulos yhdistämällä jonot
   // kaavalla (3.4).

   for (k = 0; k < N; k++)
   {
      finalResult [k] = evenResult [k % (N/2)]
         + pow (w_N, -k) * oddResult [k % (N/2)];
   }

   return finalResult;
}
```

Harjoitustehtäviä

3.1. Laske käsin lukujonon $x(n) = (-1, 2, 3, 1)$ diskreetti Fourier-muunnos ja piirrä sen itseisarvojen kuvaaja. (Tässä $x(n)$ tarkoittaa lukujonoa, jonka jakso $N = 4$ ja termit $\ldots, -1, 2, 3, 1, -1, 2, 3, 1, -1, 2, 3, 1 \ldots$.)

3.2. Laske lukujonon $x(n) = (1, 1, 1, 1, 0, 0, 0, 0)$ diskreetti Fourier-muunnos.

3.3. Laske nopean Fourier-muunnoksen algoritmia jäljitellen jonon $x(n) = (4, -1, 2, 0)$ diskreetti Fourier-muunnos. Voit käyttää hyväksi tietoa, että lukujonon $(4, 2)$ DFT on $(6, 2)$ ja lukujonon $(-1, 0)$ DFT on $(-1, -1)$.

3.4. Laske nopean Fourier-muunnoksen algoritmia jäljitellen jonon $x(n) = (-7, 0, -4, -9, 5, 9, 8, 8)$ diskreetti Fourier-muunnos. Voit käyttää hyväksi tietoa, että lukujonon $(-7, -4, 5, 8)$ DFT on $(2, -12+12i, -6, -12-12i)$ ja lukujonon $(0, -9, 9, 8)$ DFT on $(8, -9+17i, 10, -9-17i)$.

3.5. *(Matlab)* Tutkitaan kolmea jaksollista signaalia, joiden yksi jakso on

$$x_1(n) = \begin{cases} 1, & \text{kun } 0 \le n \le 7 \\ 0, & \text{kun } 8 \le n \le 30 \end{cases}$$

$$x_2(n) = \begin{cases} 1, & \text{kun } 0 \le n \le 7 \\ 0, & \text{kun } 8 \le n \le 60 \end{cases}$$

$$x_3(n) = \begin{cases} 1, & \text{kun } 0 \le n \le 7 \\ 0, & \text{kun } 8 \le n \le 120 \end{cases}$$

Laske näiden diskreetit Fourier-muunnokset Matlabilla ja tulosta niiden itseisarvojen kuvaajat samaan ikkunaan (*komentoja:* `fft, abs, plot, stem, subplot`).

3.6. Osoita konvoluution Fourier-muunnoksen johtamisessa tarvittu tulos

$$\sum_{k=0}^{N-1} w_N^{kn} = \begin{cases} N, & \text{kun } n = 0, \\ 0, & \text{kun } n = 1, 2, \ldots, N-1. \end{cases}$$

3.7. Osoita, että yksikköimpulssissa $\delta(n)$ on kaikkia taajuuksia yhtä paljon. (*Vihje: Laske sopiva Fourier-muunnos.*)

3.8. *(Matlab)* Generoi yhden sekunnin mittainen signaali, jonka taajuus on 1000 Hz, kun näytteenottotaajuus on 8192 Hz. Laske signaalin DFT Matlabin komennolla `fft` ja tulosta ruudulle sen itseisarvojen kuvaaja. (`help fft, help plot, help abs`). Kuviossa pitäisi erottua selvä piikki vaaka-akselin kahdessa kohdassa. Piikki vastaa taajuutta 1000 Hz.

3.9. *(Matlab)* Luentomonisteessa laskettiin lukujonon $x(n) = 0.5^n u(n)$ diskreettiaikaiseksi Fourier-muunnokseksi $X(e^{i\omega}) = 1/(1 - 0.5e^{-i\omega})$. Tulosta tämän kuvaaja luomalla ensin taajuusvektori `omega=0:0.1:2*pi;` ja evaluoimalla funktio $|X(e^{i\omega})|$ näissä pisteissä: `X=abs(1./(1-0.5*exp(-i*omega)));`. Tulosta kuvaaja ruudulle:

`plot(omega,X);` Tuloskuvaajaa voidaan approksimoida FFT:n avulla. Ota lukujonon 64 ensimmäistä termiä (`x=0.5.^(0:63);`), laske sen FFT ja tulosta itseisarvot komennolla `stem` ruudulle. Vertaa tuloksia.

3.10. *(Matlab)* Fourier-muunnoksen matriisin (jakso N = 8) voi luoda komennolla

```
N=8;
F=exp(-2*pi*i*(0:N-1)'*(0:N-1)/N);
```

Tällöinhän satunnaisvektorin `rand(N,1)` Fourier-muunnos saadaan komennolla `X=F*rand(N,1)`. Vertaillaan laskennan tehokkuutta suhteessa FFT-algoritmiin. Matlab antaa käytetyn CPU-ajan komennoilla

```
tic; X=F*rand(N,1); toc
```

Vertaa tätä FFT:n nopeuteen, jonka saat komennoilla

```
tic; X=fft(rand(N,1)); toc
```

Tee kokeet ensin tapauksessa N = 8. Tee sitten samat kokeet kun N = 16, 32, 64, 128, 256, 512 ja 1024. Tee kokeet muutamaan kertaan, koska ajat vaihtelevat jonkin verran. Tulosta ruudulle laskenta-aikoja kuvaavat käyrät. Jos käyttämäsi kone on kovin nopea, tulokset voivat pyöristyä nollaan. Selvitä silloin esim. sadan DFT:n ja FFT:n vaatimat laskenta-ajat sijoittamalla laskenta for-silmukan sisään; esim.

```
tic; for k=1:100, X=fft(rand(N,1)); end; toc
```

3.11. *(Matlab)*

(a) Luo satunnaissignaali (=vektori), jonka pituus on 8192 näytettä (`help rand`). Laske sen (jaksollinen) konvoluutio signaalin `h=0.05*[ones(1,20),zeros(1,8192-20)];` kanssa käyttäen fft:tä ja ifft:tä.[5] Sijoita tulos vektoriin `y1`. Laske suoralla konvoluutiolla (Matlabissa komento `conv`) vastaava tulos ja sijoita tulos vektoriin `y2`. Tulosta ruudulle molempien tulossignaalien 200 ensimmäistä näytettä:

```
plot((1:200),real(y1(1:200)),(1:200),y2(1:200))
```

Muuttujasta `y1` täytyy ottaa reaaliosa, koska imaginääriosa ei ole laskentatarkkuudesta johtuen tarkalleen nolla (tyypillisesti luokkaa 10^{-15}). Kuviossa eroa pitäisi olla ainoastaan kahdessakymmenessä ensimmäisessä näytteessä, jotka FFT:llä saatava jaksollinen konvoluutio laskee eri tavalla.

(b) Vertaile menetelmien laskenta-aikoja samalla tavalla kuin tehtävässä 3.10.

3.12. *(Matlab)* Osoitteessa

[5] Tarvitset kahden vektorin pisteittäistä kertolaskua; Matlab kertoo vektorit `a` ja `b` keskenään pisteittäin komennolla `c=a.*b`, jolloin tulos menee vektoriin `c`.

`http://www.cs.tut.fi/kurssit/SGN-1200/corrupt.mat`

on vääristynyt versio Matlabin `handel`-testisignaalista. Tehtävänäsi on selvittää vääristymisprosessin impulssivaste. Lataa tätä varten molemmat edellämainitut tiedostot, jolloin alkuperäinen `handel` on muuttujassa `y` ja vääristynyt muuttujassa `z`. Katkaise vielä molemmat lyhyemmiksi laskennan nopeuttamiseksi: `y=y(1:2^16);` ja `z=z(1: 2^16);` Jos oletetaan, että vääristyminen on lineaarinen prosessi (tässä tapauksessa tämä pitää paikkansa), on voimassa yhtälö

$$z(n) = h(n) * y(n),$$

missä $h(n)$ on vääristymisen aiheuttaneen kanavan impulssivaste. Ottamalla Fourier-muunnokset yhtälön molemmista puolista, saadaan yhtälö

$$Z(n) = H(n)Y(n).$$

Koska DFT muunsi konvoluution kertolaskuksi, voidaan impulssivasteen DFT ratkaista jakolaskulla,

$$H(n) = Z(n)/Y(n),$$

josta saadaan ratkaistua impulssivaste $h(n)$ käänteisellä DFT:llä, eli komennolla `ifft`. Tulosta $h(n)$:n impulssivasteen 10 ensimmäistä termiä ruudulle.

3.13. Olkoon $x(t) = x(-t)$ ja $x(t) \in \mathbf{R}$ kaikilla $t \in \mathbf{R}$. Osoita, että signaalin $x(t)$ Fourier-muunnos

$$X(e^{i\omega}) = \int_{-\infty}^{\infty} x(t)e^{-i\omega t}\,dt.$$

on reaalinen.

3.14. Laske seuraavan jatkuvan funktion Fourier-muunnos

$$x(t) = \begin{cases} \cos(2\pi \cdot 10 \cdot t), & \text{kun } -\pi/2 < t < \pi/2, \\ 0, & \text{muulloin.} \end{cases}$$

Kyseinen äärellisen mittainen sinipulssi on käytössä tiedonsiirrossa ja tunnetaan nimellä *RF-pulssi*, joten on kiinnostavaa tietää kuinka laajan alueen se varaa taajuusspektristä. Vinkki: integraali ratkeaa käyttämällä Eulerin kaavasta saatavia kosinin ja sinin lausekkeita:

$$\cos x = \frac{e^{ix} + e^{-ix}}{2} \qquad \sin x = \frac{e^{ix} - e^{-ix}}{2i}$$

3.15. Laske jompi kumpi seuraavien jatkuvien funktioiden Fourier-muunnoksista:

(a) $x(t) = \begin{cases} 1, & \text{kun } t \geq 0 \\ 0, & \text{kun } t < 0 \end{cases}$

(b) $x(t) = \begin{cases} 1 - 2|t|, & \text{kun } |t| < \frac{1}{2} \\ 0, & \text{kun } |t| \geq \frac{1}{2} \end{cases}$

3.16. Mikä on Fourier-muunnoksen matriisi tapauksessa $N = 2$?

3.17. Mikä on Fourier-muunnoksen matriisi tapauksessa $N = 6$? Harjoituksissa voit 36:n kompleksiluvun taululle kirjaamisen sijaan kuvata kuinka ne saadaan.

Luku 4

Z-muunnos

Z-muunnosta voidaan pitää Fourier-muunnoksen yleistyksenä, ja se on tärkeä työkalu diskreettiaikaisten järjestelmien analyysissä ja suunnittelussa. Z-muunnos on myös läheistä sukua jatkuva-aikaisten järjestelmien tarkastelussa käytetylle *Laplace-muunnokselle.*

Z-muunnos muuntaa lukujonon rationaalifunktioksi, jonka ominaisuuksia tarkastelemalla saadaan selville tiettyjä lukujonon ominaisuuksia. Esimerkiksi impulssivasteen z-muunnoksesta nähdään helposti onko lukujono kausaalinen tai stabiili.

4.1 Z-muunnoksen määritelmä

Olkoon $x(n)$ lukujono. Sen z-muunnos on sarja

$$X(z) = \ldots + x(-3)z^3 + x(-2)z^2 + x(-1)z + x(0) + x(1)z^{-1} + x(2)z^{-2} + x(3)z^{-3} + \ldots$$

Jatkossa tästä käytetään lyhyempää merkintää

$$X(z) = \sum_{n=-\infty}^{\infty} x(n)z^{-n}. \tag{4.1}$$

Äärettömänä sarjana z-muunnoksen suppeneminen riippuu sekä lukujonon $x(n)$ arvoista, että pisteestä $z \in \mathbf{C}$, jossa $X(z)$ lasketaan. Siksi z-muunnoksen käsitteeseen liittyy olennaisesti se kompleksitason osa, jossa sarja suppenee. Sarjan *suppenemisalue* (engl. region of cenvergence, ROC) on se kompleksitason alue, jossa sarja (4.1) suppenee itseisesti, eli

$$\sum_{n=-\infty}^{\infty} |x(n)z^{-n}| < \infty.$$

Voidaan osoittaa, että sarjan $X(z)$ suppenemisalue on aina renkaan muotoinen alue

$$r_- < |z| < r_+.$$

Mahdollisesti suppenemisalueeseen kuuluu myös renkaan reunapisteitä eli pisteitä, jotka toteuttavat ehdon $|z| = r_-$ tai $|z| = r_+$. Lisäksi r_+ saattaa olla myös ääretön ja r_- voi olla nolla.

Koska z-muunnos on Fourier-muunnoksen yleistys, käytetään siitä samaa merkintätapaa kuin Fourier-muunnoksestakin: lukujonoa merkitään pienellä kirjaimella ja muunnosta vastaavalla isolla kirjaimella, esimerkiksi

$$\begin{aligned} x(n) &\leftrightarrow X(z), \\ y(n) &\leftrightarrow Y(z), \\ w(n) &\leftrightarrow W(z). \end{aligned}$$

Signaalinkäsittelyn ongelmissa tullaan usein käsittelemään lukujonoja, joiden z-muunnos on rationaalifunktio, eli muotoa

$$X(z) = \frac{N(z)}{D(z)} = \frac{\sum_{m=0}^{M} a_m z^{-m}}{\sum_{k=0}^{K} b_k z^{-k}}$$

Tällaisten funktioiden nimittäjän $D(z)$ nollakohtia kutsutaan *navoiksi* ja osoittajan $N(z)$ nollakohtia *nolliksi*. Huomaa, että kaikki eksponentit ovat tavallisesti negatiivisia, joten nollakohtien laskentaa varten ne täytyy laventaa positiivisiksi.

Esimerkki: Lasketaan navat ja nollat, kun

$$X(z) = \frac{1 - 4z^{-1} + z^{-2}}{1 - z^{-1} + \frac{1}{2}z^{-2}}.$$

Lavennetaan negatiiviset potenssit pois kertomalla osoittaja ja nimittäjä termillä z^2:

$$X(z) = \frac{z^2 - 4z + 1}{z^2 - z + \frac{1}{2}}.$$

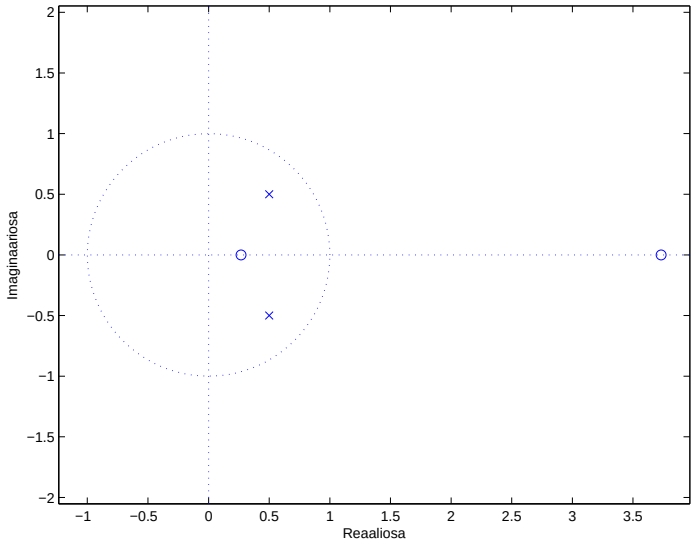
Tämän jälkeen ratkaistaan osoittajan nollakohdat (eli *nollat*) toisen asteen yhtälön ratkaisukaavalla:

$$z_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} = \frac{4 \pm \sqrt{16 - 4}}{2} = 2 \pm \sqrt{3} = \begin{cases} 3.73, \\ -0.27. \end{cases}$$

Nimittäjän nollakohdat (eli *navat*) saadaan vastaavasti:

$$p_{1,2} = \frac{1 \pm \sqrt{1 - 2}}{2} = \frac{1}{2} \pm \frac{i}{2}.$$

Navat ja nollat kuvataan usein kompleksitason koordinaatistossa. Edellisen esimerkin *napa-nollakuvio* on alla. Kuviossa nollista käytetään merkintää '$\circ$' ja navoista '$\times$'.



Funktio $X(z)$ määrää yhdessä suppenemisalueen kanssa signaalin $x(n)$ yksikäsitteisesti. Jos suppenemisalue ei ole tiedossa, saattaa $X(z)$ esittää useiden funktioiden z-muunnosta. Jatkossa tarkastellaan enimmäkseen kausaalisia signaaleja $x(n)$, joille siis $x(n) = 0$, kun $n < 0$. Tällöin z-muunnos saa muodon

$$X(z) = \sum_{n=0}^{\infty} x(n) z^{-n}.$$

4.2 Tavallisimpien jonojen z-muunnokset

Seuraavassa esitetään muutamia esimerkkejä joidenkin signaalien z-muunnoksista. Koska z-muunnos on lineaarinen, voidaan monimutkaisemman z-muunnoksen määrittäminen usein palauttaa näiden yksinkertaisten tapausten muunnoksiin.

Impulssi: Signaalin $x(n) = \delta(n)$ z-muunnos on

$$X(z) = 1$$

ja suppenemisalue on koko kompleksitaso $\mathbf{C}$.

Viivästetty impulssi: Olkoon jono $x(n) = \delta(n - d)$ ($d \in \{1, 2, 3, \ldots\}$). Silloin

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n} = \sum_{n=-\infty}^{\infty} \underbrace{\delta(n-d)}_{=0, \text{kun } n \neq d} z^{-n} = z^{-d},$$

ja suppenemisalue on $\mathbf{C} \setminus \{0\}$.

Yksikköaskel: Signaalin $x(n) = u(n)$ z-muunnos on

$$\begin{aligned} X(z) &= \sum_{n=-\infty}^{\infty} x(n)z^{-n} \\ &= \sum_{n=-\infty}^{\infty} u(n)z^{-n} \\ &= \sum_{n=0}^{\infty} z^{-n} \\ &= \sum_{n=0}^{\infty} \left(z^{-1}\right)^n \\ &= \frac{1}{1-z^{-1}} = \frac{z}{z-1}, \end{aligned}$$

ja suppenemisalue $|z| > 1$ (vrt. geometrisen sarjan suppeneminen). Lisäksi lausekkeesta havaitaan, että funktiolla $X(z)$ on nolla pisteessä $z = 0$ ja napa pisteessä $z = 1$.

Eksponenttijono: Signaalin $x(n) = a^n u(n)$ z-muunnos on

$$\begin{aligned} X(z) &= \sum_{n=-\infty}^{\infty} x(n)z^{-n} \\ &= \sum_{n=-\infty}^{\infty} a^n u(n)z^{-n} \\ &= \sum_{n=0}^{\infty} \left(az^{-1}\right)^n \\ &= \frac{1}{1-az^{-1}} = \frac{z}{z-a}, \end{aligned}$$

ja suppenemisalue $|z| > |a|$ (kts. geometrisen sarjan suppeneminen). Lisäksi z-muunnoksella on napa pisteessä $z = a$ ja nolla pisteessä $z = 0$.

Edellisessä esimerkissä tuli vastaan tilanne, jossa funktion ensimmäisessä muodossa on muuttujan z negatiivisia potensseja. Napojen ja nollien laskemista varten lauseke täytyy laventaa sellaiseksi, että kaikki eksponentit ovat positiivisia. Esimerkiksi funktion

$$X(z) = \frac{1 - 0.9z^{-1} + z^{-2} - 0.9z^{-3}}{1 + 0.5z^{-1} - 0.25z^{-2}}$$

pienin eksponentti on -3, joten se lavennetaan termillä z^3. Tällöin saadaan tulokseksi

$$X(z) = \frac{z^3 - 0.9z^2 + z - 0.9}{z^3 + 0.5z^2 - 0.25z}.$$

Nyt navat ovat $p_1 = -0.809, p_2 = 0$ ja $p_3 = 0.309$. Nollat puolestaan ovat pisteissä $z_1 = 0.9, z_2 = i$ ja $z_3 = -i$.

Kahdella eri signaalilla voi olla sama z-muunnoksen lauseke, mutta eri suppenemisalue. Nimittäin, signaalin

$$x(n) = \begin{cases} -a^n, & \text{kun } n < 0 \\ 0, & \text{kun } n \geq 0 \end{cases}$$

z-muunnos on

$$\begin{aligned} X(n) &= -\sum_{n=-\infty}^{-1} a^n z^{-n} \\ &= -\sum_{n=1}^{\infty} \left(a^{-1}z\right)^n \\ &= 1 - \sum_{n=0}^{\infty} \left(a^{-1}z\right)^n \\ &= 1 - \frac{1}{1-a^{-1}z} = 1 + \frac{a}{z-a} = \frac{z}{z-a}, \end{aligned}$$

ja sarjan suppenemisalue on $\left|\frac{z}{a}\right| < 1$, eli $|z| < |a|$. Tällä ja edellisen esimerkin sarjalla on siis sama z-muunnoksen lauseke, mutta eri suppenemisalue. Näin ollen suppenemisalue liittyy olennaisesti z-muunnokseen.

4.3 Käänteinen z-muunnos

Osoitetaan seuraavaksi, että z-muunnoksella on olemassa käänteismuunnos. Tämä tulos on teoreettinen, eikä esitettävää menetelmää tulla käytännössä juuri tarvitsemaan.

Olkoon $X(z)$ signaalin $x(n)$ z-muunnos. Siis

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}.$$

Ratkaistaan tästä yhtälöstä lähtien termi $x(k)$, $k \in \mathbf{Z}$. Kertomalla yhtälön molemmat puolet lausekkeella z^{k-1} ($k \in \mathbf{Z}$), saadaan yhtälö

$$X(z) z^{k-1} = \sum_{n=-\infty}^{\infty} x(n) z^{-n+k-1}.$$

Integroidaan tämä yhtälö yli suppenemisalueella sijaitsevan r-säteisen ympyrän

$$C : |z| = r, r_- < r < r_+,$$

ja kerrotaan molemmat puolet luvulla $1/(2\pi i)$.

$$\frac{1}{2\pi i} \int_C X(z) z^{k-1} \, dz = \frac{1}{2\pi i} \int_C \left(\sum_{n=-\infty}^{\infty} x(n) z^{-n+k-1} \right) \, dz.$$

Yhtälön oikea puoli on nyt muotoa

$$\begin{aligned} & \frac{1}{2\pi i}\int_C \left(\sum_{n=-\infty}^{\infty} x(n)z^{-n+k-1}\right) dz. \\ = & \sum_{n=-\infty}^{\infty} x(n) \underbrace{\frac{1}{2\pi i}\int_C z^{-n+k-1}\, dz}_{(*)}. \end{aligned}$$

Kompleksianalyysistä saadaan tulos, jonka mukaan integraali $(*)$ saa arvon 1, kun $n = k$ ja 0 muulloin. Näin ollen alkuperäisen yhtälön oikeasta puolesta jää jäljelle lauseke $x(k)$.

Siis mikä tahansa signaalin $x(n)$ arvo on mahdollista selvittää sen z-muunnoksen avulla. Funktion $X(z)$ käänteismuunnos on

$$x(n) = \frac{1}{2\pi i}\int_C X(z)z^{n-1}\, dz.$$

Käytännössä $x(n)$ saadaan kuitenkin kätevämmin jakolaskulla tai osamurtojen avulla.

Esimerkki: Olkoon signaalin $x(n)$ z-muunnos

$$X(z) = \frac{z^{-2} + 2z^{-1} + 2}{z^{-1} + 1}.$$

Selvitä $x(n)$, kun suppenemisalue on $|z| > 1$.

Yritetään muokata lauseke sellaiseen muotoon, että se koostuu 'tutuista' z-muunnoksista.

$$\begin{aligned} & \frac{z^{-2} + 2z^{-1} + 2}{z^{-1} + 1} \\ = & \frac{z^{-1}(z^{-1} + 1) + (z^{-1} + 1) + 1}{z^{-1} + 1} \\ = & z^{-1} + 1 + \frac{1}{z^{-1} + 1} \\ = & z^{-1} + 1 + \frac{1}{1 - (-z^{-1})} \end{aligned}$$

Nyt havaitaan, että $X(z)$ koostuu signaalien $\delta(n-1)$, $\delta(n)$ ja $(-1)^n u(n)$ z-muunnosten summasta. Siis

$$x(n) = \delta(n-1) + \delta(n) + (-1)^n u(n) = \begin{cases} 0, & \text{kun } n < 0 \\ 2, & \text{kun } n = 0 \\ 0, & \text{kun } n = 1 \\ (-1)^n, & \text{kun } n > 1 \end{cases}$$

Edellä käytetty rationaalilausekkeen hajottaminen ei välttämättä onnistu aina näin helposti. Yleispätevä menetelmä on käyttää polynomien jakolaskua.

Esimerkki: Tarkastellaan signaalia $x(n)$, jonka z-muunnos on

$$X(z) = \frac{z^3 - 2z - 1}{z^3 - z^2 - z}$$

suppenemisalueenaan $z \neq 0$. Suoritetaan polynomien jakolasku jakokulmassa:

$$
\begin{array}{r|l}
z^3\text{-}z^{\,2}\text{-}z & \begin{array}{l} 1+z^{-1} \\ \hline z^3 \quad -2\ z\ -1 \\ \underline{\mp z^3 \mp z^2 \mp z} \\ \quad z^2\text{-}z \\ \quad \underline{\mp z^2 \mp z \quad \mp 1} \\ \quad\quad 0 \end{array}
\end{array}
$$

Siis $X(z)$ koostuu signaalien $\delta(n-1)$ ja $\delta(n)$ z-muunnosten summasta, joten

$$x(n) = \delta(n-1) + \delta(n) = \begin{cases} 1, & \text{kun } n = 0, 1 \\ 0, & \text{muulloin.} \end{cases}$$

Taulukossa on yleisimpien signaalien z-muunnokset.

Jono	Z-muunnos	Suppenemisalue
$\delta(n)$	1	kaikilla z
$u(n)$	$\frac{1}{1-z^{-1}}$	$\lvert z\rvert > 1$
$\delta(n-m)$	z^{-m}	kaikilla z paitsi $z = 0$ (jos $m > 0$)
$\alpha^n u(n)$	$\frac{1}{1-\alpha z^{-1}}$	$\lvert z\rvert > \lvert\alpha\rvert$

4.4 Z-muunnoksen ominaisuuksia

Z-muunnos on lineaarinen: Olkoot $X(z)$ ja $Y(z)$ jonojen $x(n)$ ja $y(n)$ z-muunnokset. Silloin jonon $ax(n) + by(n)$ z-muunnos on $aX(z) + bY(z)$. Jos $X(z)$ suppenee alueessa R_x ja $Y(z)$ alueessa R_y, niin $aX(z) + bY(z)$ suppenee *ainakin* alueessa $R_x \cap R_y$.

Jonon viivästys: Jos jonon $x(n)$ z-muunnos on $X(z)$ ja jono $w(n) = x(n-d)$ ($d \in \mathbf{N}$) niin jonon $w(n)$ z-muunnos on

$$W(z) = z^{-d}X(z).$$

Vastaava tulos on voimassa myös jonon edistämiselle, mutta se on vähemmän käytetty. Se muotoillaan seuraavasti:

$$w(n) = x(n+d) \Rightarrow W(z) = z^{d}X(z),$$

aina kun $d \in \mathbf{N}$.

Konvoluution z-muunnos: Merkitään $w(n) = x(n) * y(n)$. Silloin $W(z) = X(z)Y(z)$, ja funktion $W(z)$ suppenemisalue sisältyy funktioiden $X(z)$ ja $Y(z)$ suppenemisalueiden leikkaukseen.

Jonojen komponenteittainen tulo: Merkitään $w(n) = x(n)y(n)$. Silloin jonon $w(n)$ z-muunnos on

$$W(z) = \frac{1}{2\pi i}\int_C Y\left(\frac{z}{v}\right) X(v)v^{-1}\,dv,$$

missä C on origon kiertävä ympyrä joka on sekä funktion $X(z)$ että funktion $Y(z)$ suppenemisalueessa.

Kompleksikonjugaatti: Jos $y(n) = \overline{x(n)}$, niin $Y(z) = \overline{X(\overline{z})}$.

Parsevalin lause:

$$\sum_{n=-\infty}^{\infty} x_1(n)\overline{x_2(n)} = \frac{1}{2\pi i}\oint_C X_1(v)\overline{X_2\left(\frac{1}{\overline{v}}\right)}v^{-1}\,dv.$$

Funktion $X(z)$ derivointi: Lukujonon $nx(n)$ z-muunnos on $-zX'(z)$.

Ajan kääntäminen: Jonon $x(-n)$ z-muunnos on $X(z^{-1})$.

Alkuarvolause: Jos jono $x(n)$ on kausaalinen (eli $x(n) = 0$, kun $n < 0$), niin

$$x(0) = \lim_{|z|\to\infty} X(z).$$

Tässä siis riittää, että $|z|$ lähestyy ääretöntä mitä reittiä hyvänsä.

Osoitetaan seuraavassa konvoluution laskenta z-muunnoksen avulla. Oletetaan, että $w(n) = x(n) * y(n)$. Silloin

$$\begin{aligned}
W(z) &= \sum_{n=-\infty}^{\infty} w(n)z^{-n} \\
&= \sum_{n=-\infty}^{\infty}\left(\sum_{k=-\infty}^{\infty} x(k)y(n-k)\right)z^{-n} \\
&= \sum_{n=-\infty}^{\infty}\sum_{k=-\infty}^{\infty} x(k)y(n-k)z^{-k}z^{-(n-k)} \\
&= \sum_{k=-\infty}^{\infty}\left(x(k)z^{-k}\sum_{n=-\infty}^{\infty} y(n-k)z^{-(n-k)}\right) \\
&= \left(\sum_{k=-\infty}^{\infty} x(k)z^{-k}\right)\left(\sum_{n=-\infty}^{\infty} y(n)z^{-n}\right) \\
&= X(z)Y(z).
\end{aligned}$$

4.5 Siirtofunktio

Impulssivasteen lisäksi suodin voidaan esittää siirtofunktion avulla. Tässä kappaleessa tutustutaan siirtofunktion laskentaan FIR- ja IIR-suodinten tapauksissa ja johdetaan siirtofunktiosta muita hyödyllisiä käsitteitä.

4.5.1 FIR-suotimen siirtofunktio

Aiemmin todettiin, että lineaarinen aikainvariantti järjestelmä on täysin määrätty, kun sen impulssivaste $h(n)$ on tiedossa. Yleisempi tapa on kuitenkin käyttää impulssivasteen $z-$muunnosta $H(z)$.

Impulssivasteen avulla esitettynä herätteen $x(n)$ ja vasteen $y(n)$ välinen relaatio on muotoa

$$y(n) = h(n) * x(n).$$

Ottamalla $z-$muunnokset yhtälön molemmista puolista saadaan yhtälö

$$Y(z) = H(z)X(z). \tag{4.2}$$

Impulssivasteen $h(n)$ $z-$muunnoksesta $H(z)$ käytetään nimeä *siirtofunktio* (transfer function).

Esimerkki. Tarkastellaan järjestelmää, joka määritellään herätteen $x(n)$ ja vasteen $y(n)$ välisenä differenssiyhtälönä

$$y(n) = 0.0349x(n)+0.4302x(n-1)-0.5698x(n-2)+0.4302x(n-3)+0.0349x(n-4). \tag{4.3}$$

Järjestelmän impulssivaste on

$$h(n) = \begin{cases} 0.0349, & \text{kun } n = 0 \text{ tai } n = 4, \\ 0.4302, & \text{kun } n = 1 \text{ tai } n = 3, \\ -0.5698, & \text{kun } n = 2, \\ 0, & \text{muulloin.} \end{cases}$$

Siirtofunktio on impulssivasteen z-muunnos:

$$H(z) = \sum_{k=-\infty}^{\infty} h(k)z^{-k} = 0.0349 + 0.4302z^{-1} - 0.5698z^{-2} + 0.4302z^{-3} + 0.0349z^{-4}.$$

4.5.2 Taajuusvaste, amplitudivaste ja vaihevaste

Siirtofunktion avulla määritellään myös muita tärkeitä käsitteitä. Sijoittamalla $z = e^{i\omega}$ yhtälöön (4.2) saadaan sen erikoistapaus

$$Y(e^{i\omega}) = H(e^{i\omega})X(e^{i\omega}). \tag{4.4}$$

Tämä tarkoittaa, että signaalia suodatettaessa sen taajuudet (eli signaalin diskreettiaikainen Fourier-muunnos $X(e^{i\omega})$) kerrotaan järjestelmän taajuusvasteella $H(e^{i\omega})$. Näin ollen järjestelmän taajuuskäyttäytyminen riippuu täysin funktiosta $H(e^{i\omega})$. Tästä funktiosta käytetään nimitystä *taajuusvaste* (frequency response). Taajuusvaste on siis reaalimuuttujan ω kompleksiarvoinen funktio, joka saadaan evaluoimalla siirtofunktio kuljettaessa kompleksitason yksikköympyrä ympäri vastapäivään. Tämä johtuu siitä, että lauseke $e^{i\omega}$ käy läpi kaikki yksikköympyrän pisteet kun muuttuja $\omega \in [0, 2\pi]$.

Sijoittamalla $z = e^{i\omega}$ z-muunnoksen määritelmään saadaan taajuusvasteen laskentakaava:

$$H(e^{i\omega}) = \sum_{n=-\infty}^{\infty} h(n)e^{-i\omega n}.$$

Itse asiassa kyseessä on impulssivasteen $h(n)$ diskreettiaikainen Fourier-muunnos.

Esimerkki. Edellisen esimerkin järjestelmän taajuusvaste on

$$H(e^{i\omega}) = 0.0349 + 0.4302e^{-i\omega} - 0.5698e^{-2i\omega} + 0.4302e^{-3i\omega} + 0.0349e^{-4i\omega}.$$

Taajuusvasteen itseisarvoa $|H(e^{i\omega})|$ kutsutaan nimellä *amplitudivaste* (amplitude response, magnitude response tai gain) ja sen vaihekulmaa $\arg(H(e^{i\omega}))$ nimellä *vaihevaste* (phase response, phase shift). Amplitudivasteen merkitys käy ilmi kun otetaan yhtälöstä (4.4) itseisarvot molemmilta puolilta:

$$|Y(e^{i\omega})| = |H(e^{i\omega})| \cdot |X(e^{i\omega})|.$$

Amplitudivaste kertoo siis kuinka paljon signaalin eri taajuudet vaimenevat tai vahvistuvat suodatuksessa. Jos esimerkiksi suotimen amplitudivaste saa arvon 0.6 jollain taajuudella ω (eli $|H(e^{i\omega})| = 0.6$), muuttuu tuolla taajuudella olevan signaalin amplitudi 0.6-kertaiseksi. Jos amplitudivaste on yksi, kyseinen taajuus ei muutu suodatuksessa lainkaan (viivästymistä lukuunottamatta) ja jos amplitudivaste on nolla, taajuus häviää suodatuksessa täysin.

Vaihevasteen merkitys tulee ilmi tarkasteltaessa yhtälön (4.4) molempien puolten vaihekulmia. Kompleksilukujen kertolaskussa vaihekulmat lasketaan yhteen, joten saadaan yhtälö

$$\arg(Y(e^{i\omega})) = \arg(H(e^{i\omega})) + \arg(X(e^{i\omega})).$$

Vaihevasteesta voidaan siis päätellä kuinka paljon kukin taajuus viivästyy suodatettaessa. Jos suotimen vaihevaste on esimerkiksi $\arg(H(e^{i\omega})) = -\pi/2$ jollain taajuudella ω, muuntaa suodin signaalin $\sin(\omega n)$ signaaliksi $\sin(\omega n - \pi/2)$ (ja mahdollisesti vaimentaa tai vahvistaa sitä).

Esimerkki. Edellisen esimerkin järjestelmän taajuusvaste taajuudella $\omega = 0.05 \cdot 2\pi$ on

$$H(e^{0.05 \cdot 2\pi i}) = 0.2467 - 0.1793i.$$

Amplitudivaste tällä taajuudella on

$$|0.2467 - 0.1793i| = \sqrt{0.2467^2 + (-0.1793)^2} = 0.3050$$

ja vaihevaste

$$\arg(0.2467 - 0.1793i) = \arctan(-0.1793/0.2467) = -0.6283.$$

Jos järjestelmän herätteenä on nyt kyseisellä taajuudella värähtelevä signaali

$$x(n) = u(n) \sin(0.05 \cdot 2\pi n),$$

vaste on samantaajuinen signaali, jonka amplitudi on 0.3050-kertainen ja vaihe viivästynyt 0.6283 radiaania. Kaavamuodossa tämä signaali on

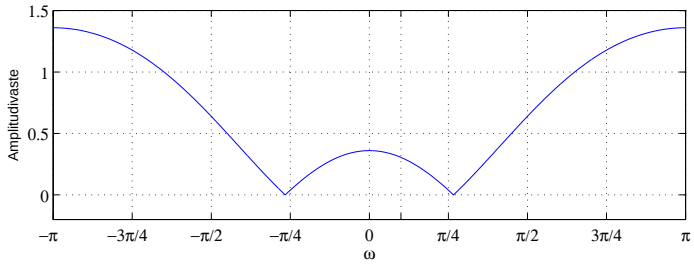
$$y(n) = 0.3050u(n) \sin(0.05 \cdot 2\pi n - 0.6283).$$

Todellinen suodatustulos poikkeaa tästä arviosta hieman suodatuksen alkuvaiheessa. Tehtävässä 4.4 havaitaan kuitenkin, että muualla arvio on tarkalleen sama kuin todellinen suodatustulos.

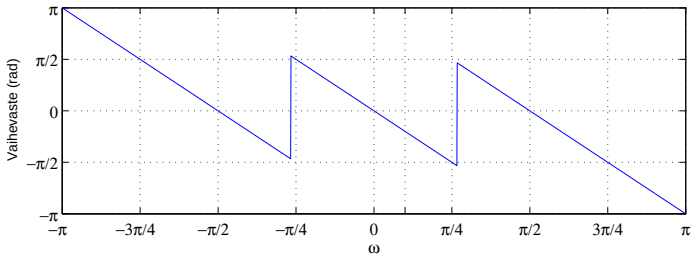
On syytä muistaa, että amplitudi- ja vaihevasteet ovat funktioita, joiden arvot riippuvat taajuusmuuttujasta ω. Edellisessä esimerkissä amplitudivasteen kaava on

$$|H(e^{i\omega})| = |0.0349 + 0.4302e^{-i\omega} - 0.5698e^{-2i\omega} + 0.4302e^{-3i\omega} + 0.0349e^{-4i\omega}|.$$

Tämän funktion kuvaaja on alla.



Vaihevasteen kuvaaja puolestaan on seuraavan näköinen. Vaihevaste tekee hyppäyksen niissä pisteissä, joissa taajuusvaste menee nollaksi. Hyppäyksen suuruus on π, ja syy tähän selviää kappaleessa 5.1.1.



Esimerkki: Tarkastellaan siirtofunktiota

$$H(z) = \frac{z^2 + 0.5z + 1}{-2z^2 + z + 0.5}.$$

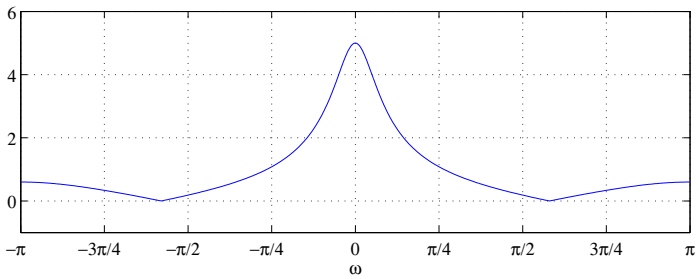
Järjestelmän taajuusvaste saadaan sijoittamalla muuttujan z paikalle lauseke $e^{i\omega}$,

$$\begin{aligned} H(e^{i\omega}) &= \frac{(e^{i\omega})^2 + 0.5e^{i\omega} + 1}{-2(e^{i\omega})^2 + e^{i\omega} + 0.5} \\ &= \frac{e^{2i\omega} + 0.5e^{i\omega} + 1}{-2e^{2i\omega} + e^{i\omega} + 0.5}. \end{aligned}$$

Amplitudivaste on nyt tämän lausekkeen itseisarvo,

$$|H(e^{i\omega})| = \frac{|e^{2i\omega} + 0.5e^{i\omega} + 1|}{|-2e^{2i\omega} + e^{i\omega} + 0.5|},$$

joka ei enää sievene. Sen kuvaaja (alla) voidaan kuitenkin piirtää Matlabilla.



Esimerkki: Olkoon $h(n) = 0.9^n u(n)$. Silloin

$$H(z) = \frac{1}{1 - 0.9z^{-1}}, \quad |z| > 0.9.$$

Siirtofunktion $H(z)$ taajuusvaste on

$$H(e^{i\omega}) = \frac{1}{1 - 0.9e^{-i\omega}},$$

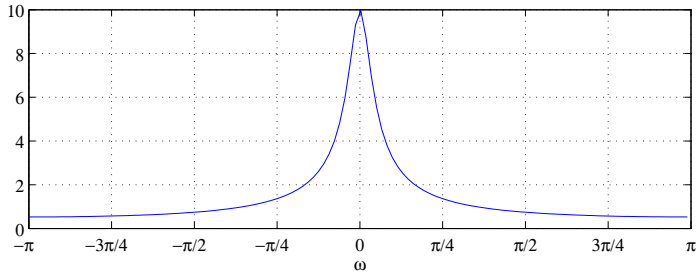
jolloin amplitudivaste on

$$\left|H(e^{i\omega})\right| = \frac{1}{|1 - 0.9e^{-i\omega}|},$$

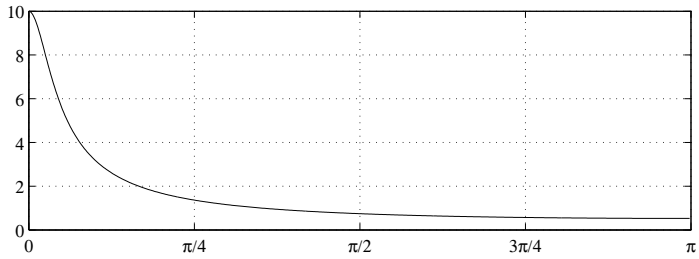
ja vaihevaste

$$\arg(H(e^{i\omega})) = \arg\left(\frac{1}{1 - 0.9e^{-i\omega}}\right).$$

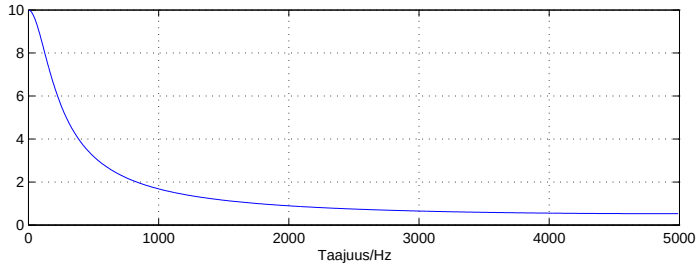
Seuraavassa kuvassa on esitetty amplitudivasteen kuvaaja.



Jos impulssivaste koostuu reaaliluvuista (kuten yleensä on), amplitudivaste on symmetrinen pystyakselin suhteen. Näin ollen tavallisesti kuvataan vain oikeanpuoleinen haara.



Kuten aiemmin mainittiin, taajuusvasteen merkitys on, että siitä nähdään, kuinka järjestelmä toimii saadessaan eritaajuisia signaaleja syötteeksi. Oletetaan, että edellisen esimerkin järjestelmä saa syötteekseen signaalin, jonka näytteenottotaajuus on 10000 Hz. Tällöin edellisen kuvaajan asteikon nollataajuus vastaa signaalia, jonka taajuus on 0 Hz, ja taajuus $\omega = \pi$ (äärimmäisenä oikealla) vastaa Nyquistin rajataajuutta 5000 Hz. Kuvaaja on siis seuraavan näköinen.



Nyt nähdään suoraan, että esimerkiksi taajuudella 2500 Hz soiva signaali pienenee amplitudiltaan noin 0.8-kertaiseksi. Vastaavasti pienet taajuudet voimistuvat.

4.5.3 Desibeliasteikko

Signaalinkäsittelyssä on usein tapana esittää amplitudivasteen kuvaajat *desibeleinä* (dB). Koska desibeliasteikko on logaritminen, käyrien kuvaajat tuovat esiin tarkemmin pieniä eroja pienillä funktion arvoilla. Desibeli määritellään seuraavasti.

Jos järjestelmän herätesignaalin teho on P_0 ja vasteen teho on P, vahvistus *beleissä* on

$$\log_{10} \frac{P}{P_0} \text{ B}.$$

Yleensä saatavat arvot ovat luokkaa 1-10 B. Pyöristysvirheistä johtuen yleisemmässä käytössä on pienempi yksikkö, desibeli, joka on kymmenesosa belistä. Desibeleissä vahvistus on

$$10 \log_{10} \frac{P}{P_0} \text{ dB}.$$

Usein signaalinkäsittelyssä työskennellään amplitudin muutoksen kanssa. Teho on suhteessa amplitudin neliöön. Jos siis herätteen amplitudi on A_0 ja vasteen A, vahvistus desibeleissä on

$$10 \log_{10} \frac{A^2}{A_0^2} = 20 \log_{10} \frac{A}{A_0} \text{ dB}.$$

Jos suhde $\frac{A}{A_0}$ on ykköstä pienempi, puhutaan vaimennuksesta.

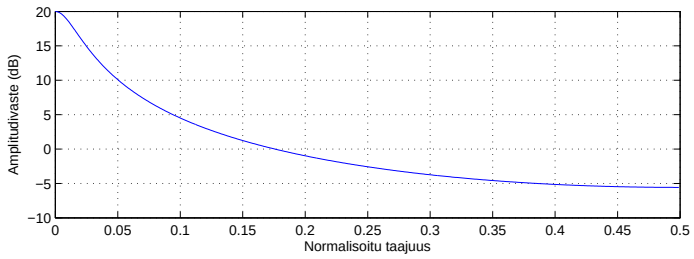
Jos järjestelmä vaimentaa signaalin amplitudin puoleen (0.5-kertaiseksi), on vaimennus desibeleinä

$$20 \log_{10} 0.5 = -6.02 \text{ dB}.$$

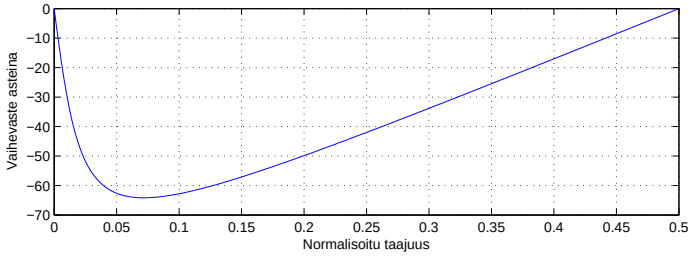
Jos taas teho vaimenee puoleen, on muutos desibeleissä

$$10 \log_{10} 0.5 = -3.01 \text{ dB}.$$

Seuraavassa kuvassa on edellisen esimerkin amplitudivaste desibeliasteikolla. Huomaa, että taajuusakseli on nyt normalisoitu näytteenottotaajuudella. Tämä on yleisin ja havainnollisin tapa taajuuksien esittämisessä. Matlab tekee vastaavan kuvan automaattisesti komennolla `freqz`.



Myös vaihevaste on tapana kuvata vain positiivisilta taajuuksilta. Myös tässä tapauksessa taajuusakseli skaalataan Nyquistin rajataajuuden mukaan ja pystyakseli muunnetaan asteiksi tai radiaaneiksi. Alla on näin kuvattuna järjestelmän $h(n) = 0.9^n u(n)$ vaihevaste.



4.5.4 IIR-suotimen siirtofunktio

Z-muunnos on tehokas työkalu myös IIR-suodinten analysoinnissa. Koska IIR-suodinten impulssivasteessa on ääretön määrä termejä, siirtofunktion laskeminen suoraan määritelmän perusteella on hankalaa. Siirtofunktion voi laskea helpommin ottamalla z-muunnos suoraan IIR-suotimen määräävästä differenssiyhtälöstä.

Tarkastellaan IIR-suotimen yleistä muotoa:

$$y(n) = \sum_{k=0}^{K} a_k x(n-k) + \sum_{m=1}^{M} b_m y(n-m).$$

Käyttämällä hyväksi z-muunnoksen lineaarisuutta ja viivästetyn signaalin z-muunnoksen kaavaa, voidaan soveltaa z-muunnosta tämän differenssiyhtälön molempiin puoliin.

$$Y(z) = \sum_{k=0}^{K} a_k X(z) z^{-k} + \sum_{m=1}^{M} b_m Y(z) z^{-m}.$$

Siirtämällä jälkimmäinen summa vasemmalle puolelle ja ottamalla $X(z)$ ja $Y(z)$ tekijäksi saadaan yhtälö

$$Y(z)\left(1 - \sum_{m=1}^{M} b_m z^{-m}\right) = X(z) \sum_{k=0}^{K} a_k z^{-k},$$

josta voidaan ratkaista ulostulon z-muunnos:

$$Y(z) = \frac{\sum_{k=0}^{K} a_k z^{-k}}{1 - \sum_{m=1}^{M} b_m z^{-m}} X(z).$$

Vertaamalla tätä kaavaan (4.2) havaitaan, että siirtofunktio on

$$H(z) = \frac{\sum_{k=0}^{K} a_k z^{-k}}{1 - \sum_{m=1}^{M} b_m z^{-m}}.$$

Esimerkki: Tarkastellaan suodinta, jonka määrittelee differenssiyhtälö

$$y(n) = 0.5x(n) - x(n-1) + 2x(n-2) - y(n-1) + y(n-2).$$

Ottamalla z-muunnokset puolittain saadaan yhtälö

$$Y(z) = 0.5X(z) - X(z)z^{-1} + 2X(z)z^{-2} - Y(z)z^{-1} + Y(z)z^{-2}.$$

Edelleen ryhmittelemällä saadaan:

$$Y(z)(1 + z^{-1} - z^{-2}) = X(z)(0.5 - z^{-1} + 2z^{-2}).$$

Siis siirtofunktio $H(z)$ on

$$H(z) = \frac{Y(z)}{X(z)} = \frac{0.5 - z^{-1} + 2z^{-2}}{1 + z^{-1} - z^{-2}} = \frac{0.5z^2 - z + 2}{z^2 + z - 1}.$$

Etsitään tämän funktion navat ja nollat. Navat ovat nimittäjän nollakohtia, eli

$$\begin{aligned} p_1 &= \frac{-1 - \sqrt{5}}{2} \\ p_2 &= \frac{-1 + \sqrt{5}}{2}. \end{aligned}$$

Nollat puolestaan ovat osoittajan nollakohtia, eli

$$\begin{aligned} z_1 &= 1 + \sqrt{3}i \\ z_2 &= 1 - \sqrt{3}i \end{aligned}$$

Seuraavassa napa-nollakuvio ja amplitudivaste. Amplitudivasteesta voidaan taas suoraan päätellä vaikkapa järjestelmän vahvistavan Nyquistin rajataajuutta noin kymmenellä desibelillä. Tämä tarkoittaa, että tuolla taajuudella amplitudin muutos a saadaan yhtälöstä

$$10\ \mathrm{dB} = 20 \log_{10} a$$

eli

$$\frac{1}{2} = \log_{10} a.$$

Korottamalla molemmat puolet kymmenen potenssiin saadaan yhtälö

$$10^{\frac{1}{2}} = a$$

eli

$$a = \sqrt{10} \approx 3.16.$$

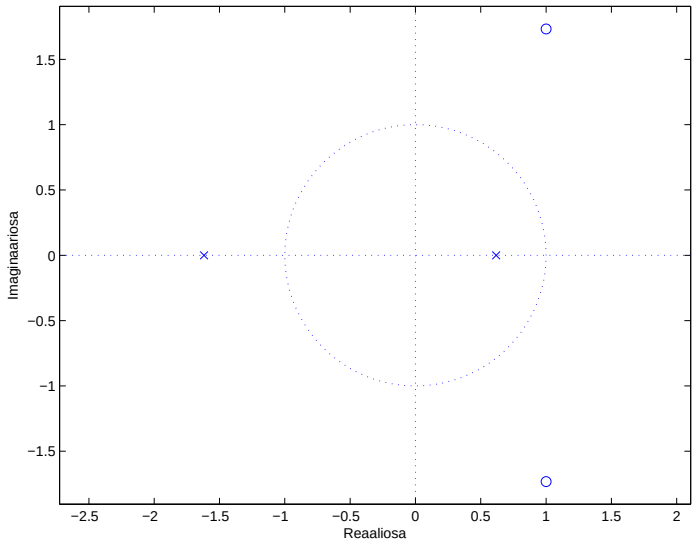
Signaalin amplitudi kasvaa siis 3.16-kertaiseksi. Jos halutaan tehon muutos P, se saadaan kaavasta

$$10\ \mathrm{dB} = 10 \log_{10} P,$$

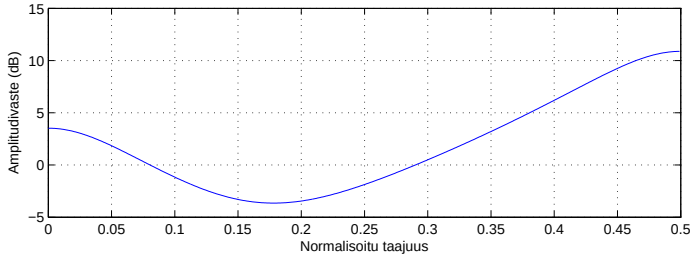
josta voidaan ratkaista

$$P = 10.$$

Teho kasvaa näin ollen kymmenkertaiseksi.



(**Matlab:** `zplane([0.5,-1,2],[1,1,-1]);`)



Toinen tavallinen tilanne on se, jossa tiedetään siirtofunktio ja halutaan differenssiyhtälö. Olkoon

$$H(z) = \frac{0.5z^2 + z + 0.25}{z^3 - z + 0.5}.$$

Ensin täytyy supistaa muuttujan z potenssit negatiivisiksi eli kertoa osoittaja ja nimittäjä luvulla z^{-3}.

$$H(z) = \frac{0.5z^{-1} + z^{-2} + 0.25z^{-3}}{1 - z^{-2} + 0.5z^{-3}}.$$

Sitten sijoitetaan $\frac{Y(z)}{X(z)} = H(z)$:

$$\frac{Y(z)}{X(z)} = \frac{0.5z^{-1} + z^{-2} + 0.25z^{-3}}{1 - z^{-2} + 0.5z^{-3}}.$$

Kerrotaan ristiin nimittäjät pois:

$$Y(z)(1 - z^{-2} + 0.5z^{-3}) = X(z)(0.5z^{-1} + z^{-2} + 0.25z^{-3})$$

eli

$$Y(z) - Y(z)z^{-2} + 0.5Y(z)z^{-3} = 0.5X(z)z^{-1} + X(z)z^{-2} + 0.25X(z)z^{-3}.$$

Jokaista termiä vastaava signaali tiedetään:

$$y(n) - y(n-2) + 0.5y(n-3) = 0.5x(n-1) + x(n-2) + 0.25x(n-3).$$

Tämä saadaan helposti muotoon, joka voidaan jo toteuttaa algoritmisesti:

$$y(n) = 0.5x(n-1) + x(n-2) + 0.25x(n-3) + y(n-2) - 0.5y(n-3).$$

4.6 Stabiilisuus

IIR-suotimen stabiilisuus voidaan helposti tutkia napa-nollakuvion avulla. Voidaan nimittäin osoittaa[1], että (kausaalinen) IIR-suodin on stabiili, jos ja vain jos kaikki sen siirtofunktion navat ovat yksikköympyrän sisäpuolella.

Esimerkiksi aiemmin esillä ollut suodin

$$y(n) = 0.5x(n) - x(n-1) + 2x(n-2) - y(n-1) + y(n-2)$$

ei ole stabiili, koska navat ovat

$$p_{1,2} = \frac{-1 \pm \sqrt{5}}{2},$$

joista toinen on yksikköympyrän ulkopuolella ($|\frac{-1-\sqrt{5}}{2}| > 1$).

Otetaan vielä yksi esimerkki. Suotimen

$$y(n) = x(n) - 2x(n-1) + 2x(n-2) + \frac{1}{2}y(n-1) - \frac{1}{8}y(n-2)$$

[1]Stabiilisuusehdon mukaan LTI-järjestelmä on stabiili jos ja vain jos

$$\sum_{k=-\infty}^{\infty} |h(k)| < \infty.$$

Siirtofunktion

$$H(z) = \sum_{k=-\infty}^{\infty} h(k) z^{-k}$$

suppenemisalueen muodostavat ne pisteet $z \in \mathbf{C}$, joissa

$$\sum_{k=-\infty}^{\infty} |h(k) z^{-k}| < \infty.$$

Tästä seuraa rationaalisen siirtofunktion tapauksessa, että *järjestelmä on stabiili, jos ja vain jos yksikään siirtofunktion napa ei ole yksikköympyrän kehällä.* Lisäksi kausaaliselle järjestelmälle $h(n) = 0$, kun $n < 0$, jolloin siirtofunktion z-muunnos koostuu ainoastaan muuttujan z ei-positiivisista potensseista. Siis kun muuttujan z itseisarvo kasvaa, niin z-muunnoksen itseisarvo pienenee. Näin ollen, jos kausaalisella järjestelmällä ylipäätään on suppenemisalue, on se origosta kaukaisimman navan ulkopuolella. Nämä ehdot yhdistämällä saadaan väite.

siirtofunktio on

$$\begin{aligned}
& y(n) - \frac{1}{2}y(n-1) + \frac{1}{8}y(n-2) = x(n) - 2x(n-1) + 2x(n-2) \\
\Leftrightarrow\ & Y(z) - \frac{1}{2}Y(z)z^{-1} + \frac{1}{8}Y(z)z^{-2} = X(z) - 2X(z)z^{-1} + 2X(z)z^{-2} \\
\Leftrightarrow\ & Y(z)(1 - \frac{1}{2}z^{-1} + \frac{1}{8}z^{-2}) = X(z)(1 - 2z^{-1} + 2z^{-2}) \\
\Leftrightarrow\ & \frac{Y(z)}{X(z)} = \frac{1 - 2z^{-1} + 2z^{-2}}{1 - \frac{1}{2}z^{-1} + \frac{1}{8}z^{-2}} = \frac{z^2 - 2z + 2}{z^2 - \frac{1}{2}z + \frac{1}{8}} \qquad (= H(z)).
\end{aligned}$$

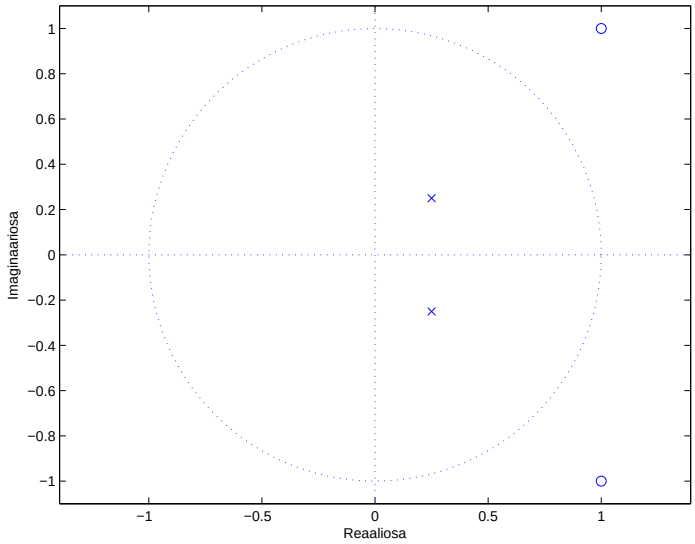
Navat ovat polynomin

$$z^2 - \frac{1}{2}z + \frac{1}{8}$$

nollakohdat, eli

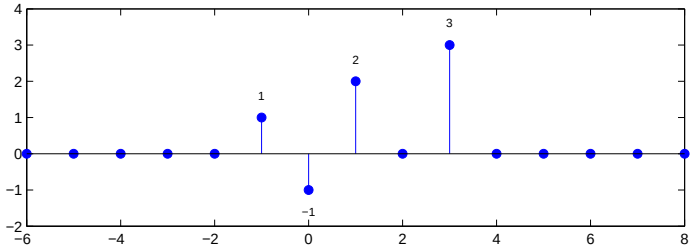
$$p_{1,2} = \frac{\frac{1}{2} \pm \sqrt{\frac{1}{4} - 4 \cdot \frac{1}{8}}}{2} = \frac{\frac{1}{2} \pm \sqrt{-\frac{1}{4}}}{2} = \frac{\frac{1}{2} \pm i\sqrt{\frac{1}{4}}}{2} = \frac{\frac{1}{2} \pm \frac{i}{2}}{2} = \frac{1}{4} \pm \frac{i}{4}.$$

Molemmat ovat yksikköympyrän sisällä ($|p_{1,2}| < 1$), joten suodin on stabiili. Järjestelmän napa-nollakuvio on alla olevassa kuvassa.

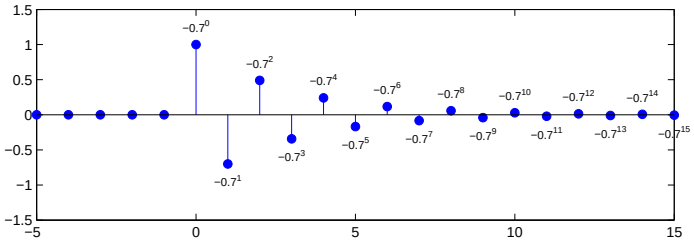


Harjoitustehtäviä

4.1. (a) Laske alla olevan kuvan mukaisen signaalin z-muunnoksen lauseke.



(b) Laske alla olevan kuvan mukaisen signaalin z-muunnoksen lauseke.



(c) Laske (a) ja (b) -kohdan signaalien diskreettiaikaisen Fourier-muunnoksen lausekkeet.

4.2. (a) Olkoon kausaalinen LTI-järjestelmä määritelty yhtälöllä $y(n) = x(n) - 2x(n-2) + x(n-3)$. Onko järjestelmä FIR vai IIR?

(b) Olkoon kausaalinen LTI-järjestelmä määritelty yhtälöllä $y(n) + 2y(n-1) = x(n) + x(n-1)$. Onko järjestelmä FIR vai IIR?

4.3. Olkoon N jokin positiivinen kokonaisluku. Laske lukujonon

$$x(n) = \begin{cases} 0, & \text{kun } n < 0, \\ n, & \text{kun } 0 \leq n \leq N, \\ N, & \text{kun } n > N \end{cases}$$

z-muunnos. *Vihje: käytä z-muunnoksen derivaatan kaavaa tai muodosta differenssiyhtälö, jonka ratkaisu x(n) on ja ota z-muunnokset puolittain.*

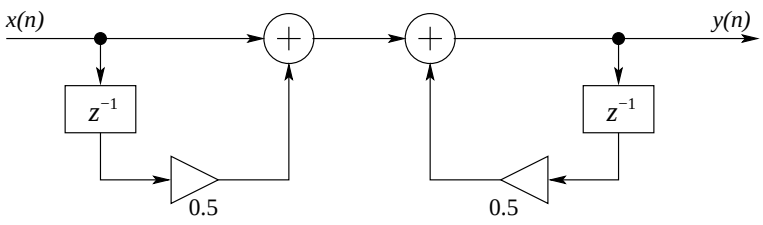
4.4. *(Matlab)* Suodata signaali $x(n) = \sin(0.05 \cdot 2\pi n)$ sivulla 61 olevan kaavan (4.3) mukaisella suotimella. Vertaa tulosta arvioituun vasteeseen $y(n) = 0.3050 \sin(0.05 \cdot 2\pi n - 0.6283)$. Tulosta samaan ikkunaan alkuperäinen signaali ja arvioitu sekä todellinen vaste.

4.5. Signaali $x(n) = 0.7u(n) \sin(0.2 \cdot 2\pi n)$ suodatetaan järjestelmällä, jonka impulssivaste on

$$h(n) = \begin{cases} -\frac{1}{4}, & \text{kun } n = 0 \text{ tai } n = 2, \\ \frac{1}{2}, & \text{kun } n = 1, \\ 0, & \text{muulloin.} \end{cases}$$

Vaste on muotoa $y(n) = Au(n) \sin(0.2 \cdot 2\pi n + \phi)$. Määritä reaaliluvut A ja ϕ.

4.6. Tarkastellaan alla olevan kuvan LTI-järjestelmää. Onko kysessä FIR- vai IIR-suodatin? Määritä järjestelmän amplitudivaste. Sievennä saamasi lauseke reaaliseen muotoon.



4.7. Oletetaan, että kausaalisen LTI-järjestelmän heräte $x(n)$ ja vaste $y(n)$ toteuttavat seuraavan differenssiyhtälön:

$$y(n-2) - \frac{10}{3}y(n-1) + y(n) = x(n) - \frac{1}{2}x(n-1) + \frac{1}{4}x(n-2).$$

(a) Määritä järjestelmän siirtofunktio $H(z)$.

(b) Piirrä napa-nollakuvio.

(c) Onko suodin stabiili?

4.8. Oletetaan, että kausaalisen järjestelmän heräte $x(n)$ ja vaste $y(n)$ toteuttavat differenssiyhtälön

$$y(n) = x(n) - 2x(n-1) + \frac{5}{4}x(n-2) + y(n-1) - \frac{5}{16}y(n-2).$$

(a) Määritä siirtofunktio.

(b) Piirrä napa-nollakuvio.

(c) Onko järjestelmä stabiili?

4.9. Olkoon kausaalisen LTI-järjestelmän siirtofunktio

$$H(z) = \frac{z^{-2} + z^{-1} + 1}{z^{-2} + \frac{1}{4}}.$$

Määritä sitä vastaava differenssiyhtälö.

4.10. Erään järjestelmän siirtofunktion navat ovat $p_1 = 0.9$, $p_2 = 0.7+0.7i$ ja $p_3 = 0.7-0.7i$. Nollat ovat $z_1 = -1$, $z_2 = i$ ja $z_3 = -i$. Lisäksi tiedetään, että nollataajuudella ($\omega = 0$) järjestelmän taajuusvaste $H(e^{i\omega}) = 1$. Mikä on siirtofunktion $H(z)$ lauseke?

4.11. *(Matlab)* Erään LTI-järjestelmän siirtofunktio on

$$H(z) = \frac{0.0122 + 0.0226z^{-1} + 0.0298z^{-2} + 0.0204z^{-3} + 0.0099z^{-4}}{1 - 0.9170z^{-1} + 0.0540z^{-2} - 0.2410z^{-3} + 0.1990z^{-4}}.$$

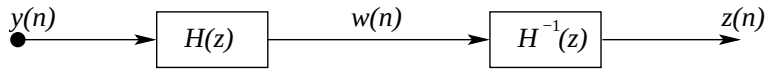
Sijoita kertoimet vektoreihin `a` ja `b` ja piirrä Matlabilla järjestelmän napa-nollakuvio (`help zplane`), amplitudi- ja vaihevasteet (`help freqz`) sekä impulssivaste (`help impz`). Vertaa näitä käänteisen järjestelmän

$$H^{-1}(z) = \frac{1 - 0.9170z^{-1} + 0.0540z^{-2} - 0.2410z^{-3} + 0.1990z^{-4}}{0.0122 + 0.0226z^{-1} + 0.0298z^{-2} + 0.0204z^{-3} + 0.0099z^{-4}}$$

vastaaviin. Huomaa että voit käyttää alkuperäistä järjestelmää hyväksesi käänteistä järjestelmää tutkiessasi. Älä siis kirjoita kaikkia kertoimia uudelleen.

4.12. *(Matlab)* Lataa signaali "laughter" komennolla `load laughter`. Signaali on tämän jälkeen muuttujassa `y`. Kuuntele signaali komennolla `sound(y)`. Suodata signaali edellisen tehtävän suotimella $H(z)$ (`help filter`). Kuuntele tulos (alla olevan

kuvan signaali $w(n)$. Sovella edellisen tehtävän käänteissuodinta $H^{-1}(z)$ tulossignaaliin. Vertaa tulosta (alla olevan kuvan signaali $z(n)$) alkuperäiseen "laughter"-signaaliin.



4.13. a) Signaalin amplitudi (jännite) muuttuu seuraavilla kertoimilla[2]. Muunna arvot desibeleiksi. 0.5, 0.2, 0.1, 0.01, 0.001, 2.

b) Signaalin vaimennus on seuraavien desibeliarvojen mukainen. Miten amplitudi muuttuu? -25 dB, -3 dB, -6 dB, -12 dB, -30 dB, -60 dB.

4.14. a) Signaalin teho muuttuu seuraavilla kertoimilla. Muunna arvot desibeleiksi. 0.5, 0.2, 0.1, 0.01, 0.001, 2.

b) Signaalin vaimennus on seuraavien desibeliarvojen mukainen. Miten teho muuttuu? -25 dB, -3 dB, -6 dB, -12 dB, -30 dB, -60 dB.

4.15. *(Matlab)* Eräs LTI-järjestelmä toteutetaan differenssiyhtälöllä

$$\begin{aligned} y(n) &= 1.143y(n-1) - 0.4128y(n-2) + 0.0675x(n) + 0.1349x(n-1) \\ &+ 0.0675x(n-2) \end{aligned}$$

järjestelmässä, jossa näytteenottotaajuus on 20000 Hz. Kuinka suuri vaimennus (desibeleinä) on taajuudella 5000 Hz värähtelevällä signaalilla kun se syötetään järjestelmään? *Vihje: Laske ensin kulmataajuus* $\omega = 2\pi f/F_s$*, sen jälkeen sitä vastaava kompleksitason piste* $z = e^{i\omega}$*, ja lopuksi siirtofunktion* $H(z)$ *arvo tässä pisteessä. Funktion* $H(z)$ *lauseke on laskettava käsin.*

4.16. *(Matlab)* Luo signaali `y`, jonka taajuus nousee tasaisesti komennoilla

```
t=0:1/8192:4;
y=chirp(t,0,1,1000);
```

Kuuntele tulos (mahdollisuuksien mukaan) komennolla `soundsc(y)`. Vaihtoehtoisesti voit tutkia signaalia komennolla `spectrogram`, joka näyttää aika- ja taajuusakseleilla kuvan. Suodata signaali tehtävän 4.15. suotimella. Kuuntele tulos ja/tai katso sen spektrogrammi. Vertaa tulosta suotimen amplitudivasteeseen.

4.17. Lukujonon z-muunnos on

$$X(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 + 4z^{-1} + 4z^{-2}}.$$

Piirrä sen napa-nollakuvio.

[2]Esimerkiksi kerroin 0.2 tarkoittaa, että amplitudi muuttuu 0.2-kertaiseksi alkuperäiseen nähden.

4.18. Laske z-muunnos lukujonolle

$$x(n) = \begin{cases} \left(\frac{1}{2}\right)^n, & \text{kun } n \geq 5, \\ 0, & \text{muulloin.} \end{cases}$$

4.19. Suodin

$$y(n) = \frac{1}{4}x(n) - \frac{1}{2}x(n-1) + \frac{1}{4}x(n-2)$$

toteutetaan laitteistossa, jonka näytteenottotaajuus on 16000 Hz. Mikä on suotimen amplitudivaste (eli vahvistus / vaimennus) 4000 Hertsin taajuudella? *Vihje:* Laske $H(z)$ ja $H(e^{i\omega})$, sijoita kulmataajuus ω saamaasi kaavaan ja ota itseisarvo. Taajuutta f vastaava kulmataajuus on

$$\omega = \frac{2\pi f}{F_s},$$

missä F_s on näytteenottotaajuus.

Luku 5

Suotimen suunnittelu taajuustasossa

Edellisessä kappaleessa havaittiin FIR-suotimella olevan selkeä tulkinta taajuusalueella. Suodin muuttaa kunkin taajuuskomponentin amplitudia ja vaihetta, eli vaimentaa tai vahvistaa sekä viivästää sitä. FIR-suotimen käyttökohde määräytyy tätä kautta: suotimella voidaan vaimentaa tiettyjä taajuuksia, ja vastaavasti vahvistaa toisia taajuuksia. Usein vastaan tuleva erikoistapaus tästä on suodin, joka poistaa täysin tietyn taajuuskaistan ja säilyttää toisen kaistan alkuperäisenä. Taajuuksilla operoivalla suotimella on useita käyttökohteita. Sen avulla voidaan esimerkiksi poistaa signaalista jokin kapealla taajuusalueella oleva häiriö, jakaa signaali kahtia pieniin ja suuriin taajuuksiin tehokkaampaa kompressiota varten tai vaikkapa korostaa pieniä ja suuria taajuuksia ja näin kompensoida halpojen kaiuttimien aiheuttamaa vääristymää.

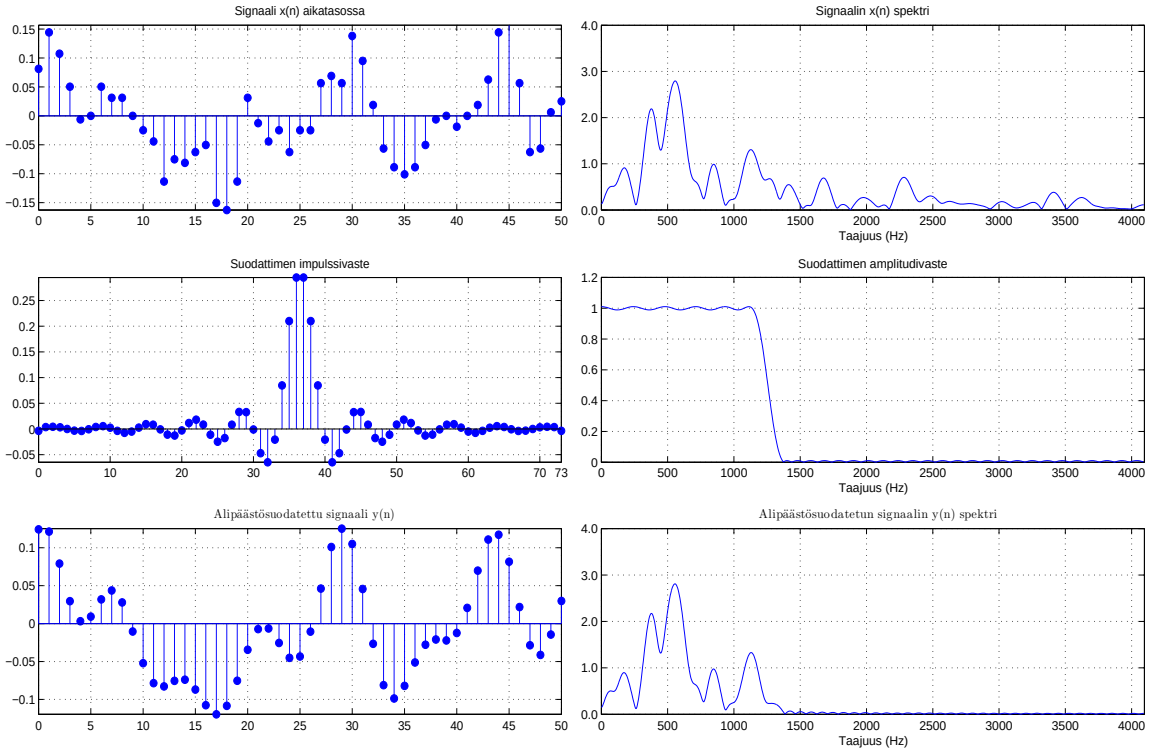
Kuten z-muunnosta ja Fourier-muunnosta tarkasteltaessa havaittiin, konvoluutio aikatasossa vastaa kertolaskua z- tai taajuustasossa. Kääntäen voidaan ilmaista, että taajuustason kertolasku on mahdollista toteuttaa aikatason konvoluutiona. Konvoluution kertoimet eli suotimen impulssivaste täytyy siis suunnitella sellaiseksi, että taajuusvaste on halutunlainen. Suunnittelutehtävä voisi siis olla esimerkiksi seuraava: selvitä sellaisen suotimen impulssivaste, jonka amplitudivaste on yksi taajuuksilla 0 Hz – 1100 Hz ja nolla taajuuksilla 1400 Hz – 4096 Hz, mikä on samalla Nyquistin rajataajuus. Tällainen suodin säilyttäisi taajuudet 1100 Hertsiin asti alkuperäisinä ja poistaisi 1400 Hertsiä suuremmat taajuudet. Tässä kappaleessa tutustutaan yhteen menetelmään, jolla voidaan laskea sopivat impulssivasteen kertoimet.

Signaalin $x(n)$ sisältämät taajuudet käyvät ilmi sen diskreettiaikaisesta Fourier-muunnoksesta $X(e^{i\omega})$. Kun signaali suodatetaan suotimella, jonka taajuusvaste on $H(e^{i\omega})$, saadaan tuloksen $y(n)$ taajuussisältö yhtälöstä

$$Y(e^{i\omega}) = H(e^{i\omega})X(e^{i\omega}).$$

Alla olevissa kuvissa on esimerkki suodatuksen vaikutuksesta signaalin taajuuksiin. Ylimmässä kuvaparissa on vasemmalla eräs testisignaali ja oikealla sen sisältämät taajuudet. Keskimmäisessä kuvaparissa on vasemmalla erään suotimen impulssivaste ja oikealla sen amplitudivaste. Suodin on nyt suunniteltu yllä olevien vaatimusten mukaiseksi, eli säilyttämään pienet taajuudet ja poistamaan suuret. Suodatettaessa testisignaali tällä suotimella (eli laskemalla signaalin ja impulssivasteen konvoluutio) tapahtuu samalla taajuustasossa kertolasku. Konvoluution tuloksessa (alin kuvapari) ovat suuret taajuudet todellakin poistuneet ja pienet taajuudet säilyneet ennallaan. Tämä johtuu siitä, että suurilla

taajuuksilla amplitudivaste on lähellä nollaa ja pienillä taajuuksilla lähellä ykköstä. Nollalla kertominen poistaa ja ykkösellä kertominen säilyttää taajuudet.

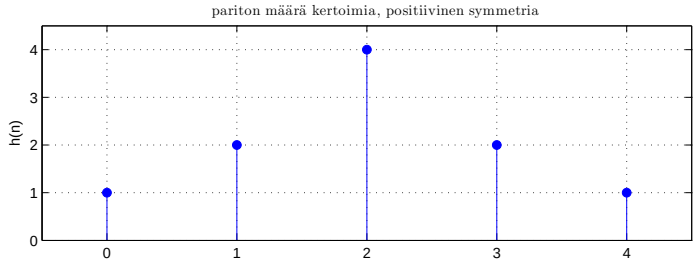


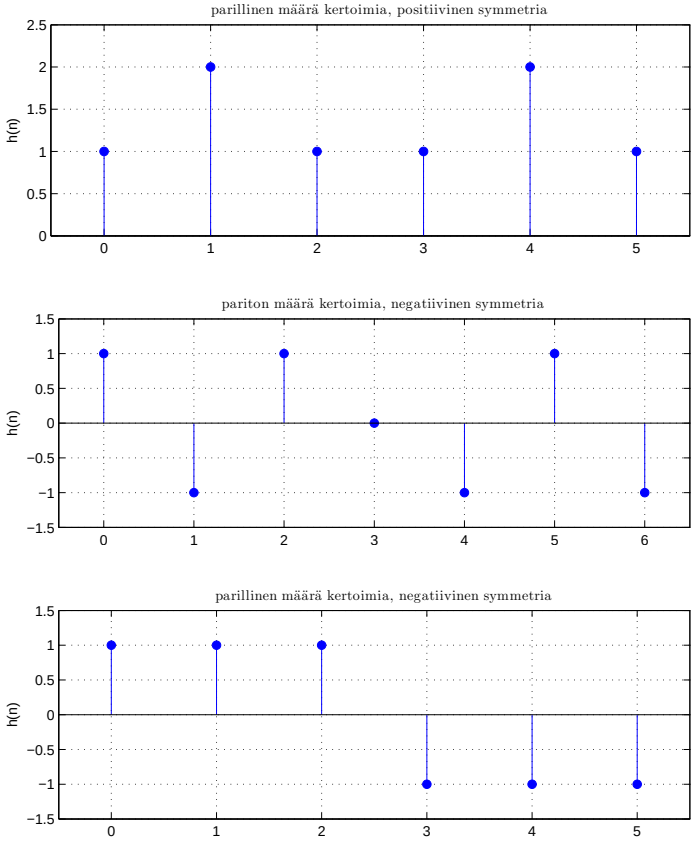
5.1 FIR-suodinten suunnittelu: suunnittelukriteerit

FIR-suodinten suunnitteluvaatimukset esitetään tavallisesti taajuustasossa. Vaatimukset voidaan jakaa amplitudivasteelle asetettaviin vaatimuksiin ja vaihevasteelle asetettaviin vaatimuksiin.

5.1.1 Vaihevasteen vaatimukset

Vaihevasteelle asetettavat vaatimukset ovat useimmiten yksinkertaisempia ja tyypillinen vaatimus onkin lineaarinen vaihevaste. Jos vaihevaste on lineaarinen, kaikki taajuudet viivästyvät saman verran. Tämä vaatimus on helppo toteuttaa tarkastelemalla ainoastaan sellaisten FIR-suodinten luokkaa, joiden impulssivaste on negatiivisesti tai positiivisesti symmetrinen (s.o. $h(n) = \pm h(N - n - 1)$). Näin saadaan neljä eri tapausta. Näitä tapauksia vastaavat kuvat ovat alla.





Edellisessä kappaleessa suotimen vaihevaste määriteltiin taajuusvasteen vaihekulmaksi: $\arg(H(e^{i\omega}))$. Esimerkiksi ylimmän kuvan suotimen vaihevaste lasketaan seuraavasti. Suotimen impulssivaste on

$$h(n) = \delta(n) + 2\delta(n-1) + 4\delta(n-2) + 2\delta(n-3) + \delta(n-4).$$

Tämän suotimen taajuusvaste on

$$H(e^{i\omega}) = 1 + 2e^{-i\omega} + 4e^{-2i\omega} + 2e^{-3i\omega} + e^{-4i\omega},$$

ja sen termit voidaan ryhmitellä niin että vaihevaste näkyy selvästi:

$$\begin{aligned} H(e^{i\omega}) &= (1 + e^{-4i\omega}) + 2(e^{-i\omega} + e^{-3i\omega}) + 4e^{-2i\omega} \\ &= e^{-2i\omega}\left[(e^{2i\omega} + e^{-2i\omega}) + 2(e^{i\omega} + e^{-i\omega}) + 4\right] \\ &= e^{-2i\omega}\left[2\cos(2\omega) + 4\cos(\omega) + 4\right]. \end{aligned}$$

On helpohkoa osoittaa että taajuuksilla $\omega \in [0, \pi]$ hakasulkeiden sisällä oleva lauseke on reaalinen ja positiivinen, joten vaihevaste on sama kuin kompleksitermin $e^{-2i\omega}$ vaihekulma, eli -2ω.

Tässä vaiheessa saattaa herätä kysymys, eikö vaihevasteen tulisi olla vakio, jotta kaikki taajuudet viivästyisivät yhtä paljon. Vastaus on kuitenkin kielteinen, sillä vaihevaste ilmaisee montako astetta tai radiaania kyseinen taajuuskomponentti viivästyy. Esimerkiksi neljännesjakson (90°) viive pienellä taajuudella on ajassa paljon enemmän kuin sama viive suurella taajuudella. Vaihevasteen derivaatta kertoo montako näytettä viive on. Tästä

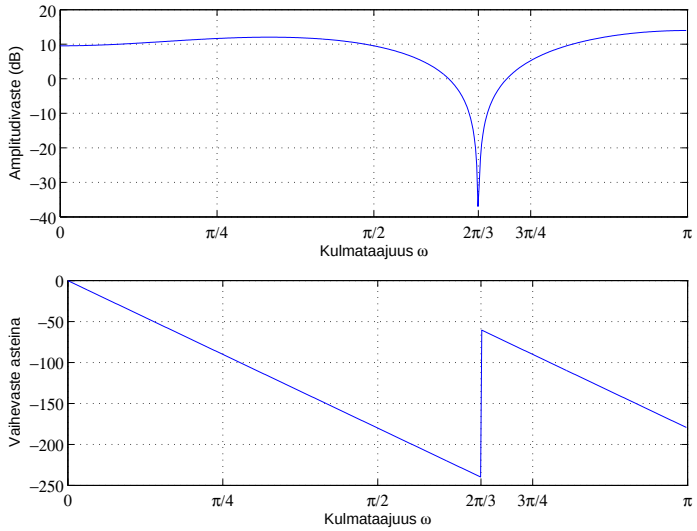
käytetään nimitystä *ryhmäviive* ja se määritellään kaavalla $\tau(\omega) = -\frac{d\theta(\omega)}{d\omega}$, missä $\theta(\omega)$ on vaihevaste taajuudella ω.

Ryhmäviiveen avulla voidaan määritellä lineaarisen vaihevasteen käsite riittävän yleisesti: *vaihevaste on lineaarinen, jos ryhmäviive on vakio kaikissa pisteissä, joissa se on määritelty.* Tämä määritelmä hyväksyy myös suotimet, joiden vaihevasteessa on epäjatkuvuuskohta. Vaihevastetta ei nimittäin ole määritelty tapauksessa $H\left(e^{i\omega}\right) = 0$, koska luvun nolla vaihekulma on määrittelemätön. Tällaisten pisteiden ympäristössä vaihevasteeseen tulee yleensä π:n korkuinen hyppäys.

Esimerkiksi suotimella

$$h(n) = -\delta(n) + 2\delta(n-1) + \delta(n-2) + 2\delta(n-3) - \delta(n-4)$$

on alla olevan kuvan mukaiset amplitudi- ja vaihevasteet.



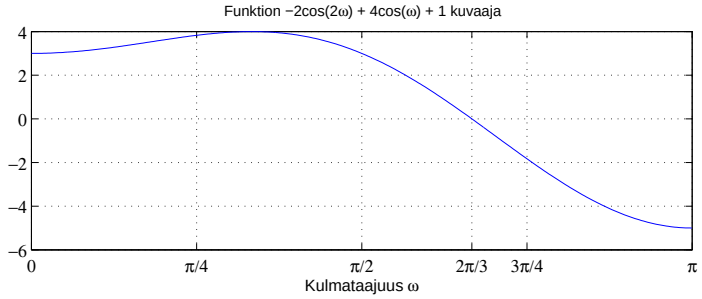
Suotimen amplitudivaste on nolla taajuudella $\omega = \frac{2\pi}{3}$, ja vaihevasteeseen tulee epäjatkuvuuskohta tällä taajuudella. Ryhmäviive on kuitenkin vakio kaikissa pisteissä paitsi taajuudella $\omega = \frac{2\pi}{3}$. Tämä nähdään laskemalla siirtofunktio

$$H(z) = -1 + 2z^{-1} + z^{-2} + 2z^{-3} - z^{-4}$$

ja taajuusvaste

$$\begin{aligned} H(e^{i\omega}) &= -1 + 2e^{-i\omega} + e^{-2i\omega} + 2e^{-3i\omega} - e^{-4i\omega} \\ &= (-1 - e^{-4i\omega}) + 2(e^{-i\omega} + e^{-3i\omega}) + e^{-2i\omega} \\ &= e^{-2i\omega}\left[(-e^{2i\omega} - e^{-2i\omega}) + 2(e^{i\omega} + e^{-i\omega}) + 1\right] \\ &= e^{-2i\omega}\left[-2\cos(2\omega) + 4\cos(\omega) + 1\right] \end{aligned}$$

Hakasulkeissa oleva termi on reaalinen, joten koko lausekkeen vaihekulma määräytyy alussa olevan eksponenttilausekkeen $e^{-2i\omega}$ sekä hakasuluissa olevan lausekkeen etumerkin perusteella. Hakasulkulausekkeen kuvaaja on alla.



Kuvasta nähdään, että lauseke on positiivinen välillä $[0, \frac{2\pi}{3})$ ja negatiivinen välillä $(\frac{2\pi}{3}, \pi]$. Näin ollen vaihevasteen lauseke on seuraava:

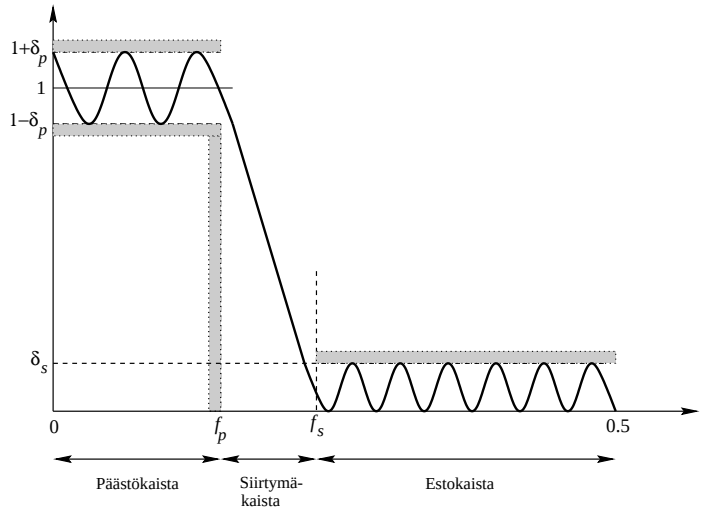
$$\arg(H(e^{i\omega})) = \begin{cases} -2\omega, \text{ kun } \omega < \frac{2\pi}{3}, \\ -2\omega + \pi, \text{ kun } \omega > \frac{2\pi}{3}. \end{cases}$$

Lausekkeeseen tulee vakio π kertoimen ollessa negatiivinen, koska $-1 = e^{\pi i}$. Ryhmäviive on nyt

$$\tau(\omega) = -\frac{d}{d\omega}\arg(H(e^{i\omega})) = 2, \text{ kun } \omega \neq \frac{2\pi}{3}.$$

5.1.2 Amplitudivasteen vaatimukset

Amplitudivasteelle vaatimukset kertovat mitkä taajuudet poistetaan ja mitkä säilytetään. Nämä esitetään *päästö-*, *esto-* ja *siirtymäkaistan* (passband, stopband, transition band) kautta. Tarkastellaan oheista kuvaa.



Kuvassa siis vaaka-akselilla on taajuudet, tässä tapauksessa esitettynä normalisoituina näytteenottotaajuuden suhteen. Pystyakseli puolestaan esittää suotimen amplitudivastetta. Tässä tapauksessa suodin on määritelty seuraavien parametrien avulla.

δ_p	Päästökaistan maksimipoikkeama
δ_s	Estokaistan maksimipoikkeama
f_p	Päästökaistan rajataajuus
f_s	Estokaistan rajataajuus

Päästökaista muodostuu siis tässä tapauksissa taajuuksista, joissa amplitudivaste halutaan lähelle ykköstä ($f \in [0, f_p]$), estokaista taajuuksista, joissa amplitudivaste halutaan lähelle

nollaa ($f \in [f_s, 0.5]$) ja siirtymäkaista taajuuksista tällä välillä ($f \in (f_p, f_s)$). Päästökaistalla sallitaan, että amplitudivasteen arvot poikkeavat ideaaliarvosta 1 enintään luvun δ_p verran, ja estokaistalla enintään luvun δ_s verran ideaaliarvosta 0.

Tyypilliset määrittelyt saattaisivat olla esimerkiksi seuraavat:

δ_p	0.026 dB
δ_s	-30 dB
f_p	5000 Hz
f_s	6000 Hz
Näytteenottotaajuus	16000 Hz

Muuntaminen normalisoiduiksi taajuuksiksi tapahtuu jakamalla taajuudet f_p ja f_s näytteenottotaajuudella. Normalisoituina vaatimukset ovat:

f_p	$5000/16000 = \frac{5}{16}$
f_s	$6000/16000 = \frac{3}{8}$

Tästä esitysmuodosta päästään edelleen kulmataajuuksiin skaalaamalla lukuarvot välille $[0, 2\pi]$, ts. kertomalla luvulla 2π.

ω_p	$2\pi \cdot 5000/16000 = \frac{5\pi}{8}$ rad
ω_s	$2\pi \cdot 6000/16000 = \frac{3\pi}{4}$ rad

5.2 Suunnittelu ikkunamenetelmällä

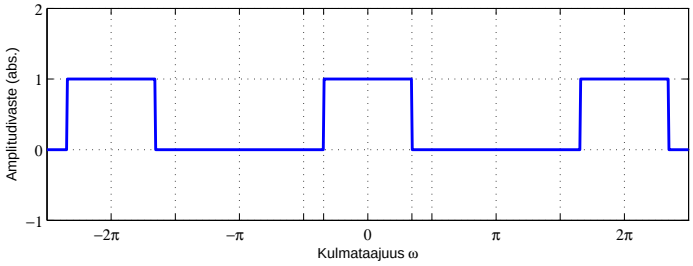
Ensimmäinen mieleen tuleva tapa on yksinkertaisesti keksiä sellainen amplitudivasteen kuvaaja, että se toteuttaa annetut ehdot. Tämä kuitenkin tuottaa ongelmia, kuten kohta tulemme näkemään. Tarkastellaan esimerkiksi edellä annettuja ehtoja:

δ_p	0.026 dB
δ_s	-30 dB
ω_p	$2\pi \cdot 5000/16000 = \frac{5\pi}{8}$ rad
ω_s	$2\pi \cdot 6000/16000 = \frac{3\pi}{4}$ rad

Nämä ehdot toteuttaa esimerkiksi taajuusvaste, joka putoaa ykkösestä nollaan ω_p:n ja ω_s:n puolivälissä, eli kulmataajuudella $\frac{11\pi}{32}$:

$$H(e^{i\omega}) = \begin{cases} 1, & \omega < \frac{11\pi}{32} \\ 0, & \omega \geq \frac{11\pi}{32}. \end{cases}$$

Kuvassa nähdään tämän taajuusvasteen kuvaaja, jossa on huomioitu myös vasteen periodisuus ja konjugaattisymmetrisyys välillä $[0, 2\pi]$.



Voidaan osoittaa[1] yleisesti, että taajuusvaste

$$H(e^{i\omega}) = \begin{cases} 1, & \text{kun } \omega < \omega_c, \\ 0, & \text{kun } \omega \geq \omega_c \end{cases}$$

löytyy suotimelta, jonka impulssivaste on

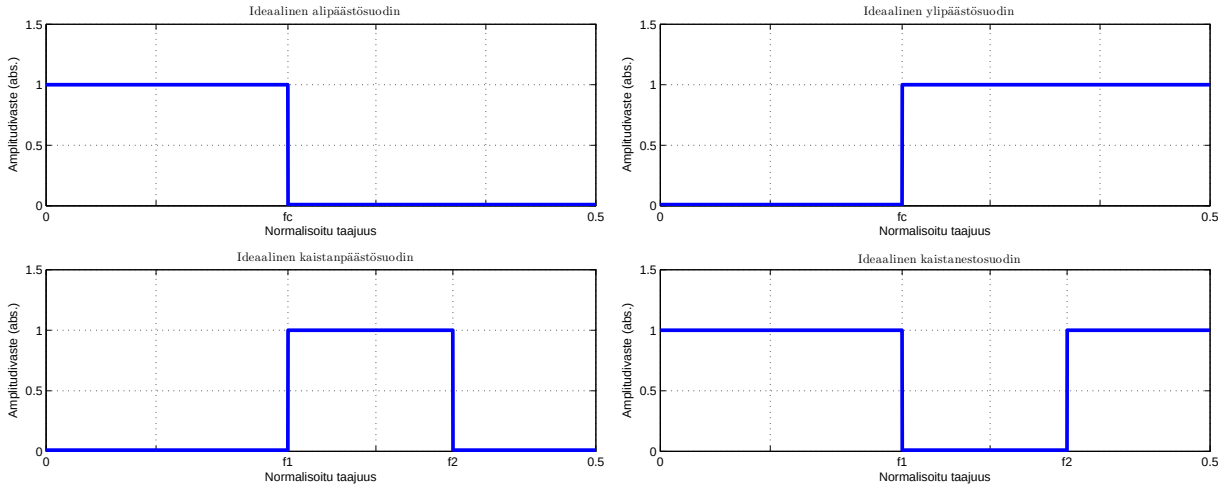
$$h(n) = \begin{cases} 2f_c \operatorname{sinc}(\omega_c n), & \text{kun } n \neq 0, \\ 2f_c, & \text{kun } n = 0. \end{cases}$$

Tässä funktio $\operatorname{sinc}(x) = \sin(x)/x$. Ongelmana tämän suotimen toteuttamisessa on impulssivasteen ääretön pituus. Käytännössä se täytyy aina katkaista, mikä puolestaan aiheuttaa ongelmia taajuusvasteessa. Siitä tarkemmin jatkossa.

Seuraavassa taulukossa on esitetty vastaavat ideaaliset impulssivasteet erityyppisille suotimille: *alipäästösuotimille* (taajuudet välillä $[0, f_c]$ päästetään läpi), *ylipäästösuotimille* (taajuudet välillä $[f_c, 0.5]$ päästetään läpi), *kaistanpäästösuotimille* (taajuudet välillä $[f_1, f_2]$ päästetään läpi) ja *kaistanestosuotimille* (taajuudet välillä $[f_1, f_2]$ poistetaan).

Suodintyyppi	Impulssivaste kun	
	$n \neq 0$	$n = 0$
Alipäästö	$2f_c\operatorname{sinc}(n \cdot 2\pi f_c)$	$2f_c$
Ylipäästö	$-2f_c\operatorname{sinc}(n \cdot 2\pi f_c)$	$1 - 2f_c$
Kaistanpäästö	$2f_2\operatorname{sinc}(n \cdot 2\pi f_2) - 2f_1\operatorname{sinc}(n \cdot 2\pi f_1)$	$2(f_2 - f_1)$
Kaistanesto	$2f_1\operatorname{sinc}(n \cdot 2\pi f_1) - 2f_2\operatorname{sinc}(n \cdot 2\pi f_2)$	$1 - 2(f_2 - f_1)$

Näitä suodintyyppejä vastaavien ideaalisten taajuusvasteiden kuvaajat ovat alla.



[1] Todistus tapahtuu ottamalla taajuusvasteen käänteinen diskreettiaikainen Fourier-muunnos:

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{i\omega})e^{i\omega n}\,d\omega = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} 1 \cdot e^{i\omega n}\,d\omega,$$

joka voidaan melko helposti saada muotoon

$$h(n) = \begin{cases} 2f_c \operatorname{sinc}(\omega_c n), & \text{kun } n \neq 0, \\ 2f_c, & \text{kun } n = 0. \end{cases}$$

Ideaalisen suotimen impulssivasteen suoraviivainen katkaiseminen on luonnollisin menetelmä pyrkiä lähelle ideaalisen alipäästösuotimen ominaisuuksia. Näin saatu impulssivaste vastaa ideaalista impulssivastetta kerrottuna 'ikkunasignaalilla'

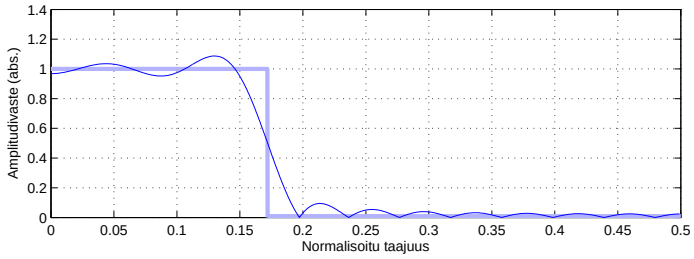
$$w(n) = \begin{cases} 1, & \text{kun } -M < n < M, \\ 0, & \text{muulloin.} \end{cases}$$

Kyseisestä ikkunafunktiosta käytetään nimeä *suorakulmainen ikkuna* (*engl. rectangular window*). Ikkunan kertoimien kokonaismäärästä käytetään merkintää N, ja se riippuu rajasta M kaavan $N = 2M + 1$ mukaan. Toisaalta raja M saadaan jakamalla N kahdella ja alaspäin pyöristämällä, t.s. $M = \lfloor \frac{N}{2} \rfloor$.

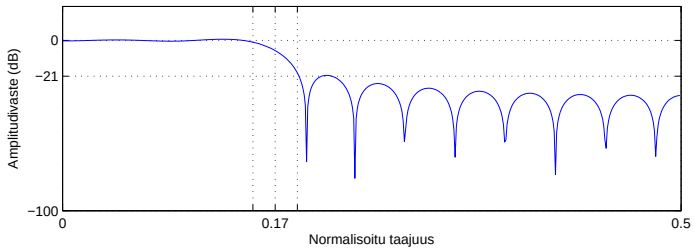
Katkaistun impulssivasteen $h_t(n) = w(n)h(n)$ käyttäytymistä taajuustasossa voidaan approksimoida sen diskreetin Fourier-muunnoksen avulla. Tulomuodossa olevan signaalin DFT voidaan ilmaista konvoluution avulla:

$$H_t(n) = \frac{1}{N} W(n) * H(n).$$

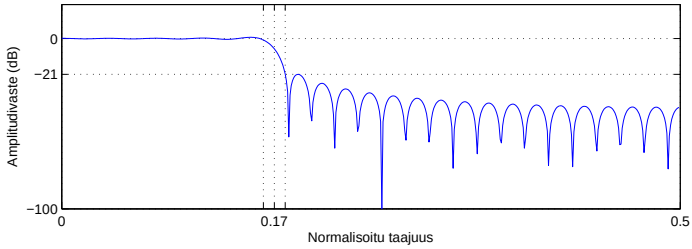
Suoran katkaisun vaikutus taajuustasossa on siis konvoluutio ikkunafunktion $w(n)$ diskreetin Fourier-muunnoksen kanssa. Toisaalta $W(n)$ on vastaavaa muotoa kuin $h(n)$, siinä on nimittäin tärkeällä sijalla sinc-funktio. Konvoluutio tällaisen signaalin kanssa aiheuttaa alkuperäiseen taajuusvasteeseen värähtelyä ja Gibbsin ilmiön, missä värähtelyä on erityisesti taajuuskaistojen reunoilla. Gibbsin ilmiö oli esillä jo Fourier-sarjan yhteydessä. Alla olevassa kuvassa on ideaalisen alipäästösuotimen amplitudivaste impulssivasteen katkaisun jälkeen. Kyseessä on edellä ollut esimerkki, jossa rajataajuus oli $\omega_c = \frac{11\pi}{32}$, eli normalisoituna $f_c = \frac{11}{64} \approx 0.17$. Impulssivasteeseen on otettu mukaan $N = 25$ termiä, eli ideaalinen impulssivaste pisteissä $-12, -11, -10, \ldots, 10, 11, 12$.



Kuvasta nähdään selvästi ylimääräinen värähtely rajataajuuden ympärillä. Erityisen selvästi tämä värähtely näkyy desibeliasteikolla, jossa korkein huippu on -21 desibelin kohdalla.



Kokeilemalla eri mittaisia impulssivasteita havaitaan, ettei pituuden N kasvattaminen paranna vaimennusominaisuuksia lainkaan: amplitudivasteen korkein huippu estokaistalla pysyy -21:ssä desibelissä. Alla oleva amplitudivaste saadaan kaksinkertaistamalla impulssivasteen pituus 51:een. Ainoa vaikutus on estokaistan huippujen sekä siirtymäkaistan kapeneminen.



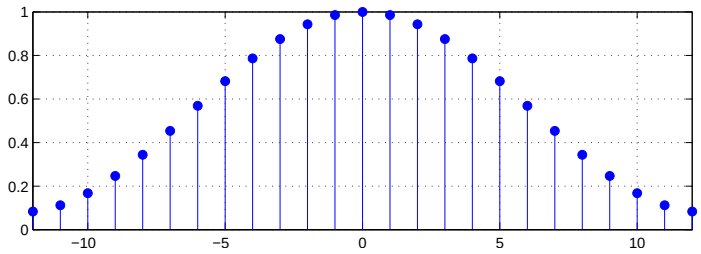
Myöskään päästökaistan värähtelyyn ei voida näin vaikuttaa, vaan sitäkin on aina vakiomäärä, noin 0.74 dB. Sen sijaan suurempi N kaventaa siirtymäkaistaa, ja siirtymäkaistan leveyden Δf ja kertoimien määrän välillä onkin voimassa suorakulmaisen ikkunan tapauksessa kaava

$$\Delta f = \frac{0.9}{N}.$$

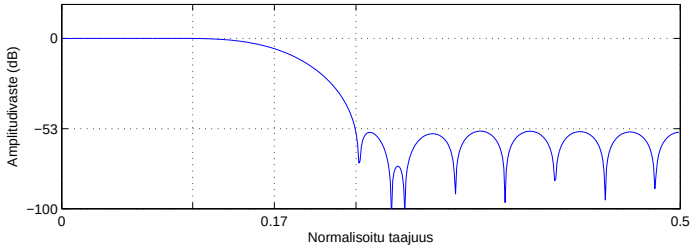
Suotimen vaimennusominaisuuksiin ei siis voida vaikuttaa kertoimia lisäämällä, mutta niitä voidaan parantaa käyttämällä jotain pehmeämmin laskevaa ikkunafunktiota suoran katkaisun asemesta. Yksi usein käytetyistä ikkunafunktioista on *Hamming-ikkuna* (Hamming window), joka määritellään seuraavasti:

$$w(n) = \begin{cases} 0.54 + 0.46\cos(\frac{2\pi n}{N}), & \text{kun } -M < n < M, \\ 0, & \text{muulloin.} \end{cases}$$

Alla olevassa kuvassa on Hamming-ikkunan kertoimien kuvaaja tapauksessa $N = 25$.



Hamming-ikkunaa käytetään siis niin, että yllä olevalla lausekkeella $w(n)$ kerrotaan vastaava ideaalisen impulssivasteen termi. Alla olevassa kuvassa on näin saatava amplitudivaste suunniteltaessa vastaavaa alipäästösuodinta kuin aikaisemmassakin esimerkissä.



Amplitudivasteen kuvaajasta havaitaan Hamming-ikkunan parantavan vaimennusominaisuuksia. Nyt suodin vaimentaa kaikkia estokaistan taajuuksia vähintään 53 dB. Lisäksi päästökaistan värähtely ei ole enää paljaalla silmällä havaittavissa. Kokeilemalla eri kertoimien määriä havaitaan korkeimman estokaistalla olevan huipun pysyvän 53 dB:ssä sekä päästökaistan värähtelyn pysyvän noin 0.019 desibelissä. Lisäksi siirtymäkaistan leveys näyttäisi nytkin olevan kääntäen verrannollinen kertoimien määrään nähden. Siirtymäkaista on selvästi leveämpi kuin vastaava suorakulmaista ikkunaa käyttäen saatu siirtymäkaista ja osoittautuukin, että leveys riippuu kertoimien määrästä kaavan

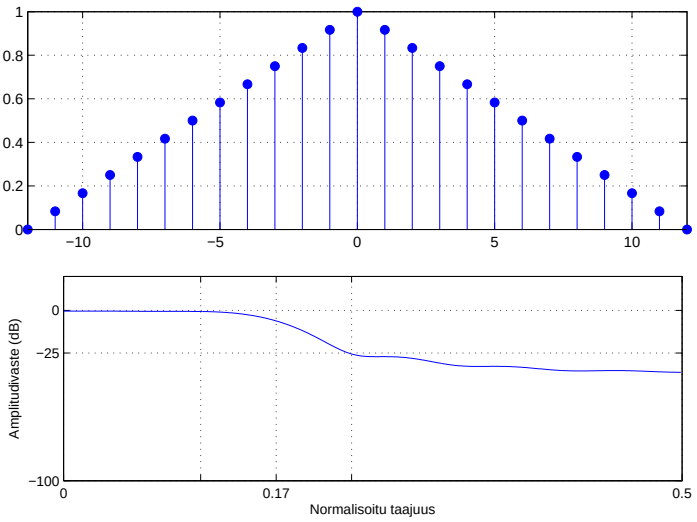
$$\Delta f = \frac{3.3}{N}$$

mukaisesti. Ikkunan valinta on näin ollen aina kompromissi vaimennusominaisuuksien ja kertoimien määrän välillä. Hamming-ikkuna tarvitsee nimittäin aina suuremman määrän kertoimia (noin 3.7-kertaisen määrän) suorakulmaiseen ikkunaan verrattuna, jotta siirtymäkaistat olisivat yhtä leveät.

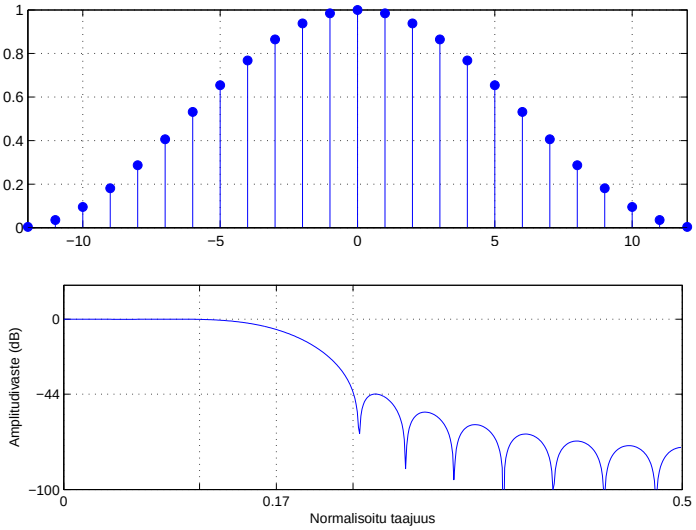
Muita tavallisesti käytettyjä ikkunafunktioita on lueteltu alla. Taulukosta voidaan ratkaista myös tarvittavien kertoimien määrä, kun halutaan jokin tietty normalisoitu siirtymäkaistan leveys. Mitä paremmat ikkunalla saatavat vaimennus- ja värähtelyominaisuudet ovat, sitä suurempi kerroin siirtymäkaistan leveyden kaavassa on ja sitä enemmän kertoimia tarvitaan saman siirtymäkaistan saamiseksi.

Ikkuna-funktion nimi	Siirtymäkaistan leveys (normalisoitu)	Päästökaistan värähtely (dB)	Estokaistan minimi-vaimennus (dB)	Ikkunan lauseke $w(n)$, kun $\lvert n\rvert \leq (N-1)/2$
Suorakulmainen	0.9/N	0.7416	21	1
Bartlett	3.05/N	0.4752	25	$1 - \frac{2\lvert n\rvert}{N-1}$
Hanning	3.1/N	0.0546	44	$0.5 + 0.5\cos\left(\frac{2\pi n}{N}\right)$
Hamming	3.3/N	0.0194	53	$0.54 + 0.46\cos\left(\frac{2\pi n}{N}\right)$
Blackman	5.5/N	0.0017	74	$0.42 + 0.5\cos\left(\frac{2\pi n}{N}\right)$ $+0.08\cos\left(\frac{4\pi n}{N}\right)$

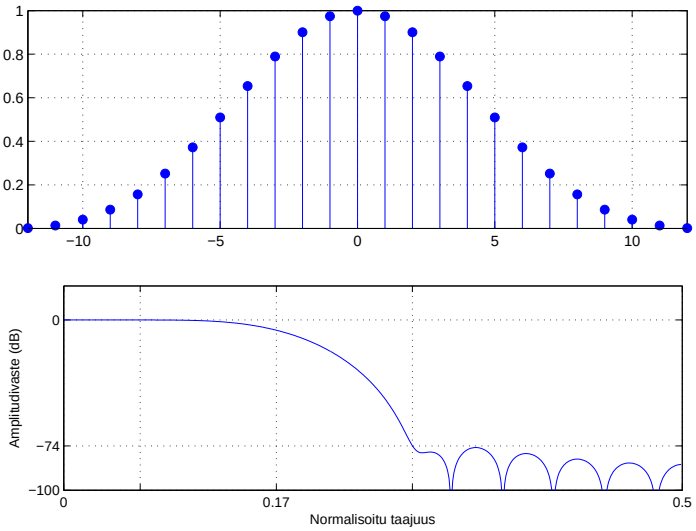
Taulukon ikkunoista Bartlett-ikkuna on harvemmin suodinsuunnittelussa käytetty, kolmionmuotoinen ikkunafunktio. Alla on Bartlett-ikkunan kuvaaja ja sitä käyttäen suunnitellun alipäästösuotimen amplitudivaste.



Hanning- eli Hann-ikkuna on melko lähellä Hamming-ikkunaa. Kuitenkin sen vaimennus- ja värähtelyominaisuudet ovat hieman huonommat ja kertoimien määrä on toisaalta pienempi. Alla on Hanning-ikkunan kuvaaja ja sitä käyttäen suunnitellun alipäästösuotimen amplitudivaste.



Taulukossa alimpana on Blackman-ikkuna, jonka lauseke koostuu kahdesta kosinitermistä. Näin saadaan aikaan erittäin hyvä minimivaimennus sekä päästökaistan värähtely, mutta vastaavasti tarvittavien kertoimien määrä on suurempi. Alla on Blackman-ikkunan kuvaaja ja sitä käyttäen suunnitellun alipäästösuotimen amplitudivaste.



5.3 Yhteenveto ikkunamenetelmän käytöstä

- Määritä ideaalinen taajuusvaste suotimelle.
- Laske tätä taajuusvastetta vastaava ideaalinen impulssivaste käänteisen Fourier-muunnoksen avulla. Tavallisimmat impulssivasteet on lueteltu edellä olleessa taulukossa.

- Valitse sellainen ikkunafunktio, joka täyttää halutut vaatimukset päästökaistalla ja estokaistalla. Selvitä myös tarvittavien kertoimien määrä N normalisoidusta siirtymäkaistan leveydestä Δf lähtien.
- Suunnitellun suotimen impulssivaste on

$$h_t(n) = w(n)h(n),$$

  Missä $h(n)$ on ideaalinen impulssivaste ja $w(n)$ on valittu ikkunafunktio.

Esimerkki. Edellisessä kappaleessa oli esitetty alipäästösuotimelle seuraavat vaatimukset:

δ_p	0.026 dB
δ_s	-30 dB
f_p	5000 Hz
f_s	6000 Hz
Näytteenottotaajuus	16000 Hz

Koska $\delta_p = 0.026$ dB, ainoat mahdolliset ikkunat ovat Hamming-ikkuna ($\delta_p = 0.0194$ dB) ja Blackman-ikkuna ($\delta_p = 0.0017$ dB). Näistä valitsemme Hamming-ikkunan, sillä sitä käytettäessä tarvittavien kertoimien määrä on Blackman-ikkunaa pienempi:

$$N = \begin{cases} \frac{3.3}{\Delta f}, & \text{Hamming-ikkunalle} \\ \frac{5.5}{\Delta f}, & \text{Blackman-ikkunalle.} \end{cases}$$

Taulukosta nähdään lisäksi, että Hamming-ikkunaa käyttäen on mahdollista täyttää myös annetut estokaistan vaatimukset. Sarakkeesta *Estokaistan minimivaimennus* nähdään nimittäin, että minimivaimennus on 53 dB, kun vaatimuksena oli ainoastaan $\delta_s = -30$ dB (-53 dB $<$ -30 dB). Seuraavaksi täytyy selvittää suotimen kertoimien määrä. Se saadaan siirtymäkaistan leveydestä. Meidän tapauksessamme siirtymäkaistan tulee olla välillä 5000-6000 Hz. Näytteenottotaajuudella normalisoituina tämä on väli $[5000/16000, 6000/16000]$, joten siirtymäkaistan normalisoitu leveys on $\Delta f = 1000/16000 = 1/16$. Koska Hamming-ikkunaa käytettäessä

$$\Delta f = \frac{3.3}{N},$$

niin tarvittavien kertoimien määrä N on tarkoilla arvoilla

$$N = \frac{3.3}{\Delta f} = \frac{3.3}{1/16} = 3.3 \cdot 16 = 52.8.$$

Tämä pyöristetään[2] ylöspäin lukuun $N = 53$.

Ideaalinen impulssivaste on alipäästösuotimen tapauksessa

$$h(n) = \begin{cases} 2f_c \operatorname{sinc}(n \cdot 2\pi f_c), & n \neq 0 \\ 2f_c, & n = 0, \end{cases}$$

[2]Tällä kurssilla pyöristys tehdään aina suurempaan *parittomaan* kokonaislukuun. Alipäästö- ja kaistanpäästösuotimilla myös parillinen kertoimien määrä on mahdollinen.

eli meidän tapauksessamme

$$h(n) = \begin{cases} 2 \cdot 5500/16000 \,\mathrm{sinc}\,(2\pi \cdot 5500/16000 \cdot n), & n \neq 0 \\ 2 \cdot 5500/16000, & n = 0, \end{cases}$$

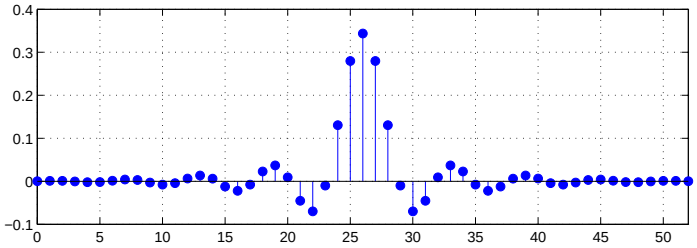
Tässä luku $f_c = \frac{5500}{16000} = \frac{11}{32}$ on valittu siirtymäkaistan puolivälistä. Todellinen impulssivaste saadaan nyt kertomalla tämä $h(n)$ Hamming-ikkunalla

$$h_t(n) = \begin{cases} (0.54 + 0.46\cos(2\pi n/53)) \cdot \\ 2 \cdot 11/32 \cdot \mathrm{sinc}\,(2\pi \cdot 11/32 \cdot n), & 0 < |n| \leq 26 \\ 2 \cdot 11/32 \cdot, & n = 0, \\ 0, & \text{muulloin,} \end{cases}$$

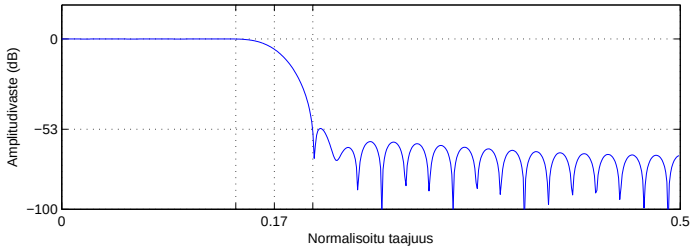
joka sievenee muotoon

$$h_t(n) = \begin{cases} (11/16) \cdot (0.54 + 0.46\cos(2\pi n/53)) \cdot \mathrm{sinc}\,(11\pi n/16), & 0 < |n| \leq 26 \\ 11/16, & n = 0, \\ 0, & \text{muulloin.} \end{cases}$$

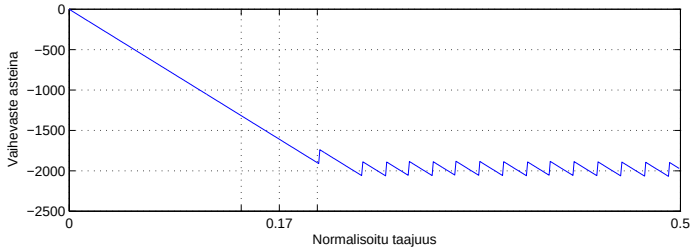
Saadaksemme kausaalisen suotimen, meidän on siirrettävä tätä impulssivastetta vielä 26 askelta oikealle (viivästys). Tämä ei vaikuta tulokseen muuten kuin 26 askeleen viivästymisenä, taajuustason käyttäytyminen pysyy ennallaan. Näin saatavan impulssivasteen kuvaaja on alla.



Tämän suotimen amplitudivasteen kuvaaja on seuraavanlainen.

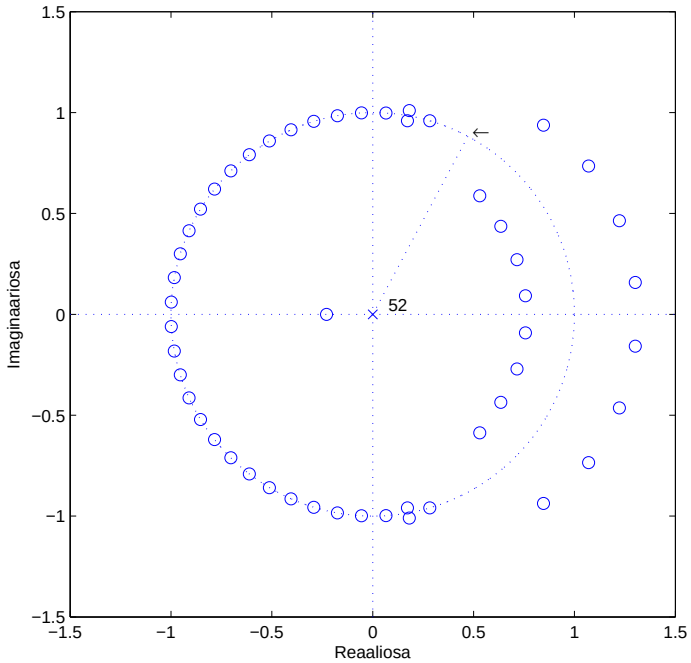


Vaihevaste on alla. Huomaa lineaarisuus päästökaistalla. Ryhmäviive on $\tau(\omega) = 26$, eli suodin viivästää kaikkia taajuuksia 26 askelta.



Lopuksi vielä nollakuvio. Kuvasta puuttuu kauimmainen nolla, joka on pisteessä $z = 4.37$. Kuviosta näkyy selvästi kuinka Fourier-muunnoksen taajuusakseli vastaa yksikköympyrän ylempää puolikasta eli pisteitä $\{e^{i\omega} \mid \omega \in [0, \pi]\}$. Taajuusvaste taajuudella ω saadaan evaluoimalla siirtofunktio $H(z)$ pisteessä $z = e^{i\omega}$. Kuvassa on merkitty katkoviivalla siirtymäkaistan puoliväliä vastaava vaihekulma $\omega_c = \frac{11\pi}{32}$, ja taajuusvaste tällä taajuudella saadaan evaluoimalla siirtofunktio nuolella merkityssä pisteessä.

Tätä pienemmät taajuudet saadaan kiertämällä yksikköympyrää myötäpäivään ja suuremmat vastapäivään. Nollataajuus on kompleksitason pisteessä $1 + 0i$ ja Nyquistin rajataajuus pisteessä $-1 + 0i$. Näin ollen kaikki siirtofunktion nollat sijaitsevat yksikköympyrällä vaihekulmaa $\omega_c = \frac{11\pi}{32}$ suuremmassa kulmassa ja jokainen niistä aiheuttaakin amplitudivasteeseen yhden alaspäin ulottuvan piikin.



Toisena esimerkkinä suunnitellaan ylipäästösuodin, kun vaaditaan, että

δ_p	0.06 dB
δ_s	-28 dB
f_s	9 kHz
f_p	10 kHz
Näytteenottotaajuus	48 kHz

Taulukosta nähdään, että suorakulmainen ja Bartlett-ikkuna eivät käy, ja Hanning-ikkuna on ensimmäinen, joka toteuttaa päästökaistan $(0.0546 < 0.06)$ ja estokaistan $(-44 \text{ dB} < -28 \text{ dB})$ vaatimukset. Koska Hanning-ikkunalle tarvitaan vähemmän kertoimia kuin muilla sopivilla, käytetään sitä. Tällöin siis

$$\Delta f = \frac{3.1}{N}.$$

Näytteenottotaajuudella normalisoitu siirtymäkaistan leveys on

$$\Delta f = \frac{10 \text{ kHz}}{48 \text{ kHz}} - \frac{9 \text{ kHz}}{48 \text{ kHz}} = \frac{1}{48}.$$

Nyt tarvittavien kertoimien määrä on

$$N = \frac{3.1}{\Delta f} = \frac{3.1}{1/48} = 148.8,$$

joka pyöristetään lukuun $N = 149$. Valitaan luku f_c normalisoidun siirtymäkaistan [9/48, 10/48] puolivälistä eli luku $f_c = 9.5/48$. Silloin saadaan

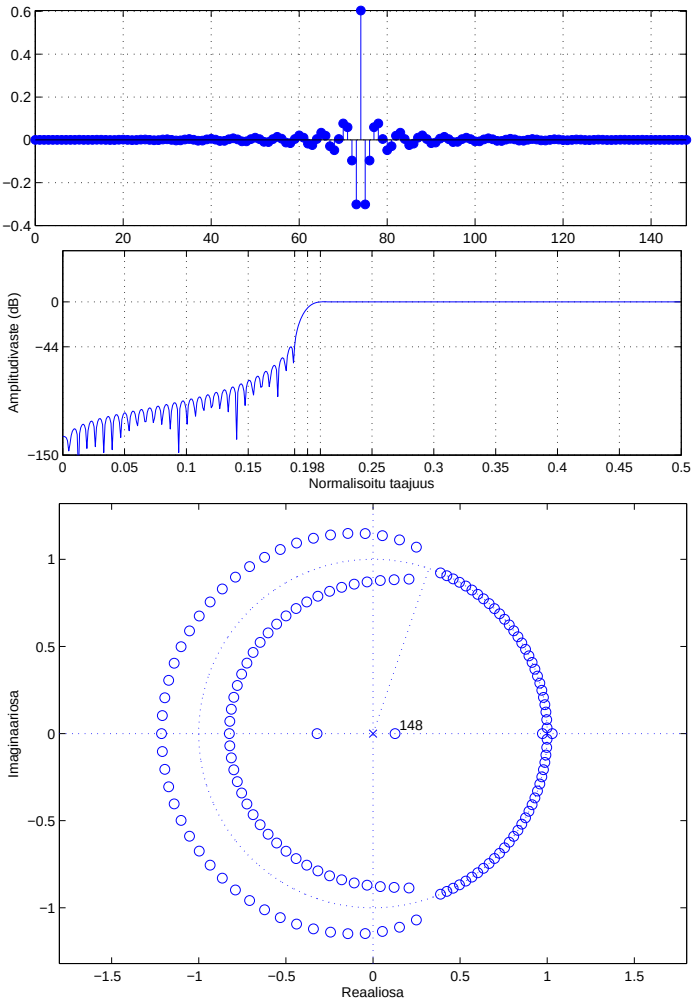
$$h(n) = \begin{cases} -2 \cdot \frac{9.5}{48} \mathrm{sinc}(2\pi \cdot \frac{9.5}{48} n), & n \neq 0, \\ 1 - 2 \cdot \frac{9.5}{48}, & n = 0. \end{cases}$$

Tämä lauseke on vielä kerrottava Hanning-ikkunan lausekkeella, jolloin todelliseksi impulssivasteeksi saadaan sievennettynä

$$h_t(n) = \begin{cases} -(0.5 + 0.5\cos(\frac{2\pi n}{149}))(\frac{19}{48}\mathrm{sinc}(\frac{19}{48}\pi n)), & 0 < |n| \leq 74 \\ \frac{29}{48}, & n = 0, \\ 0, & \text{muulloin.} \end{cases}$$

Jälleen on siirrettävä tätä impulssivastetta 74 askelta oikealle, jotta saadaan kausaalinen suodin.

Tämän impulssivasteen ja amplitudivasteen kuvaajat ovat alla, ja lopuksi nollakuvio. Nollakuviosta puuttuvat kauimmaiset nollat, jotka ovat pisteissä $z = 7.9116$ ja $z = -3.1176$.



	0.2952	0.3262	0.3606
0.1702	1	2	3
0.1880	4	5	6
0.2080	7	8	9
0.2297	$*$	0	#

Taulukko 5.1: DTMF-taajuudet (dual tone multiple frequency) näppäinpuhelimelle kun näytteistystaajuus on 8192 Hz. Esimerkiksi näppäintä '5' vastaava signaali on $x(n) = \sin(0.1880\pi n) + \sin(0.3262\pi n)$.

Harjoitustehtäviä

5.1. Suotimen impulssivaste on $h(n) = \delta(n-4)$. Laske sen vaihevasteen ja ryhmäviiveen lausekkeet.

5.2. Suotimen impulssivaste on $h(n) = \delta(n-1)+\delta(n-2)$. Laske sen vaihevasteen ja ryhmäviiveen lausekkeet. Vinkki: ota taajuusvasteen lausekkeessa yhteiseksi tekijäksi $e^{-\frac{3}{2}i\omega}$.

5.3. *(Matlab)* Lataa tiedosto

`http://www.cs.tut.fi/kurssit/SGN-1200/numero.mat`

itsellesi. Tiedosto sisältää näppäinpuhelimen valintaäänen (seitsemän numeroa). Tehtäväsi on selvittää mikä puhelinnumero on kyseessä.

Lataa signaali Matlabiin komennolla `load numero.mat`. Signaali on tämän jälkeen vektorissa nimeltä `salainen`. Voit kuunnella sen komennolla `sound(salainen)`.

Valintaäänet koostuvat aina kahden eritaajuisen komponentin summasta (taulukko 5.1). Tunnista nämä komponentit Matlabin `spectrogram`-komennon avulla. Jos tunnistus on hankalaa, voit zoomata kuvaan suurennuslasityökalulla.

5.4. *(Matlab)* Erotellaan edellisen tehtävän taajuuskomponentit toisistaan suodattamalla. Suunnittele suodin käyttäen ns. Remez-menetelmää.[3] Itse menetelmä on myöhempien kurssien asiaa, mutta komentoa on helppo käyttää; komento

```
f=remez(N, [a,b,c,d], [0, 0, 1, 1]);
```

suunnittelee suotimen, jossa on $N+1$ kerrointa (N on suotimen *aste*, joka on aina yhtä pienempi kuin kertoimien määrä), ja jossa välillä $[a, b]$ olevat taajuudet poistetaan (viimeisen vektorin 2 ensimmäistä kerrointa nollia) ja välillä $[c, d]$ olevat taajuudet säilytetään (viimeisen vektorin 2 jälkimmäistä kerrointa ykkösiä). Kertoimet sijoitetaan vektoriin `f`. Suunnittele suodin, joka poistaa taajuutta 0.33 pienemmät taajuudet ja säilyttää taajuutta 0.36 suuremmat taajuudet (1 tarkoittaa Nyquistin rajataajuutta).

[3] Remez-algoritmin hyvänä puolena on se, että suotimen amplitudivasteen värähtelyhuiput ovat aina keskenään samalla korkeudella. Ikkunamenetelmällä suunnitellun suotimen huiput ovat sitä korkeammalla mitä lähempänä siirtymäkaistaa ne ovat. Remez-algoritmin haittapuolena on sen mutkikas toteutus.

Suotimen aste N voi olla vaikkapa 98 (=riittävästi). Katso amplitudivasteesta, onnistuiko suunnittelu (`help freqz`). Suodata signaali `salainen` em. suotimella ja kuuntele tulos (`help filter`).

5.5. *(Matlab)* Luo Matlabissa vektori, joka sisältää ideaalisen alipäästösuotimen impulssivasteen katkaistuna välille $-20 \leq n \leq 20$ (41 kerrointa) kun rajataajuus $f_c = 0.3$ (Nyquist $= 0.5$). Tulosta ruudulle sen amplitudivaste. (`help sinc`). Konversio Matlab-esitykseen kannattaa tehdä huolella. Lopputuloksessa päästökaistan pitäisi olla nollan desibelin kohdalla.

(a) Mikä on estokaistan ensimmäisen (vasemmanpuolimmaisen) värähtelyhuipun vaimennus (suunnilleen)?

(b) Tee em. testi kun kertoimet otetaan väliltä $-30 \leq n \leq 30$. Mitä ensimmäiselle värähtelyhuipulle tapahtuu?

(c) Mitä tapahtuu kun kertoimien määrää edelleen lisätään? Voidaanko estokaistan vaimennusta parantaa lisäämällä kertoimia?

5.6. Tavoitteena on suunnitella FIR-suodin ikkunamenetelmällä. Järjestelmän näytteenottotaajuus on 44100 Hz, ja tarkoituksena on päästää läpi 10000 Hz pienemmät taajuudet ja poistaa 12000 Hz suuremmat taajuudet.

(a) Laske näytteenottotaajuudella normalisoidut taajuudet sekä siirtymäkaistan normalisoitu leveys.

(b) Kuinka monta kerrointa (N) tarvitaan käytettäessä

i. suorakulmaista (rectangular),
ii. Hanning-,
iii. Hamming-,
iv. Blackman-

ikkunaa?

5.7. Minkä ikkunan valitset kun suotimelta vaaditaan, että

(a) päästökaistalla saa värähtelyä olla enintään 0.01 dB ja estokaistan vaimennus on vähintään 30 dB,

(b) päästökaistalla saa värähtelyä olla enintään 0.02 dB ja estokaistan vaimennus on vähintään 40 dB,

(c) päästökaistalla saa värähtelyä olla enintään 0.5 dB ja estokaistan vaimennus on vähintään 28 dB,

(d) päästökaistalla saa värähtelyä olla enintään 0.1 dB ja estokaistan vaimennus on vähintään 50 dB?

Valitse aina se ikkuna, jota käyttäen tarvitaan mahdollisimman vähän kertoimia.

5.8. Suunnittele ikkunamenetelmällä ylipäästösuodin (selvitä käsin impulssivasteen lauseke), jonka vaatimukset ovat seuraavat:

Päästökaista	[18 kHz, 22.05 kHz]
Estokaista	[0 kHz, 15 kHz]
Päästökaistan maksimivärähtely	0.1 dB
Estokaistan minimivaimennus	30 dB
Näytteenottotaajuus	44.1 kHz

5.9. *(Matlab)* Suunnittele Matlabin avulla edellisen tehtävän vaatimukset toteuttava ylipäästösuodin. Tulosta ruudulle impulssivaste (`help impz`), amplitudi- ja vaihevasteet (`help freqz`) sekä napa-nollakuvio (`help zplane`). **Ohje:** Matlabissa komento `d = fir1 (N-1, 2*fc, tyyppi, ikkuna)` antaa vektorina määrittelyn mukaisen FIR-suotimen:

- sisältää `N` kerrointa
- siirtymäkaista on keskitetty taajuuteen `fc`. `fc` voi olla myös vektori, jossa on kaksi arvoa. Ne ilmoittavat kaistanpäästö- ja estosuodinten tapauksessa tarkasteltavan kaistan alun ja lopun.
- suodintyyppi saadaan muuttujasta `tyyppi`, joka voi saada arvot (hipsut mukaanlukien):
  - `'low'`, jolloin tuloksena on alipäästösuodin
  - `'high'`, jolloin tuloksena on ylipäästösuodin
  - `'stop'`, jolloin tuloksena on kaistanestosuodin
  - `'bandpass'`, jolloin tuloksena on kaistanpäästösuodin

  Tätä ja edellistä yhdistelemällä saadaan aikaan kaikki tarkastellut suodintyypit
- `ikkuna` määrää suunnittelussa käytettävän ikkunan tyypin (pituus `N`), sallittuja tyyppejä ovat
  - `bartlett(N)` - Bartlett-ikkuna
  - `blackman(N)` - Blackman-ikkuna.
  - `boxcar(N)` - Suorakulmainen ikkuna.
  - `chebwin(N)` - Chebyshev-ikkuna.
  - `hamming(N)` - Hamming-ikkuna.
  - `hanning(N)` - Hanning-ikkuna.
  - `kaiser(N)` - Kaiser-ikkuna.
  - `triang(N)` - Kolmioikkuna.

  Myös mikä tahansa muu vektori kelpaa ikkunaksi.

5.10. *(Matlab)* Lataa signaali 'handel' muuttujaan `y` komennolla `load handel`. Signaalin näytteenottotaajuus on 8192 Hz. Suunnittele ikkunamenetelmällä suotimet, joiden aste on 50, ja joiden päästö- ja estokaistat ovat seuraavat.

(a) Päästökaista 0 Hz–1000 Hz ja estokaista 1200 Hz–4096 Hz (alipäästösuodin).

(b) Päästökaista 1800 Hz–4096 Hz ja estokaista 0 Hz–1500 Hz (ylipäästösuodin).

(c) Päästökaista 2000 Hz–3000 Hz ja estokaistat 0 Hz–1500 Hz ja 3500 Hz–4096 Hz (kaistanpäästösuodin).

(d) Päästökaistat 0 Hz–500 Hz ja 3000 Hz–4096 Hz ja estokaista 750 Hz–2500 Hz.

Tulosta suodinten amplitudivasteet ruudulle (komento `freqz`). Suodata signaali em. suotimilla ja kuuntele tulokset. Jos äänentoisto ei toimi katso tulos `spectrogram`-komennolla. Huomaa, että todellisessa tilanteessa täytyy huomioida myös vaimennusvaatimukset. Tässä tehtävässä niistä ei yksinkertaisuuden vuoksi välitetä.

5.11. Täydennä luentomonisteessa ollut ideaalisen alipäästösuotimen taajuusvasteen lasku. Osoita siis, että

$$\frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{i\omega n}\,d\omega = 2f_c\,\mathrm{sinc}(\omega_c n),$$

missä $\omega_c = 2\pi f_c$ ja $\mathrm{sinc}(x) = \sin x/x$, kun $x \neq 0$ ja $\mathrm{sinc}(x) = 1$, kun $x = 0$.

5.12. Suunnittele ikkunamenetelmällä alipäästösuodin (selvitä käsin impulssivasteen lauseke), jonka vaatimukset ovat seuraavat:

Päästökaista	[0 kHz, 12 kHz]
Estokaista	[13.5 kHz, 16 kHz]
Päästökaistan maksimivärähtely	0.1 dB
Estokaistan minimivaimennus	50 dB
Näytteenottotaajuus	32 kHz

5.13. *(Matlab)* Suunnittele Matlabin avulla kaistanestosuodin, joka toteuttaa seuraavat vaatimukset.

Päästökaistat	[0 Hz, 500 Hz] ja [3500 Hz, 4096 Hz]
Estokaista	[1000 Hz, 3000 Hz]
Päästökaistan maksimivärähtely	0.02 dB
Estokaistan minimivaimennus	40 dB
Näytteenottotaajuus	8192 Hz

Tulosta suotimen amplitudivaste ruudulle komennolla `freqz`.

5.14. (*Matlab*) Suunnittele ylipäästösuodin seuraavilla vaatimuksilla.

Päästökaista	[1240 Hz, 4096 Hz]
Estokaista	[0 kHz, 820 Hz]
Päästökaistan maksimivärähtely	0.1 dB
Estokaistan minimivaimennus	60 dB
Näytteenottotaajuus	8192 Hz

Tulosta suotimen amplitudivaste ruudulle komennolla `freqz`.

Luku 6

IIR-suodinten suunnittelu

Perinteinen menetelmä IIR-suodinten suunnittelussa on suunnitella ensin vastaava analoginen suodin ja muuntaa se sitten digitaaliseen muotoon. Muunnoksessa käytetään ns. bilineaarimuunnosta tai impulssi-invarianttitekniikkaa. Tällä kurssilla ei perehdytä suunnittelumenetelmiin, vaan ne jätetään myöhemmille lineaarisen suodatuksen kursseille (lukuunottamatta tehtävää 6.6, jossa luodaan lyhyt katsaus alipäästösuotimen suunnittelun toteutukseen). Seuraavassa esitellään IIR-suodinten neljä eri tyyppiä ja tutkitaan kuinka suunnittelu tehdään Matlabin valmiilla rutiineilla.

Matlab-ohjelmiston ja sen signaalinkäsittelyn toolboxin avulla on mahdollista suunnitella IIR-suotimia samaan tyyliin kuin FIR-suotimiakin. Matlabille annetaan siis suotimelle asetettavat vaatimukset vektorimuodossa, ja Matlab palauttaa sitä vastaavan IIR-suotimen kertoimet kahtena vektorina. Nämä vektorit sisältävät kertoimet herätteen ja vasteen viivästetyille termeille.

IIR-suotimet jaetaan neljään luokkaan niiden päästö- ja estokaistan värähtelyominaisuuksien mukaan:

- Butterworth–suotimet,
- Tyypin I Chebyshev–suotimet,
- Tyypin II Chebyshev–suotimet,
- Elliptiset suotimet.

6.1 Butterworth–suotimet

Butterworth-tyyppisille IIR-suotimille on tyypillistä, että päästökaistan alku ja estokaistan loppu (alipäästösuotimen tapauksessa) ovat molemmat mahdollisimman tasaisia (maximally flat stopband; maximally flat passband). Butterworth-tyyppisen IIR-suotimen suunnittelussa tarvitaan kahta komentoa, joista ensimmäinen laskee tarvittavien kertoimien määrän ja skaalaa argumenttina saamansa päästö– ja estokaistat vastaavaksi analogisen prototyypin argumentiksi. Käytännössä suunnittelussa ei siis tarvitse muistaa lainkaan analogisen prototyypin käyttöä; Matlab hoitaa sen käytön. Alipäästösuodinta suunniteltaessa kertoimien määrää voidaan arvioida komennolla

```
[N, Wn] = buttord(2*Wp, 2*Ws, Rp, Rs)
```

missä `N` on ulostulona saatava aste ja `Wn` on analogisen prototyypin rajataajuus. Kahden ensimmäisen parametrin kerroin tarvitaan, koska Matlab skaalaa taajuudet välille $[0, 1]$ ja tällä kurssilla ne skaalataan välille $[0, 0.5]$. Tarvittavat argumentit ovat

- `Wp` on päästökaistan rajataajuus
- `Ws` on estokaistan rajataajuus
- `Rp` on suurin sallittu värähtely päästökaistalla desibeleinä
- `Rs` on estokaistan minimivaimennus

Suunnittelu tapahtuu tämän jälkeen komennolla

```
[b,a] = butter (N, Wn);
```

Samaa komentoa käytetään myös ylipäästösuodinten suunnittelussa. Tällöin lisätään komennon viimeiseksi argumentiksi merkkijono 'high'. Esimerkiksi:

```
[b,a] = butter (N, Wn, 'high');
```

Jos halutaan kaistanpäästö- tai kaistanestosuotimia, pitää argumenttien Ws ja Wp sijasta antaa vektorit [W1, W2] ja [W3, W4], jotka ilmoittavat amplitudivasteiden reunoja vastaavat siirtymäkaistat. Lisäämällä viimeiseksi argumentiksi merkkijono 'stop', saadaan kaistanestosuodin; muutoin tuloksena on kaistanpäästösuodin.

Kaikissa tapauksissa tuloksena saadaan kaksi vektoria, jotka sisältävät siirtofunktion kertoimet: `b`$= (b_0, b_1, \ldots, b_N)$ ja `a`$= (1, a_1, \ldots, a_N)$. Näistä ensimmäisessä on osoittajan ja jälkimmäisessä nimittäjän kertoimet:

$$\frac{Y(z)}{X(z)} = H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_N z^{-N}}.$$

Kuten aiemmin on opittu, tästä saadaan varsinaisen suotimen kertoimet kertomalla yhtälön molemmat puolet termillä $X(z) \cdot \left(1 + \sum_{k=1}^{N} a_k z^{-k}\right)$

$$Y(z) \cdot \left(1 + \sum_{k=1}^{N} a_k z^{-k}\right) = X(z) \sum_{k=0}^{N} b_k z^{-k}$$

ja edelleen käänteisen z-muunnoksen avulla

$$y(n) + \sum_{k=1}^{N} a_k y(n-k) = \sum_{k=0}^{N} b_k x(n-k).$$

Tämä voidaan ilmaista muodossa

$$y(n) = \sum_{k=0}^{N} b_k x(n-k) - \sum_{k=1}^{N} a_k y(n-k).$$

Suunnittelun jälkeen tämä kaava voidaankin jo sellaisenaan toteuttaa. Myös kappaleissa 6.2–6.4 suunnittelukomento palauttaa kaksi vektoria, joilla on sama tulkinta.

Esimerkki: Suunnitellaan alipäästösuodin, jonka päästökaista on normalisoiduissa taajuuksissa välillä $[0, 0.11]$ ja estokaista välillä $[0.145, 0.5]$; päästökaistan maksimivärähtely on 0.01 dB ja estokaistan minimivaimennus on 47 dB.

```
[N, Wn] = buttord (0.22, 0.29, 0.01, 47)
```

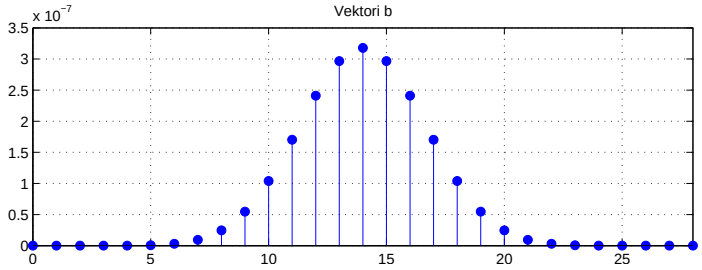
Komento palauttaa arvot:

```
N = 28
Wn = 0.2443
```

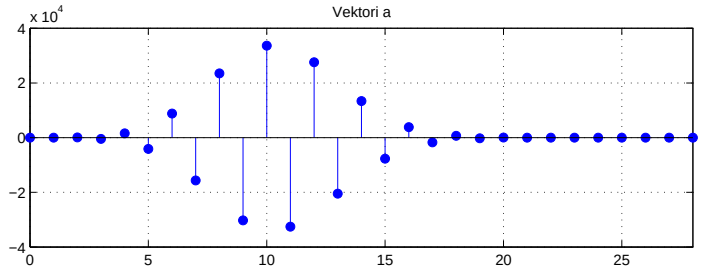
Varsinainen suunnittelu tapahtuu komennolla

```
[b,a] = butter(N,Wn);
```

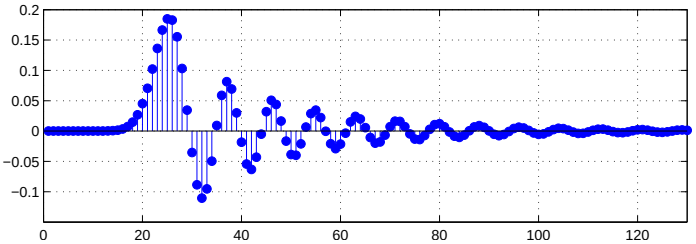
Näin saadaan kaksi vektoria; `b`, jonka kuvaaja on seuraavassa



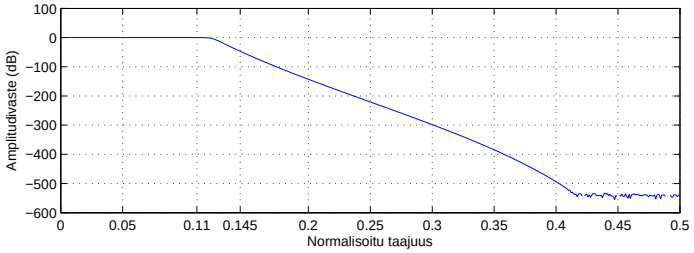
ja vektori `a`:



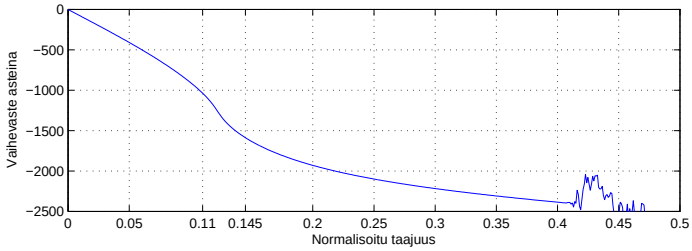
Suunnitellun suotimen impulssivasteen alku saadaan komennolla `impz(b,a)`



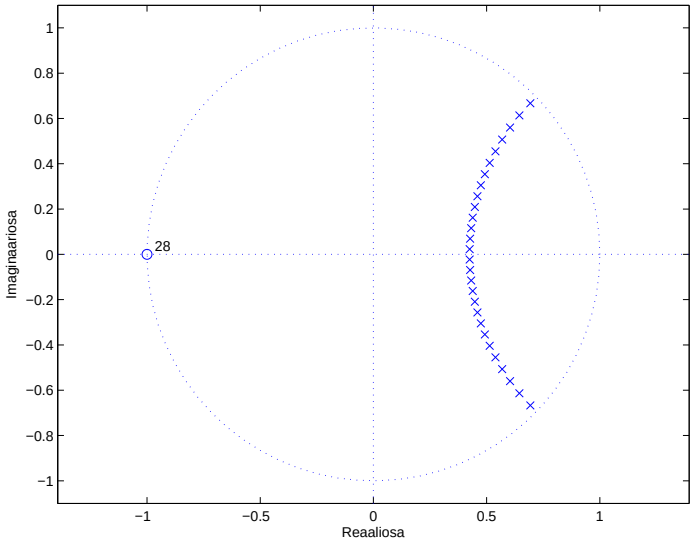
Taajuusvaste (`freqz(b,a)`) on seuraavassa kuvassa. Huomaa Butterworth-suotimelle tyypillinen huikea vaimennus estokaistan loppupäässä. Vaimennus lähestyy arvoa $-\infty$ dB lähestyttäessä normalisoitua taajuutta 0.5. Selvästikin Matlabin laskentatarkkuus loppuu, kun $|H(e^{i\omega})|$ on alle -500 dB (lineaarisella asteikolla luvut ovat tällöin luokkaa 10^{-25}).



Vaihevaste on melko lineaarinen päästökaistalla eikä näin ollen haitanne useimmissa sovelluksissa. Sen kuvaaja on alla.



Lopuksi navat ja nollat:



Kuvasta nähdään kaikkien suotimen nollien sijaitsevan pisteessä $z = -1$, joka vastaa Nyquistin taajuutta. Näin ollen suotimen amplitudivaste tällä taajuudella on lineaarisella asteikolla nolla, eli desibeleinä amplitudivaste lähestyy arvoa $-\infty$. Amplitudi- ja vaihevasteen kuvaajista nähdäänkin, että Matlabin laskentatarkkuus loppuu lähellä Nyquistin taajuutta. Samasta syystä `zplane`-komento ei osaa laskea tämän suotimen napa-nollakuviota oikein. Yllä oleva kuvio saadaan käyttämällä komentoa `butter` kolmella ulostuloarvolla:

```
[z,p,K] = butter(N,Wn);
```

Tällöin nollat ovat pystyvektorissa `z`, navat vektorissa `p` ja vahvistus nollataajuudella (ns. *gain*) muuttujassa `K`. Näin saatavat navat ja nollat ovat oikeat, koska Butterworth-suotimen suunnittelualgoritmi laskee ne ensin ja muuntaa tuloksen suotimen kertoimiksi, ks. tehtävä 6.6.

6.2 Tyypin I Chebyshev-suotimet

Ensimmäisen tyypin Chebyshev-suotimille on tyypillistä, että niiden estokaista on maksimaalisen tasainen (maximally flat), mutta päästökaista on tasavärähtelevä (equiripple). Niiden suunnittelussa on käytettävissä vastaavat komennot kuin Butterworth-suodinten tapauksessa. Suotimen aste voidaan arvioida komennolla

```
[N, Wn] = cheb1ord(2*Wp, 2*Ws, Rp, Rs)
```

missä `N` on ulostulona saatava aste ja `Wn` on analogisen prototyypin rajataajuus. Tarvittavat argumentit ovat

- `Wp` on päästökaistan lopputaajuus
- `Ws` on estokaistan alkutaajuus
- `Rp` on suurin sallittu värähtely päästökaistalla desibeleinä
- `Rs` on estokaistan minimivaimennus

Suunnittelukomento on muotoa

```
[b,a] = cheby1(N,Rp,Wn),
```

missä `N` on suotimen aste, `Rp` on suurin sallittu värähtely päästökaistalla ja `Wn` on analogisen prototyypin rajataajuus (joka arvioidaan `cheb1ord` -funktiolla).

Muut kuin alipäästösuotimet suunnitellaan samalla tavalla kuin Butterworth-suotimen tapauksessa. Tarkastellaan edellisen esimerkin suodinta vastaavaa tyypin I Chebyshev-suodinta:

```
[N, Wn] = cheb1ord(0.22, 0.29, 0.01, 47)
```

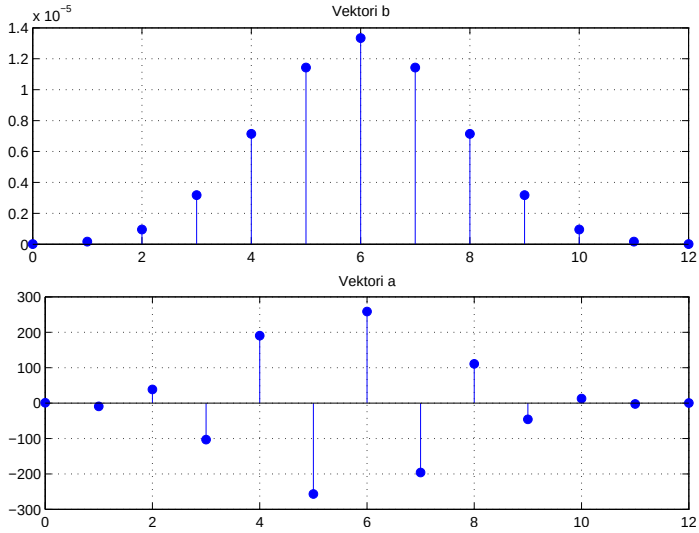
Komento palauttaa arvot:

```
N = 12
Wn = 0.2200
```

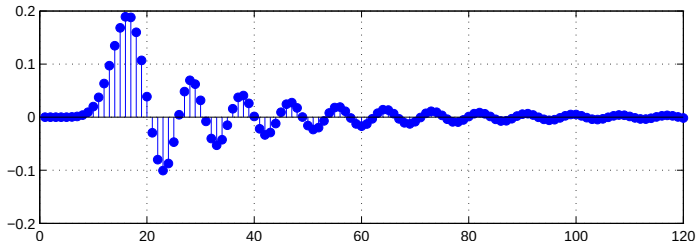
Suunnittelu tapahtuu komennolla:

```
[b,a] = cheby1 (N, 0.01, Wn);
```

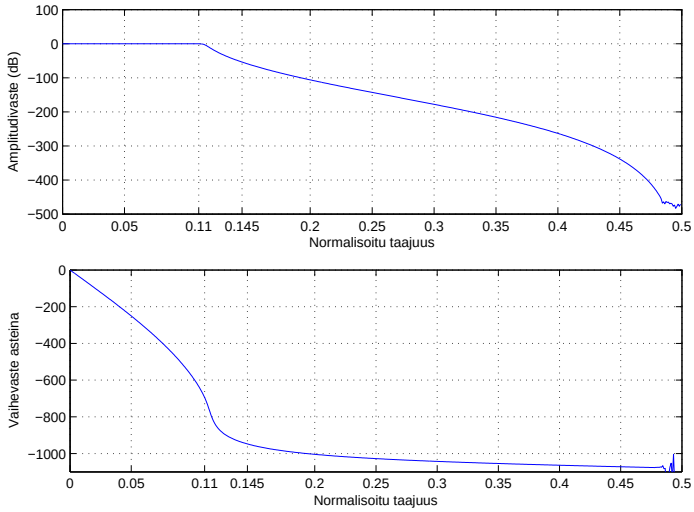
Saadun suotimen kertoimet ovat vektoreissa `b` ja `a`, joiden kuvaajat ovat alla.



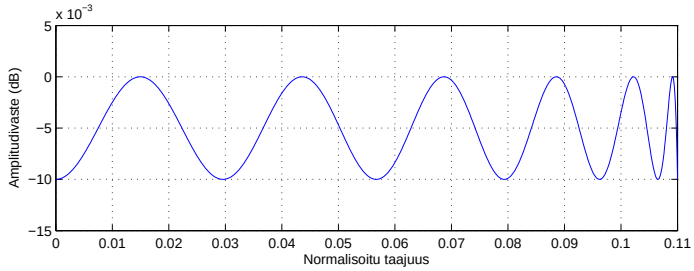
Suotimen impulssivasteen alku on alla olevassa kuvassa.



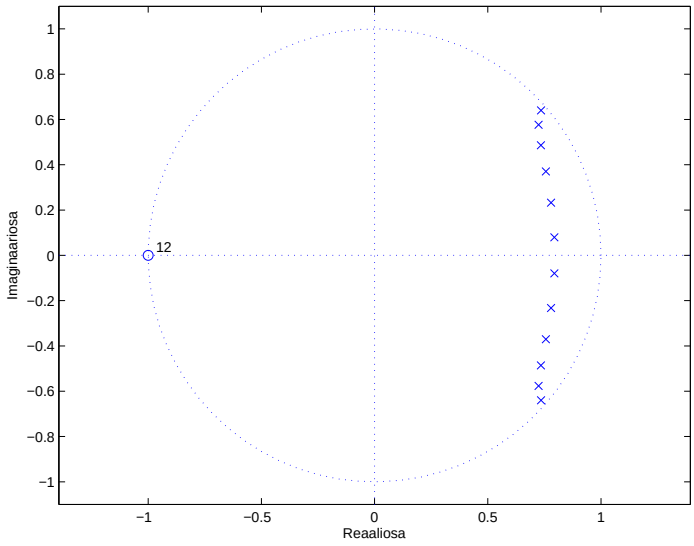
Suotimen amplitudi- ja vaihevasteet ovat seuraavassa.



Ylemmästä kuvasta nähdään, että suodin on estokaistalla samanlainen kuin Butterworth-suodinkin: molempien amplitudivaste lähestyy äärettömän desibelin vaimennusta Nyquistin rajataajuutta lähestyttäessä. Erona näillä kahdella suotimella on kuitenkin päästökaistan käyttäytyminen. Butterworth-suotimen amplitudivaste on tasainen laskeva funktio, mutta tyypin I Chebyshev-suotimen amplitudivaste värähtelee ideaaliarvon alapuolella (ks. kuva alla).



Suotimen napa-nollakuvio on alla. Tässäkin tapauksessa `zplane` laskee suotimen navat ja nollat väärin, ja ne saadaan komennosta `cheb1` samoin kuin Butterworth-suotimen tapauksessakin.



6.3 Tyypin II Chebyshev-suotimet

Toisen tyypin Chebyshev-suotimille on tyypillistä, että niiden estokaista on tasavärähtelevä (equiripple), mutta päästökaista on maksimaalisen tasainen (maximally flat). Tyypin I Chebyshev-suotimiin nähden siis päästö- ja estokaistan tyypit ovat vaihtaneet paikkaa. Tämänkin suodintyypin suunnittelussa on käytettävissä vastaavat komennot kuin edellä mainituissa tapauksissa. Suotimen aste voidaan arvioida komennolla

```
[N, Wn] = cheb2ord(2*Wp, 2*Ws, Rp, Rs)
```

missä `N` on ulostulona saatava aste ja `Wn` on analogisen prototyypin rajataajuus. Tarvittavat argumentit ovat

- `Wp` on päästökaistan lopputaajuus
- `Ws` on estokaistan alkutaajuus
- `Rp` on suurin sallittu värähtely päästökaistalla desibeleinä
- `Rs` on estokaistan minimivaimennus

Suunnittelukomento on muotoa

```
[b,a] = cheby2(N,Rs,Wn),
```

missä `N` on suotimen aste, `Rs` on suurin sallittu värähtely estokaistalla ja `Wn` on analogisen prototyypin rajataajuus (joka arvioidaan `cheb2ord` -funktiolla).

Edellisen esimerkin suodinta vastaava tyypin II Chebyshev-suodin suunnitellaan seuraavasti:

```
[N, Wn] = cheb2ord(0.22, 0.29, 0.01, 47)
```

Tulokseksi saadaan seuraavat arvot:

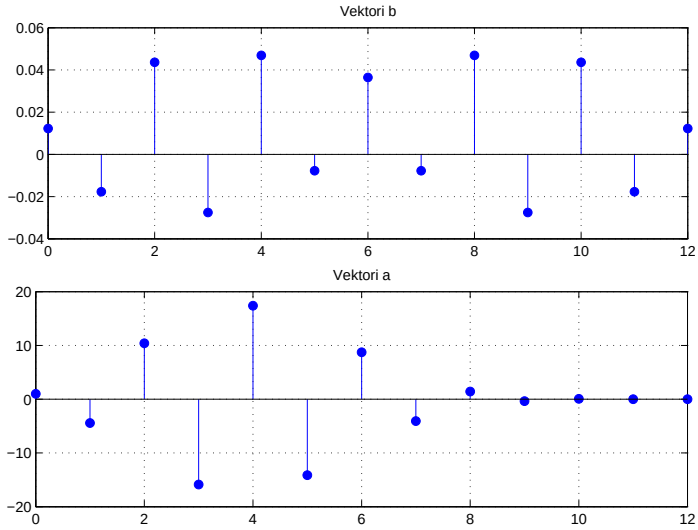
```
N = 12
```

```
Wn = 0.2795
```

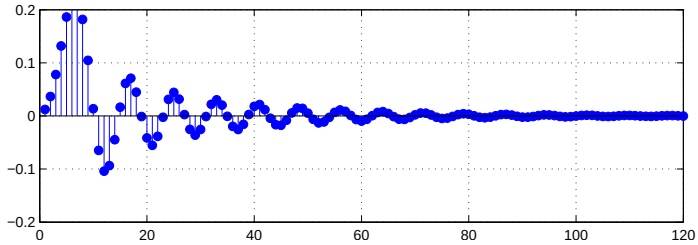
Varsinainen suunnittelu tapahtuu komennolla

```
[b,a] = cheby2 (N, 47, Wn);
```

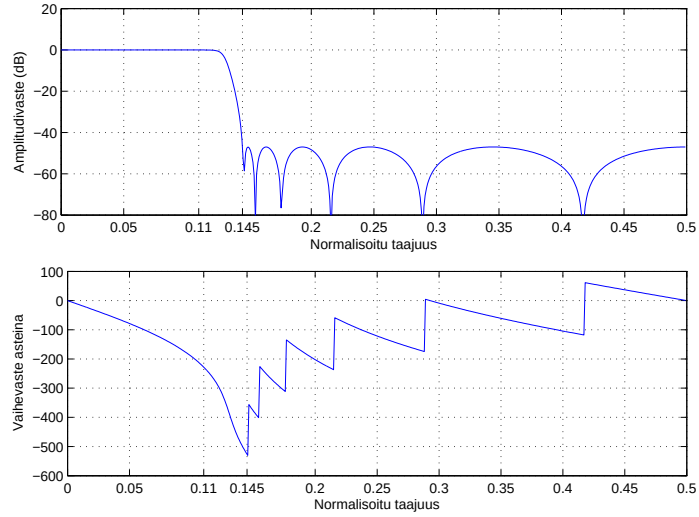
Saadun suotimen kertoimet (vektorit `b` ja `a`) on kuvattu alla.



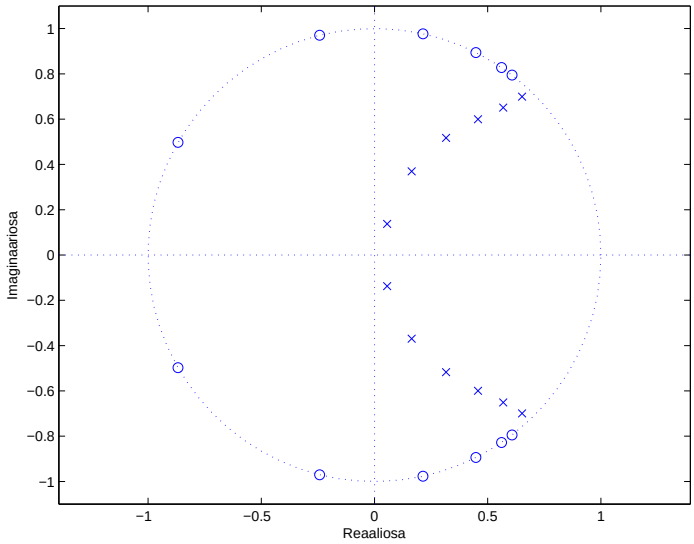
Suotimen impulssivasteen alku näyttää hyvin samanlaiselta kuin muillakin suodintyypeillä.



Amplitudi- ja vaihevasteissa näkyy kuitenkin selkeä ero Butterworth-suotimiin ja tyypin I Chebyshev-suotimiin nähden. Nyt estokaista on näet tasavärähtelevä, mikä tarkoittaa kaikkien huippujen olevan samalla korkeudella. Päästökaista puolestaan on samanlainen kuin Butterworth-suotimella, eli se laskee monotonisesti taajuuden kasvaessa.



Alla oleva napa-nollakuvio osittain selittää tasavärähtelyominaisuuden estokaistalla. Nollat ovat sijoittuneet sopivin välein estokaistalle, jolloin niiden yhteinen vaikutus tasapainottaa estokaistan värähtelyn.



6.4 Elliptiset suotimet eli Cauer-suotimet

Elliptisille suotimille on ominaista, että *sekä* päästökaistan amplitudivaste *että* estokaistan amplitudivaste ovat tasavärähteleviä. Komennot suunnittelussa ovat

```
[N, Wn] = ellipord(2*Wp, 2*Ws, Rp, Rs)
```

ja

```
[b,a] = ellip(N,Rp,Rs,Wn)
```

Kaikkien termien merkitykset on jo selvitetty edellisissä kappaleissa. Esimerkkisuotimemme suunnittelu tapahtuu siis komennoilla

```
[N,Wn] = ellipord(0.22, 0.29, 0.01, 47)
```

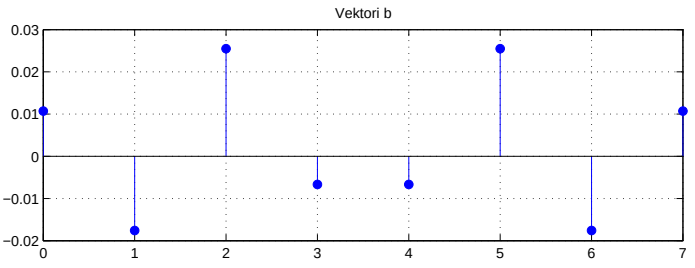
Tulokseksi saadaan:

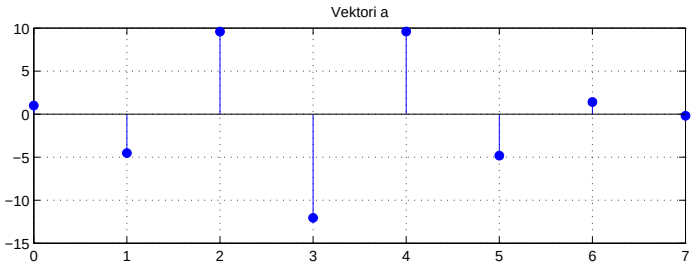
```
N = 7
Wn = 0.2200
```

Seuraavaksi suunnittelukomento:

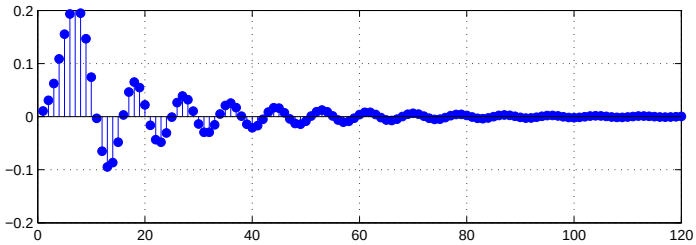
```
[b,a] = ellip (N, 0.01, 47, Wn);
```

Saadun suotimen kertoimet (vektorit `b` ja `a`) ovat alla.

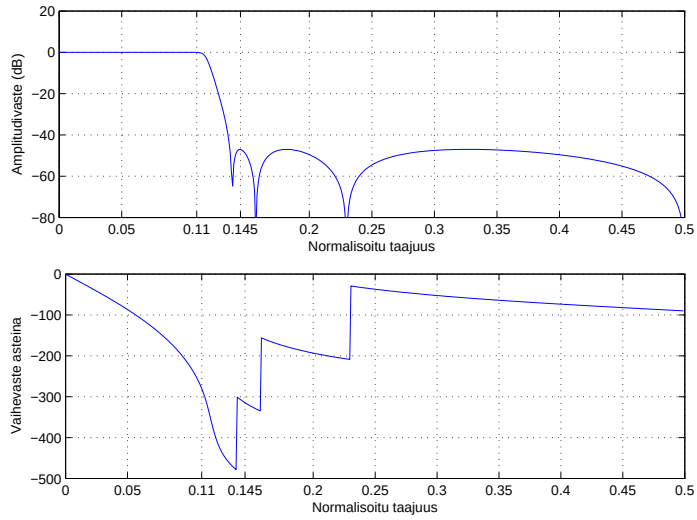




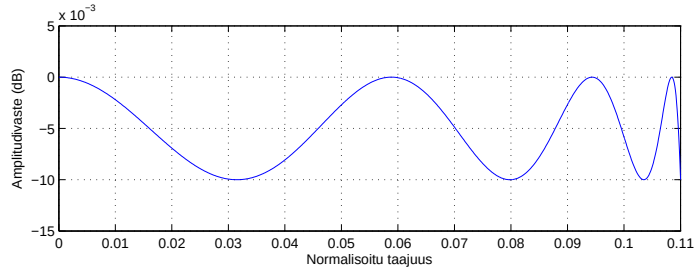
Suotimen impulssivasteen alku näyttää jälleen hyvin samanlaiselta kuin muillakin suodintyypeillä.



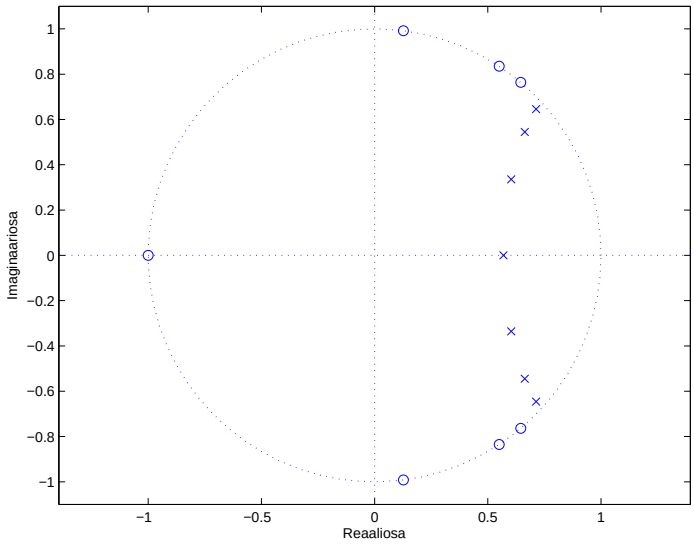
Suotimen amplitudi- ja vaihevasteet näkyvät alla.



Suotimen amplitudivaste värähtelee ideaaliarvon alapuolella (ks. kuva alla).



Suotimen napa-nollakuvio on alla.



6.5 Suodintyyppien vertailua

Alla olevassa taulukossa vertaillaan neljää IIR-suotimen tyyppiä ja niiden kertoimien määrää käsitellyssä esimerkissä. Molempien Chebyshev-suodinten kerrointen määrä on aina sama, Butterworth-suotimella on aina suurin määrä kertoimia ja elliptisillä suotimilla pienin. Vertailun vuoksi oikeanpuolimmaisessa sarakkeessa on vastaavan ikkunamenetelmällä suunnitellun FIR-suotimen kertoimien määrä, joka on selvästi IIR-suotimia suurempi.

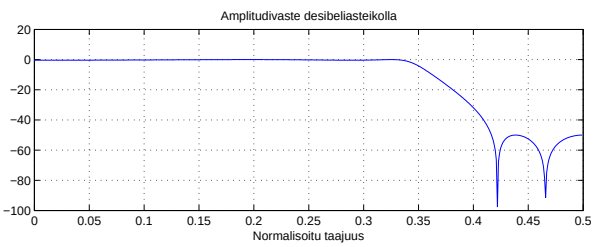
	Butterworth	*Chebyshev I*	*Chebyshev II*	*Elliptinen*	*Blackman*
Värähtely päästökaistalla	ei	kyllä	ei	kyllä	kyllä
Värähtely estokaistalla	ei	ei	kyllä	kyllä	kyllä
Kerrointen määrä	$29 + 28$	$13 + 12$	$13 + 12$	$8 + 7$	159

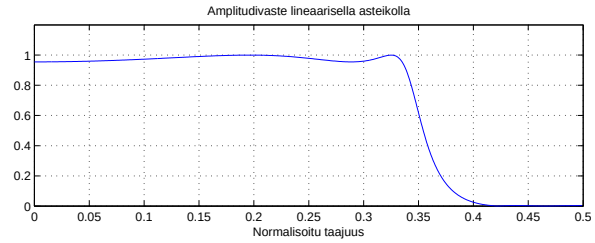
Harjoitustehtäviä

6.1. Alla olevat kuvat (a-d) esittävät eräiden IIR-suodinten amplitudivasteita. Kaikki neljä IIR-suodinten tyyppiä on mukana. Mikä suodintyyppi vastaa kutakin kuvaa?

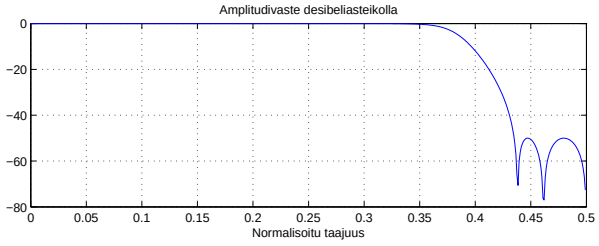
6.2. *(Matlab)* Matlab palauttaa suodinsuunnittelun tuloksena vektorit $\mathbf{b} = (0.2353, 0.8746, 1.2813, 0.8746, 0.2353)$ ja $\mathbf{a} = (1.0000, 1.1268, 1.1065, 0.2989, 0.1339)$. Alla on yksinkertaistettu esimerkki c-kielisestä ohjelmasta, joka toteuttaa mainitun IIR-suotimen. Ainoastaan varsinaisen suodatuksen toteuttavat rivit puuttuvat. Lisää sopivat rivit (kaksikin riittää). Ohjelmassa taulukko `x[]` sisältää suodatettavan signaalin ja tulossignaali tulee sijoittaa taulukkoon `y[]`. Esimerkissä signaali `x` alustetaan impulssiksi, jolloin lopputuloksena taulukossa `y` on suotimen impulssivaste. Alla oleva ohjelmakoodi löytyy verkosta:

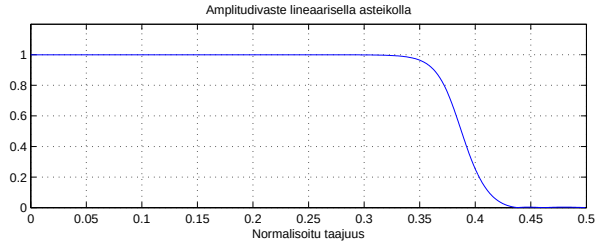
`http://www.cs.tut.fi/kurssit/SGN-1251/IIR_esim.c`



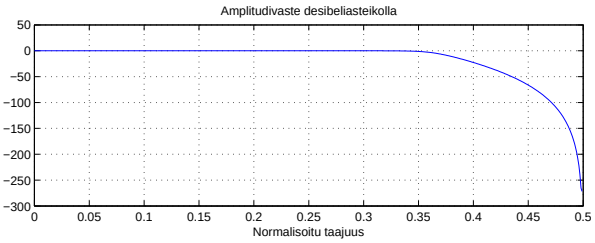


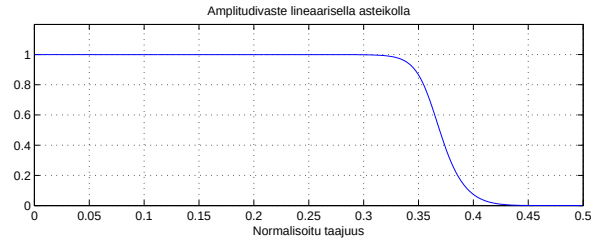
(a)



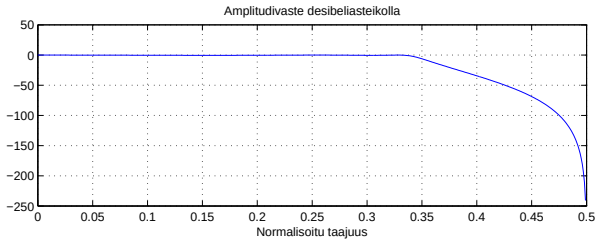


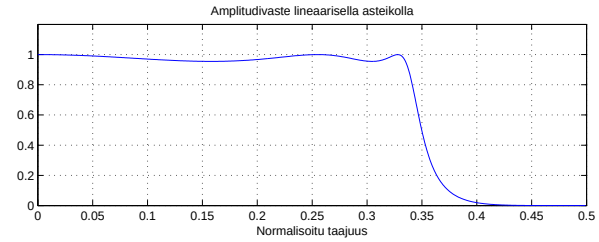
(b)





(c)



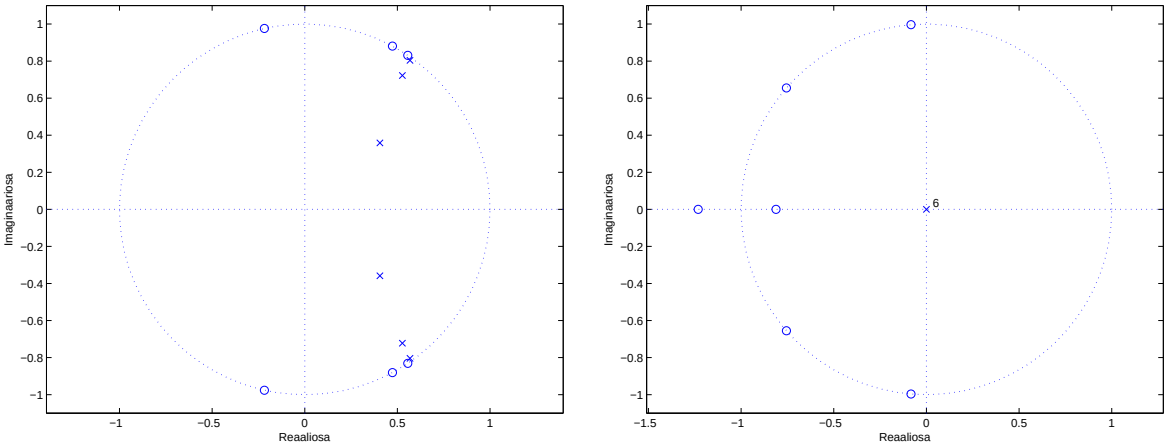


(d)

```
#include <stdio.h>

main()
{
   float x[100]; /* Suodatettava signaali */
   float y[100]; /* Suodatuksen tulos */
   float b[]={0.2353,0.8746,1.2813,0.8746,0.2353};
   float a[]={1.0000,1.1268,1.1065,0.2989,0.1339};
   short n;
   char c;

   /* Alustetaan x impulssiksi ja y nollasignaaliksi */

   for (n=0;n<100;n++)
   {
```

```
        x[n]=0.0;
        y[n]=0.0;
    }
    x[4]=1.0;

    /* Suodatetaan signaali x ja sijoitetaan tulos y:hyn */

    for (n=4;n<100;n++)
    {
        /**** Lisää tähän tarvittavat rivit ****/
    }

    /* Tulostetaan y:n arvot */

    printf("Suotimen impulssivaste:\n");

    for (n=4;n<100;n++)
    {
        printf ("%.4f ",y[n]);

        if ((n-3)%6 == 0)
        {
            printf ("\n");
        }
    }
    printf ("\n");
    printf ("Paina rivinvaihtoa.\n");

    scanf("%c", (&c));
}
```

6.3. Alla olevissa kuvissa on kaksi napa-nollakuviota. Kumpi on FIR-suotimen ja kumpi IIR-suotimen napa-nollakuvio? Millä perusteella?



6.4. *(Matlab)* Suunnittele Butterworth-tyyppinen IIR-alipäästösuodin, joka päästää läpi taajuudet välillä 0 Hz—4000 Hz ja poistaa taajuudet 7000 Hz—20000 Hz, kun näyt-

teenottotaajuus on 40000 Hz. Minimivaimennus estokaistalla on 45 dB ja maksimivärähtely päästökaistalla on 0.3 dB. Tulosta ruudulle impulssivaste (`impz`), amplitudivaste (`freqz`) ja napa-nollakuvio (`zplane`).

6.5. *(Matlab)* Laske tarvittava kertoimien määrä edellisen tehtävän vaatimuksilla, kun suodin on

(a) Chebyshev-1-suodin

(b) Chebyshev-2-suodin

(c) Elliptinen suodin

(d) Ikkunamenetelmällä suunniteltu FIR-suodin

6.6. *(Matlab)* IIR-suodinten suunnittelussa yleisin menetelmä on *bilineaarimuunnos*, joka muuntaa analogisen suotimen vastaavaksi digitaaliseksi suotimeksi. Toteutetaan tämä algoritmi alipäästösuotimen tapauksessa Matlabilla tässä ja seuraavassa tehtävässä. Valitettavasti tällä kurssilla ei voida tutustua menetelmän takana olevaan teoriaan, joten tehtävä saattaa tuntua pelkältä kaavakokoelman kirjoittamiselta. Tälläkin tasolla on kuitenkin mahdollista saada käsitys algoritmin monimutkaisuudesta sekä perusideasta. Algoritmi on pääpiirteissään seuraava.

1. Muunna IIR-suotimen suunnitteluvaatimukset vastaavan analogisen suotimen vaatimuksiksi käyttäen muunnosta[1]

$$\begin{aligned} \Omega_p &= 1 \\ \Omega_s &= c \tan(\omega_s/2), \end{aligned}$$

missä ω_p on digitaalisen suotimen päästökaistan rajataajuus, ω_s on digitaalisen suotimen estokaistan rajataajuus ja Ω_p ja Ω_s ovat analogisen suotimen vastaavat. Kerroin c saadaan kaavasta

$$c = \frac{1}{\tan(\omega_p/2)}.$$

Huomaa, että taajuudet Hertseinä pitää ensin normalisoida näytteenottotaajuudella ja kertoa luvulla 2π, jotta saadaan ω_p ja ω_s.

2. Suunnittele analoginen suodin $H(s)$, joka toteuttaa suunnitteluvaatimukset.
3. Muunna analoginen suodin digitaaliseksi bilineaarimuunnoksella, eli sijoittamalla muuttujan s paikalle lauseke

$$s = c\frac{z-1}{z+1},$$

jolloin tuloksena on siirtofunktio $H(z)$. Kerroin c määriteltiin kohdassa 1.

Tarkastellaan näitä kolmea vaihetta tässä ja seuraavassa tehtävässä. Luo tiedosto `design_lowpass.m`, jonka ensimmäinen rivi on

[1] Englanniksi tästä operaatiosta käytetään nimitystä *prewarping*.

```
function [N,D] = design_lowpass(fp, fs, Rp, Rs, Fs)
```

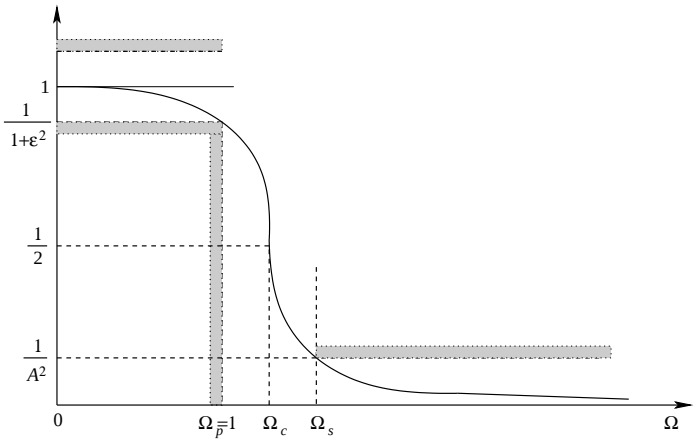
Näin määritellään siis funktio, jonka parametrit ovat päästökaistan rajataajuus (`fp`, Hertseinä), estokaistan rajataajuus (`fs`, Hertseinä), päästökaistan maksimivärähtely (`Rp`, desibeleinä, positiivinen reaaliluku), estokaistan minimivaimennus (`Rs`, desibeleinä, positiivinen reaaliluku), sekä näytteenottotaajuus (`Fs`). Näitä muuttujia käyttäen pitäisi tehdä ohjelma, joka palauttaa vektorit `N` (siirtofunktion osoittaja) ja `D` (siirtofunktion nimittäjä). Alla on ohjeita, miten ohjelma voisi edetä.

Muunna ensin rajataajuudet vastaaviksi analogisen suotimen rajataajuuksiksi.

Analogisen Butterworth-suotimen amplitudivasteen neliö on alla olevan kuvan mukainen. Kuvasta voidaan ratkaista vaimennusparametrit:

$$\begin{aligned} \epsilon &= \sqrt{10^{\delta_p/10} - 1}, \\ A &= \sqrt{10^{\delta_s/10}}, \end{aligned}$$

missä δ_p on digitaalisen suotimen päästökaistan maksimivärähtely ja δ_s on digitaalisen suotimen estokaistan minimivaimennus. Laske ϵ ja A.



6.7. *(Matlab)* (Jatkoa tehtävään 6.6) Suotimen aste voidaan arvioida kaavasta

$$M = \frac{\log_{10}\left(\frac{A^2-1}{\epsilon^2}\right)}{2\log_{10}\Omega_s}.$$

Tuloksena saadaan siis siirtofunktion aste, joka pitää pyöristää ylöspäin kokonaisluvuksi (pyöristys: `ceil`).

Vaatimukset toteuttavan Butterworth-tyyppisen analogisen suotimen siirtofunktio $H(s)$ on muotoa

$$H(s) = \frac{(-1)^M p_1 p_2 \cdots p_M}{(s-p_1)(s-p_2)\cdots(s-p_M)},$$

missä navat $p_1, p_2, \ldots, p_M$ saadaan kaavasta

$$p_k = \frac{1}{\epsilon^{1/M}} e^{\pi i[1/2+(2k-1)/(2M)]}.$$

Laske ohjelmassasi kyseiset navat ja muodosta niistä vektori.

Kun analogisen prototyypin navat ovat selvillä, saadaan vastaavat digitaalisen suotimen navat bilineaarimuunnoksella. Bilineaarimuunnoksen kaavahan oli

$$s = c\frac{z-1}{z+1}.$$

Nyt tiedetään analogisen suotimen navat (s) ja halutaan digitaalisen suotimen navat (z). Ylläolevasta kaavasta ratkaistaan siis z:

$$z = \frac{1+s/c}{1-s/c}.$$

Kukin napa p_k kuvautuu siis digitaalisen suotimen navaksi

$$p'_k = \frac{1+p_k/c}{1-p_k/c}.$$

Tee sama Matlabilla. Vektorit jaetaan termi kerrallaan operaatiolla "`./`", esimerkiksi `p3=p1./p2`.

Digitaalisen alipäästösuotimen tapauksessa kaikki nollat ovat pisteessä $z = -1$. Muodosta näistä vektori (M termiä), minkä jälkeen saat siirtofunktion komennolla `zp2tf`. Komento saa parametreinään nollat, navat sekä vahvistuksen nollataajuudella (gain). Laita tässä vaiheessa vahvistukseksi ykkönen.

Nyt sinulla on siirtofunktion kertoimet (osoittaja `N` ja nimittäjä `D`). Koska vahvistukseksi heitettiin summassa ykkönen, se on varmaankin pielessä. Vähäisellä pyörittelyllä havaitaan, että todellinen vahvistus nollataajuudella ($H(1)$) on Matlabin syntaksilla ilmaistuna `K=sum(N)/sum(D)`. Lopullinen tulos saadaan nyt jakamalla vektori `N` luvulla `K`. Rutiini palauttaa vektorit `N` ja `D` automaattisesti.

6.8. *(Matlab)* Suunnittele edellisen tehtävän rutiinilla seuraavien vaatimusten mukainen alipäästösuodin.

Päästökaista	[0 kHz, 9 kHz],
Estokaista	[12.5 kHz, 16 kHz],
Päästökaistan maksimivärähtely	0.4 dB,
Estokaistan minimivaimennus	25 dB,
Näytteenottotaajuus	32 kHz.

Tulosta ruudulle napa-nollakuviot sekä amplitudivaste.

Luku 7

Äärellisen sananpituuden vaikutukset

Tähänastinen suodinten suunnittelu on tehty oletuksella, että käytössä on ääretön sananpituus ja laskentatarkkuus. Jopa Matlabilla suunniteltaessa saatavien suodinten kertoimet on esitetty verrattain suurella tarkkuudella. Kun suotimet käytännössä toteutetaan, täytyy kuitenkin kiinnittää huomiota käytettävän lukuesityksen vaikutukseen suodinten todelliseen käyttäytymiseen. Jos suotimet toteutetaan jollain korkean tason ohjelmointikielellä (C, Pascal, Fortran, Matlab) ja liukulukuaritmetiikkaa tukevalla suorittimella, pyöristysvirheet eri vaiheissa ovat suhteellisen pieniä. Signaalinkäsittelyprosessorit eivät useimmiten laske liukuluvuilla, koska kiinteän pilkun aritmetiikan laskutoimitukset vaativat vähemmän resursseja (virtaa, tilaa, jäähdytystä, jne.) ja ne on halvempi toteuttaa. Tyypillinen signaalinkäsittelyprosessori toteuttaa laskutoimitukset kahdeksan, kahdentoista tai kuudentoista bitin tarkkuudella, joten pyöristysvirheisiin on syytä kiinnittää huomiota.

Digitaalisen IIR-suotimen tärkeimmät äärellisestä sananpituudesta johtuvat virhelähteet on lueteltu seuraavassa.

- Kvantisointivirhe, joka syntyy muunnettaessa sisään tuleva analoginen signaali digitaaliseksi signaaliksi, jonka esityksessä käytetään verraten pientä bittimäärää.

- IIR-suotimen kertoimien esitys äärellisellä bittimäärällä aiheuttaa muutoksia taajuusvasteessa ja stabiilisuusominaisuuksissa.

- Äärellisen sananpituuden seurauksena järjestelmän laskutoimituksissa saattaa tulla ylivuotoa, jolloin tulokset menettävät merkityksensä lähes täysin.

- Suodatettaessa tarvittavien kertolaskuoperaatioiden tulokset pyöristetään tai katkaistaan käytetyn sananpituuden mukaisesti.

Jos aritmetiikka on toteutettu *kahden komplementtiin* (two's complement) perustuen, niin ylivuototilanteessa esimerkiksi kahden suuren positiivisen luvun summaksi saadaan itseisarvoltaan suuri negatiivinen luku. Tämä luonnollisestikin sotkee koko suodatuksen, sillä tällainen suuri virhe säilyy ja kertautuu jatkossa IIR-suodinten rekursiivisen rakenteen vuoksi.

7.1 AD-muunnoksen kvantisointivirhe

Muunnettaessa analoginen signaali digitaaliseksi on käytettävissä ainoastaan äärellinen määrä bittejä kunkin signaalin arvon esittämiseen. Merkitään käytettävää bittimäärää (merkkibittiä lukuunottamatta) muuttujalla b ja tarkastellaan tapausta, jossa diskreetin signaalin kaikki lukuarvot on jaettu tasavälisesti välille $[-1, 1]$.

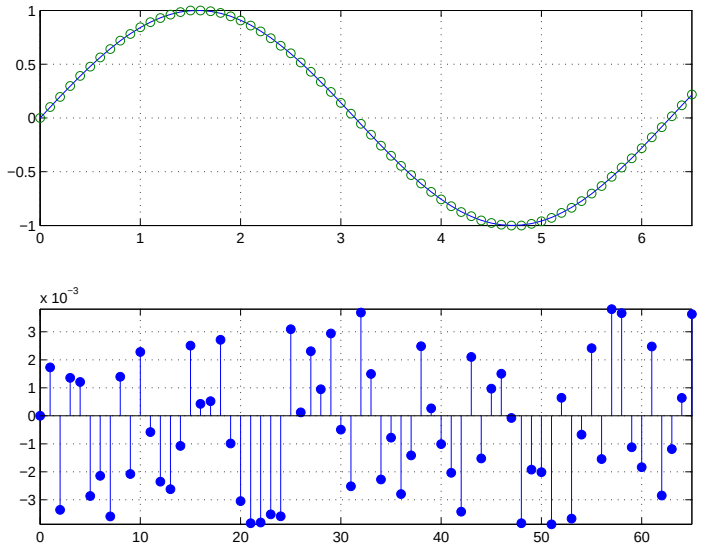
Jos käytössä on esimerkiksi seitsemän bittiä (+merkkibitti), voidaan kahden komplementtiaritmetiikalla esittää luvut $-128, -127, -126, \ldots, -1, 0, 1, \ldots, 125, 126, 127$. Nämä luvut skaalataan välille $[-1, 1]$ yksinkertaisesti jakamalla ne luvulla $2^7 = 128$. Näin saadaan kvantisointitasot $-\frac{128}{128}, -\frac{127}{128}, -\frac{126}{128}, \ldots, -\frac{1}{128}, \frac{0}{128}, \frac{1}{128}, \ldots, \frac{125}{128}, \frac{126}{128}, \frac{127}{128}$. Nämä kolme esitysmuotoa tapauksessa $b = 2$ on esitetty alla olevassa taulukossa.

binääriesitys	100	101	110	111	000	001	010	011
desimaaliesitys	-4	-3	-2	-1	0	1	2	3
kvantisointitaso	$-\frac{4}{4}$	$-\frac{3}{4}$	$-\frac{2}{4}$	$-\frac{1}{4}$	$\frac{0}{4}$	$\frac{1}{4}$	$\frac{2}{4}$	$\frac{3}{4}$

Taulukon tapauksessa kahden kvantisointivälin etäisyys on $\frac{1}{4}$ ja yleisesti ottaen se on 2^{-b} käytettäessä b:tä bittiä (+merkkibittiä). Kvantisoitaessa muodostuva virhe on enintään puolet tästä eli $2^{-b}/2$, jos käytetään pyöristystä lähimpään numeroon.[1] Tämä virhe $e(n)$ voidaan ajatella lisätyksi alkuperäiseen signaaliin $x(n)$, jolloin saadaan tulos

$$\hat{x}(n) = x(n) + e(n).$$

Alla olevissa kuvissa on esitetty sinisignaali yhtenäisellä viivalla ja tulos kvantisoitaessa seitsemään bittiin (+merkkibittiin) ympyröillä. Alemmassa kuvassa on kvantisointivirhe $e(n)$.



Tyypillisesti kvantisointivirheestä $e(n)$ tehdään seuraavat oletukset tilastollista analyysiä varten.

[1] Ylin kvantisointitaso on poikkeus tästä, koska esimerkiksi taulukon tilanteessa pitää kaikki luvut väliltä $[\frac{3}{4}, 1]$ pyöristää alaspäin lukuun $\frac{3}{4}$. Tämä poikkeus jätetään yleensä huomiotta, koska suuremmilla bittimäärillä virheen suuruusluokka on hyvin pieni ja se myöskin esiintyy hyvin harvoin.

1. Signaali $e(n)$ on stationaarinen satunnaisprosessi, eli sen tilastolliset ominaisuudet eivät muutu ajan myötä.
2. Signaalin $e(n)$ lukuarvot eivät riipu lukujonon $x(n)$ arvoista.
3. Signaalin $e(n)$ arvot ovat riippumattomia toisistaan, eli $e(n)$ on *valkoista kohinaa* (white noise).
4. Signaalin $e(n)$ arvot ovat jakautuneet tasaisesti välille $(-2^{-b}/2, 2^{-b}/2]$.

Nämä oletukset ovat voimassa, jos signaali on riittävän satunnainen. Ne eivät pidä paikkaansa kaikille signaaleille $x(n)$: jos esimerkiksi $x(n)$ on yksikköaskel, niin useimmat edellä mainituista oletuksista eivät ole voimassa. Näitä oletuksia käyttämällä saadaan kuitenkin johdettua yksinkertainen malli kvantisointivirheiden määrälle.

Jos oletetetaan järjestelmän käyttävän pyöristystä lähimpään lukuun, niin silloin signaalin $e(n)$ odotusarvo[2]

$$\mu_e = E[e(n)] = 0,$$

ja varianssi

$$\sigma_e^2 = E[(e(n) - \underbrace{\mu_e}_{=0})^2] = E[e(n)^2].$$

Odotusarvon (toisesta) määritelmästä[3] saadaan edelleen:

$$\begin{aligned} E[e(n)^2] &= \int_{-2^{-b}/2}^{2^{-b}/2} x^2 \frac{1}{2^{-b}}\,dx \\ &= 2^b \Big/_{-2^{-b}/2}^{2^{-b}/2} \frac{x^3}{3} \\ &= 2^b \left(\frac{2^{-3b}}{24} + \frac{2^{-3b}}{24} \right) \\ &= \frac{2^{-2b}}{12}. \end{aligned}$$

Kvantisoidun signaalin *signaali-kohinasuhde* (signal to noise ratio; SNR) määritellään signaalin tehon suhteena kohinan tehoon (desibeliasteikolla), ja se saadaan näiden varianssien suhteesta:

$$\mathrm{SNR} = 10 \log_{10} \frac{\sigma_x^2}{\sigma_e^2}.$$

Signaali-kohinasuhde on siis kvantisoinnissa b bittiin (+merkkibitti)

$$\begin{aligned} \mathrm{SNR} &= 10 \log_{10} \frac{\sigma_x^2}{2^{-2b}/12} \\ &= 10 \log_{10}(\sigma_x^2) + 10 \log_{10}(12) + 2b \cdot 10 \log_{10}(2) \\ &\approx 10 \log_{10}(\sigma_x^2) + 10.79 + 6.02b. \end{aligned}$$

[2]Kutakuinkin sama asia kuin lukujonon keskiarvo, s.o. $\lim_{n\to\infty} \frac{1}{2n+1} \sum_{k=-n}^{n} e(k)$.

[3]Jos tiedetään satunnaismuuttujan x jakauma $p(x)$, voidaan x:n odotusarvo laskea kaavasta $E[x] = \int_{-\infty}^{\infty} x p(x)\,dx$.

Signaali-kohinasuhde kasvaa siis noin 6 dB jokaista lisäbittiä kohden.

Yllä olevaa kaavaa sievempään muotoon ei ole mahdollista päästä tekemättä oletuksia signaalista $x(n)$ ja sen varianssista σ_x^2. Kulutuselektroniikkatuotteissa halutaan yleensä mahdollisimman hyvä signaali-kohinasuhde ja siksi teknisissä tiedoissa ilmoitettu SNR lasketaan mahdollisimman suuriamplitudiselle signaalille. Tällaiseksi sopii esimerkiksi sini- tai kosinisignaali amplitudilla 1, esimerkiksi

$$x(n) = \cos(2\pi \cdot 0.25 \cdot n).$$

Nyt signaalissa $x(n)$ toistuu jakso $\ldots, 1, 0, -1, 0, \ldots$, joten signaalin neliön odotusarvo on $\frac{1}{2}$. Samaan tulokseen päästään muillakin taajuuksilla kuin 0.25. Oletuksella $\sigma_x^2 = \frac{1}{2}$ saadaan signaali-kohinasuhteeksi

$$\text{SNR} \approx 10\log_{10}(\frac{1}{2}) + 10.79 + 6.02b = 6.02b + 7.78.$$

Esimerkiksi CD-soitin esittää näytteet 16 bitin tarkkuudella, joten tällä sinisignaalilla sen SNR$\approx 6.02 \cdot (16 - 1) + 7.78 \approx 98$ dB.

Toinen yleinen oletus signaalin $x(n)$ varianssista on, että sen amplitudi on skaalattu johonkin vakioarvoon. Käytännön tilanteissa täytyy nimittäin varautua siihen, että analoginen signaali, josta näytteitä otetaan, saa ajoittain hyvinkin suuria arvoja. Siksi sisääntulevan signaalin amplitudi on tapana kertoa jollain vakiolla A. Näin saatavan signaalin $Ax(n)$ varianssi on $A^2\sigma_x^2$, joten kvantisoinnin jälkeinen signaali-kohinasuhde on

$$\text{SNR} \approx 6.02b + 10.79 + 10\log_{10}(\sigma_x^2) + 20\log_{10}(A).$$

Nyrkkisääntönä voidaan sanoa, että valitsemalla $A = 1/(4\sigma_x)$ käytännössä eliminoidaan liian suurien arvojen saapumisen mahdollisuus. Tällöin skaalatun signaalin $Ax(n)$ varianssi on $\sigma_x^2/(16\sigma_x^2) = \frac{1}{16}$ ja signaali-kohinasuhde on desibeleissä

$$\text{SNR} \approx 6.02b + 10.79 + 10\log_{10}(\frac{1}{16}) = 6.02b - 1.25.$$

Muitakin skaalaustermejä toki käytetään ja kullekin niistä voidaan johtaa oma SNR-kaava. Mitä pienempi skaalaustermi A on, sitä pienemmäksi tulee myös SNR. Jos esimerkiksi halutaan signaali-kohinasuhteeksi tällä skaalauksella yli 80 dB, niin pitää olla $6.02b - 1.25 > 80$, eli

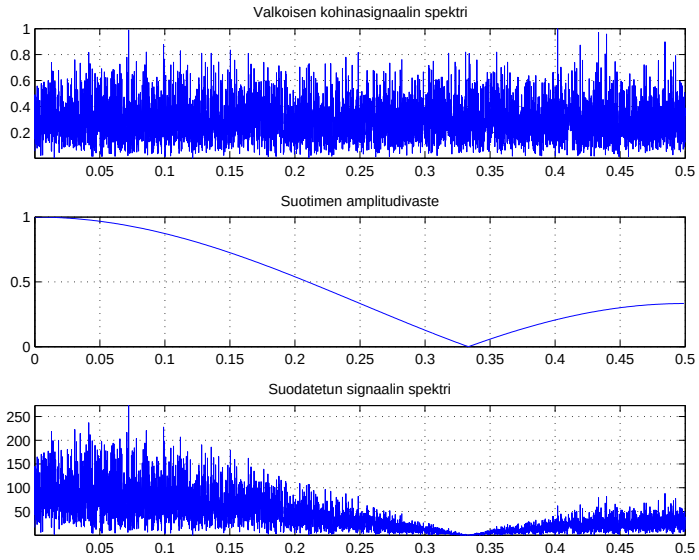
$$b > \frac{80 + 1.25}{6.02} = 13.50.$$

Näin ollen riittää valita 14-bittinen esitys (+merkki).

7.2 Suodatuksen vaikutus kvantisointikohinaan

Kulkiessaan digitaalisen suotimen läpi kvantisointikohina muuttuu kahdella tavalla: sen kokonaismäärä voi lisääntyä tai vähentyä ja toisaalta sen taajuusjakauma voi muuttua. Tässä kappaleessa keskitytään kvantisointikohinan kokonaismäärän muutokseen. Taajuusjakauman muutoshan on suoraviivainen prosessi, koska valkoista kohinaa suodatettaessa tuloksen spektri on täsmälleen saman näköinen kuin suotimen amplitudivastekin.

Kvantisointikohina käyttäytyy suodatuksessa kuten mikä tahansa muu signaalikin. Erityispiirteenä sillä on että kohina on valkoista, jolloin ennen suodatusta kaikkia taajuuksia on yhtä paljon. Esimerkiksi alla olevassa kuvassa on ylhäällä erään 10000 näytteen mittaisen kohinasignaalin taajuusjakauma. Kun se suodatetaan keskimmäisen kuvan amplitudivasteen mukaisella suotimella ($y(n) = [x(n)+x(n-1)+x(n-2)]/3$), saadaan alimman kuvan mukainen spektri. Selvästikin alimmassa kuvassa on amplitudivasteella kerrottuna ylimmän kuvan spektriarvot.



Kvantisointikohinan suodatusta tullaan myöhemmin käyttämään kappaleessa 8.5.2, jossa kvantisointikohinaa siirretään DA-muunnoksen yhteydessä korkeammille taajuuksille, jossa ihmiskorva ei sitä kuule.

Kvantisointikohinan määrän muutos suodatuksen yhteydessä on laskettavissa analyyttisesti. Tarkastellaan kvantisoidun signaalin $\hat{x}(n) = x(n) + e(n)$ suodatusta lineaarisella ja aikainvariantilla (LTI) järjestelmällä $\mathcal{F}(\cdot)$, jonka impulssivaste on $h(n)$. Koska järjestelmä on lineaarinen, niin suodatustuloksen

$$\mathcal{F}[\hat{x}(n)] = \mathcal{F}[x(n) + e(n)] = \mathcal{F}[x(n)] + \mathcal{F}[e(n)]$$

virhettä voidaan tarkastella erikseen. Suodatuksen jälkeinen pelkästään kvantisoinnista johtuva virhe on

$$f(n) = \mathcal{F}[e(n)] = \sum_{k=-\infty}^{\infty} h(k)e(n-k).$$

Tämän virheen odotusarvo on

$$\begin{aligned}
\mu_f &= E\left[\sum_{k=-\infty}^{\infty} h(k)e(n-k)\right] \\
&= \sum_{k=-\infty}^{\infty} h(k)E[e(n-k)] \\
&= \mu_e \sum_{k=-\infty}^{\infty} h(k).
\end{aligned}$$

Viimeiselle lausekkeelle saadaan myös käyttökelpoisempi muoto havaitsemalla z-muunnoksen määritelmästä, että

$$\mu_e \sum_{k=-\infty}^{\infty} h(k) = \mu_e H(1).$$

Varianssin laskenta on hieman hankalampaa, ja lausekkeen johto sivuutetaankin tässä yhteydessä. Lopputulos voidaan ilmaista seuraavasti. Kvantisointivirheestä johtuvan virheen varianssi σ_f^2 suodatuksen jälkeen voidaan esittää kolmella vaihtoehtoisella tavalla:

$$\textbf{1:} \qquad \sigma_f^2 = \sigma_e^2 \sum_{n=-\infty}^{\infty} |h(n)|^2$$

$$\textbf{2:} \qquad \sigma_f^2 = \frac{\sigma_e^2}{2\pi} \int_{-\pi}^{\pi} \left|H\left(e^{i\omega}\right)\right|^2 \, d\omega$$

$$\textbf{3:} \qquad \sigma_f^2 = \frac{\sigma_e^2}{2\pi i} \oint_C H(z)H(z^{-1})\frac{dz}{z},$$

missä σ_e^2 on sisäänmenevän (kvantisointivirheestä johtuvan) valkoisen kohinan varianssi, $h(n)$ on tarkasteltavan suotimen impulssivaste, $H(z)$ on sen siirtofunktio ja C on funktion $H(z)H(z^{-1})/z$ suppenemisalueella sijaitseva suljettu käyrä.

Erityisesti FIR-suotimen tapauksessa ensimmäinen tapa kolmesta on helpoin. Seuraavassa on kaksi esimerkkiä kvantisointivirheen käyttäytymisestä suodatettaessa IIR- ja FIR-suotimella.

Esimerkki: Oletetaan, että analoginen signaali muunnetaan digitaaliseksi (1+7)-bittisellä A/D-muuntimella, ja että näin saatu digitaalinen signaali suodatetaan IIR-suotimella, jonka siirtofunktio on

$$H(z) = \frac{1}{z + 0.5}.$$

Suodattimen toteuttava differenssiyhtälö on siis

$$y(n) = -0.5y(n-1) + x(n-1).$$

Mikä on ulostulossa oleva pelkästään kvantisoinnista johtuva virhe, kun äärellisestä laskentatarkkuudesta ja kertoimien äärellisestä esityksestä johtuvat virheet jätetään huomiotta?

Muunnosvaiheessa on käytössä 8 bittiä, joista yksi käytetään merkkibitiksi. Tässä tapauksessa siis $b = 7$. Muunnosvaiheen virheen varianssi saadaan edellä olleesta kaavasta,

$$\sigma_e^2 = \frac{2^{-2b}}{12} = \frac{2^{-14}}{12} = \frac{2^{-16}}{3} \approx 5.09 \cdot 10^{-6}.$$

Tätä vastaava keskiarvohan on 0, kun oletetaan, että analogisen signaalin arvot pyöristetään aina lähimpään lukuun suoran katkaisun asemesta. Ulostulon kvantisointivirheestä johtuvan kohinan varianssi voidaan johtaa ratkaisemalla järjestelmän impulssivaste:

$$h(n) = \left(-\frac{1}{2}\right)^{n-1} u(n-1)$$

ja sijoittamalla se edellä olleeseen kaavaan

$$\begin{aligned}\sigma_f^2 &= \sigma_e^2 \sum_{n=-\infty}^{\infty} |h(n)|^2 \\ &= \sigma_e^2 \sum_{n=1}^{\infty} \left(\frac{1}{4}\right)^{n-1} \\ &= \sigma_e^2 \frac{1}{1-\frac{1}{4}} \\ &= \frac{4}{3}\sigma_e^2.\end{aligned}$$

Kohinan varianssi kasvaa siis 4/3-kertaiseksi eli on kaikkiaan

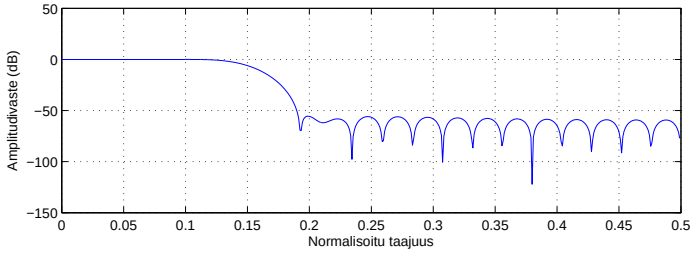
$$\sigma_f^2 = \frac{4}{3} \cdot \frac{2^{-16}}{3} \approx 6.78 \cdot 10^{-6}.$$

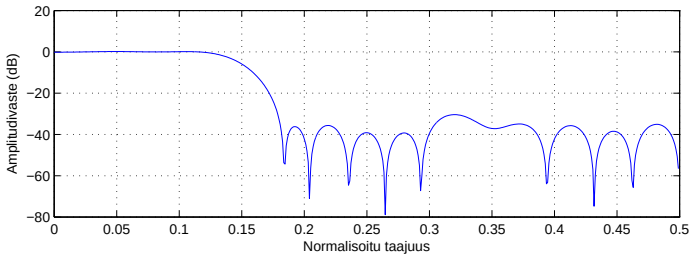
FIR-suodinten tapauksessa tilanne on yksinkertaisempi kuin IIR-suotimilla. Esimerkiksi FIR-suotimen, jonka impulssivasteen kertoimet ovat $1/3$, kun $n = 0, 1, 2$ ja 0 muulloin, ulostulossa olevan kvantisoinnista johtuvan kohinan varianssi on

$$\begin{aligned}\sigma_f^2 &= \sigma_e^2 \sum_{n=-\infty}^{\infty} |h(n)|^2 \\ &= \sigma_e^2 \left(\left(\frac{1}{3}\right)^2 + \left(\frac{1}{3}\right)^2 + \left(\frac{1}{3}\right)^2 \right) \\ &= \frac{3}{9}\sigma_e^2 = \frac{1}{3}\sigma_e^2.\end{aligned}$$

7.3 Kertoimien pyöristämisen vaikutus

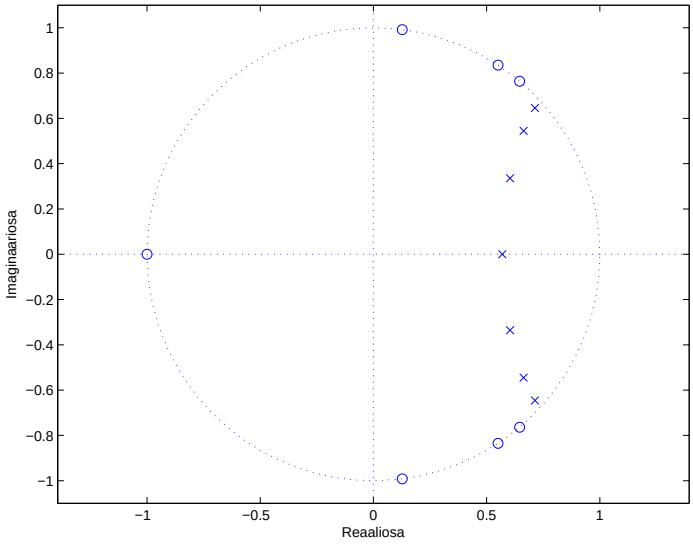
Suodinten suunnittelumenetelmien tavoitteena on suunnitella paras mahdollinen suodin tietyille kriteereille. Kun suodin sitten toteutetaan pienemmällä bittimäärällä, helpointa on pyöristää kertoimet lähimpään kvantisointitasoon. Kertoimia pyöristettäessä suodin muuttuu optimaalisesta aina huonompaan päin, jolloin suodin ei välttämättä toteutakaan annettuja vaatimuksia. Alla olevassa kuvassa on erään FIR-suotimen amplitudivaste ennen ja jälkeen kertoimien pyöristystä (1+7):n bitin tarkkuuteen.



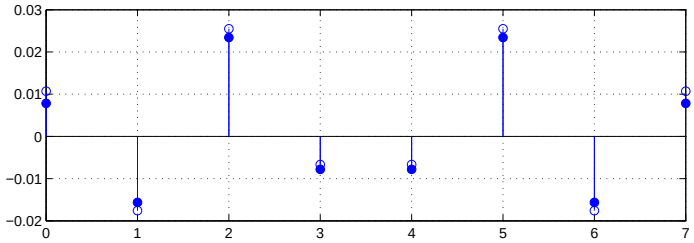


Amplitudivaste heikkenee selvästi ja suotimen toiminta ei välttämättä ole enää määrittelyjen mukaista.

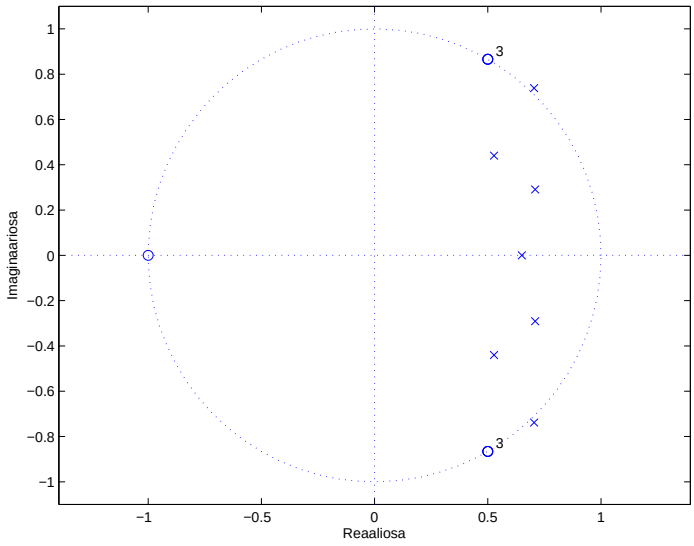
IIR-suotimen tapauksessa kvantisointi voi tuottaa vielä suuremman muutoksen suotimen toiminnassa: pahimmassa tapauksessa suotimesta tulee epästabiili. Alla olevassa kuvassa on edellisen kappaleen elliptisen IIR-suotimen napa-nollakuvio.



Kun suotimen kertoimet kvantisoidaan (1+7):n bitin tarkkuuteen, kertoimien muutos ei ole erityisen suuri. Alla olevassa kuvassa on kuvattu siirtofunktion osoittajan kertoimien muutos kvantisoitaessa. Alkuperäiset arvot on esitetty avoimilla ympyröillä ja kvantisoidut mustilla ympyröillä.



Suotimen napoihin ja nolliin tämä vaikuttaa kuitenkin paljon enemmän. Alla olevasta kuvasta nähdään, että kvantisoidusta suotimesta tuli epästabiili.



Useimmiten on olemassa parempiakin (1+7)-bittisiä suotimia kuin se, joka saadaan pyöristämällä kukin kerroin suoraan lähimpään (1+7)-bittiseen lukuun. Kertoimien muutoksen vaikutus yhteys ja nolliin ja suotimen amplitudivasteeseen on melko monimutkainen, joten suoraviivainen pyöristäminen ei välttämättä tuota parasta mahdollista amplitudivastetta. Kertoimien optimaaliseen sijoitteluun eri kvantisointitasoille perehdytään myöhemmillä signaalinkäsittelyn kursseilla.

Harjoitustehtäviä

7.1. Laske A/D-muunnoksessa syntyvän kvantisointikohinan varianssi kun käytössä on

(a) merkkibitti+3 bittiä,

(b) merkkibitti+7 bittiä,

(c) merkkibitti+11 bittiä,

(d) merkkibitti+15 bittiä.

7.2. Oletetaan, että analoginen näytteistettävä signaali skaalataan vakiolla $A = 1/(4\sigma_x)$. Mikä on digitaalisen signaalin signaali-kohinasuhde (SNR) kun käytössä on

(a) merkkibitti+3 bittiä,

(b) merkkibitti+7 bittiä,

(c) merkkibitti+11 bittiä,

(d) merkkibitti+15 bittiä?

7.3. Laske signaali-kohinasuhteen kaava, jos skaalauksessa käytetäänkin vakiota

$$A = \frac{1}{2\sigma_x}.$$

7.4. Oletetaan, että analoginen signaali on muotoa

$$x(t) = \sin(\omega t + \phi).$$

Tälle signaalille $\sigma_x^2 = 1/2$. Mikä on nyt signaali-kohinasuhteen kaava näytteenoton jälkeen, kun oletetaan, ettei analogista signaalia skaalata lainkaan? Mikä on SNR, kun käytetään 15 bittiä merkkibitin lisäksi?

7.5. *(Matlab)* Tutkitaan tässä tehtävässä kvantisointikohinaa Matlabilla.

(a) Avaa ensin testisignaali komennolla `load gong.mat`, joka lataa sen muuttujaan `y`. Kvantisoi testisignaali seitsemään bittiin (+merkkibittiin) komennolla `z=quant(y,1/2^7);`. Tulosta virhesignaali `y-z` ruudulle ja laske sen varianssi komennolla `var`). Vertaa tätä teoreettiseen tulokseen (tehtävä 7.1). Toista sama (1+3):lle, (1+11):lle ja (1+15):lle bitille.

(b) Toista testi testisignaalille `seiska.wav`, joka löytyy kurssin kotisivulta. Wav-tiedostot voi ladata Matlabiin komennolla `y = wavread('seiska.wav');`. Mistä erot kvantisointivirheen varianssissa eri testisignaaleille johtuvat?

7.6. *(Matlab)* Suunnittele Matlabilla elliptinen IIR-suodin, jonka estokaista on näytteenottotaajuudella normalisoituna $[0, 0.1]$ ja päästökaista $[0.15, 0.5]$. Estokaistan minimivaimennus on 50 dB ja päästökaistan maksimivärähtely 0.1 dB. Kvantisoi kertoimet (vektorit `b` ja `a`) (1+11):n bittiin ja tulosta ruudulle näin saadun suotimen taajuusvaste kvantisoimattoman kanssa samaan kuvaan. Kuinka paljon estokaistan vaimennus huononi?

7.7. Tarkastellaan ensimmäisen asteen IIR-suodinta, jonka siirtofunktio on

$$H(z) = \frac{1}{1 - 0.8z^{-1}}.$$

Oletetaan että analoginen signaali muunnetaan digitaaliseksi 8-bittisellä A/D-muuntimella (merkki+7 bittiä). Mikä on suotimen ulostulossa oleva pelkästään kvantisoinnista johtuva virhe, kun äärellisestä laskentatarkkuudesta ja kertoimien äärellisestä esityksestä johtuvat virheet jätetään huomiotta?

7.8. *(Matlab)* Luo normaalijakautunut satunnainen testisignaali komennolla `randn`. Pituus voi olla joitakin tuhansia näytteitä. Kvantisoi signaali kahdeksaan bittiin (merkki + 7 bittiä), ja laske empiirisesti näin saadun kvantisointikohinan varianssi. Suodata kvantisoitu sekä kvantisoimaton signaali tekstissä olevan esimerkin suotimella $y(n) = -0.5y(n-1) + x(n-1)$. Suodatuksen jälkeinen kvantisointikohina on nyt tulossignaalien erotus.

(a) Laske suodatuksen jälkeisen kvantisointikohinan varianssi ja vertaa tulosta esimerkissä laskettuun teoreettiseen tulokseen.

(b) Teoriassa kohinan varianssi kasvoi suodatettaessa $\frac{4}{3}$-kertaiseksi. Entä käytännössä?

7.9. Tarkastellaan ensimmäisen asteen IIR-suodinta, jonka siirtofunktio on

$$H(z) = \frac{1}{1 - az^{-1}}.$$

Tässä lausekkeessa a on jokin reaalivakio, jonka itseisarvo $|a| < 1$. Johda (vakiosta a riippuva) lauseke suotimen ulostulossa olevalle pelkästään kvantisoinnista johtuva virheelle, kun äärellisestä laskentatarkkuudesta ja kertoimien äärellisestä esityksestä johtuvat virheet jätetään huomiotta ja A/D-muunnin käyttää $(1 + 7)$ bitin esitystä? Mitä virheelle tapahtuu kun $|a|$ lähestyy ykköstä?

7.10. *(Matlab)* Käytettäessä IIR-suotimia äärellisellä laskentatarkkuudella, voi syntyä ilmiö nimeltä *rajasykli*. Tällöin järjestelmän vaste ei herätteen loputtua vaimene vaan jää ikuisesti värähtelemään jollain taajuudella nollan ympärille.

Lataa testiskriptin pohja osoitteesta

`http://www.cs.tut.fi/kurssit/SGN-1251/rajasykli.m`

Kirjoita kommentoidun for-silmukan tilalle koodi, jossa muuttujaan `y(n)` sijoitetaan suodatuksen tulos kvantisoinnin jälkeen. Kvantisointi tehdään viiteen bittiin (+merkkibitti) `quant`-funktiolla. Toteutettavan suotimen siirtofunktio on

$$H(z) = \frac{1}{1 + 0.5z^{-1}}.$$

Aja skripti, jolloin suotimen ulostulo tulostuu ruudulle. Vaikka heräte on nollassa heti kahden ensimmäisen näytteen jälkeen, tulos jää värähtelemään ikuisesti (tämän suotimen tapauksessa Nyquistin taajuudella).

Luku 8

Näytteenottotaajuuden muuntelu

Näytteenottotaajuuden muuntelusta A/D-muunnoksen jälkeen käytetään englannin kielessä nimitystä *multirate DSP*, mutta sille ei ole olemassa lyhyttä suomenkielistä nimeä. Yksinkertaisin menetelmä näytteenottotaajuuden muuntamiseksi on tietysti muuntaa signaali ensin analogiseksi ja tämän jälkeen muuntaa se takaisin digitaaliseksi uudella näytteenottotaajuudella. Tämä menettely johtaa kuitenkin ylimääräisiin kvantisointi- eli pyöristysvirheisiin, joita on syytä välttää. Ongelma on mahdollista ratkaista pelkästään digitaalisen signaalinkäsittelyn keinoin. Tällöin saadaan taatusti optimaalinen tulos, jossa ei ole mukana ylimääräisiä kvantisointikohinoita.

Ongelma on siis seuraava: on olemassa signaali, joka on muodostettu analogisesta signaalista näytteenottotaajuudella f. Tästä signaalista halutaan selvittää se digitaalinen signaali, joka on mahdollisimman lähellä sitä signaalia, joka olisi saatu näytteenottotaajuudella $\hat{f}$. Kouluesimerkki kyseisestä ongelmasta on muunnos CD-formaatista DAT-formaattiin. CD-levyillä data on nimittäin esitetty näytteenottotaajuudella 44.1 kHz. Sen sijaan DAT-nauhurin formaatti perustuu näytteenottotaajuuteen 48 kHz. Usein muunnos toki hoidetaan käytännössä muuntamalla signaali ensin analogiseksi ja edelleen digitaaliseksi taajudella 48 kHz. Parempaan tulokseen kuitenkin on mahdollista päästä multirate-signaalinkäsittelyn menetelmin.

Taajuuden muuntaminen on tarpeellista myös, kun signaalista on syystä tai toisesta otettu näytteitä liian korkealla taajuudella. Tällöin siis signaalissa ei ole läheskään niin suuria taajuuksia kuin näytteenottotaajuus mahdollistaa. Kun kuitenkin näytteenottotaajuus on suuri, tulee suunniteltujen suodintenkin aste tarpeettoman korkeaksi. Pienempi aste saavutetaan, jos taajuudet muunnetaan ensin pienemmäksi ja suodatetaan vasta sitten. Tämän jälkeen signaali voidaan tarvittaessa muuntaa jälleen suurempaan taajuuteen. Muuntaminen käytännössä tapahtuu kahden perusoperaation avulla:

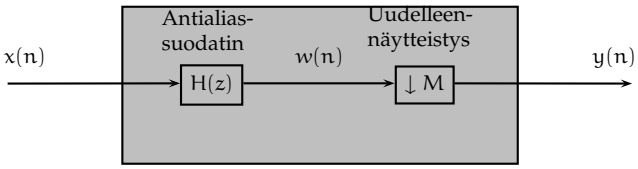
- *Interpolointi* kasvattaa signaalin näytteenottotaajuutta lisäämällä ylimääräisiä arvoja,
- *Desimointi* pienentää signaalin näytteenottotaajuutta poistamalla osan alkuperäisen signaalin arvoista.

Desimointi ja interpolointi muuntavat näytteenottotaajuutta jollain kokonaislukukertoimella (esimerkiksi 1 kHz $\mapsto$ 2kHz tai 1 kHz $\mapsto$ 0.5kHz, jne). Näitä operaatioita yhdistelemällä saadaan aikaiseksi kaikkia rationaalikertoimia vastaavat taajuusmuunnokset. Tarkastellaan lähemmin muunnosta CD-formaatista DAT-formaattiin. Käytössä on 44.1 kHz

ja tavoite olisi 48 kHz. Muunnoskerroin on $48/44.1 = 480/441 = 160/147$. Ensin taajuus korotetaan siis arvoon $44.1 \cdot 160 = 7056$ kHz interpoloimalla kertoimella 160. Tämän jälkeen signaali muunnetaan takaisin taajuuteen 48 kHz desimoimalla kertoimella 147. Merkittävää on, että muunnos tapahtuu juuri tässä järjestyksessä. Jos ensin olisi desimoitu, niin tuloksena olevassa signaalissa näytteenottotaajuus olisi ollut liian pieni, ja suurin osa signaalin informaatiosta olisi hävinnyt. Nyt tällaista informaatiohävikkiä ei tapahdu.

8.1 Desimointi eli näytteenottotaajuuden alentaminen

Oheinen kaavio esittää signaalin $x(n)$ desimointiin liittyviä vaiheita. Ennen varsinaista näytteenottotaajuuden pienentämistä signaali täytyy suodattaa alipäästösuotimella laskostumisen estämiseksi. Näin saadun signaalin näytteenottotaajuutta pienennetään jättämällä ainoastaan osa alkuperäisen signaalin arvoista jäljelle. Desimointioperaatiota merkitään alaspäin osoittavalla nuolella ja taajuuden muunnoskertoimella. Esimerkiksi näytteenottotaajuuden pudottamista kolmannekseen merkitsevä symboli on $\boxed{\downarrow 3}$. Kuvion tapauksessa alkuperäinen näytteenottotaajuus F_s pudotetaan desimoinnissa arvoon F_s/M. Tämä tehdään poimimalla joka M:s alkio desimoituun signaaliin.



Koska desimoitaessa näytteenottotaajuus pienenee, on laskostumisen vaara ilmeinen. Laskostuminenhan estetään ainostaan poistamalla signaalista taajuudet, jotka ovat suurempia kuin puolet näytteenottotaajuudesta. Tämän suorittava suodin (digital anti-aliasing filter) on siis sellainen alipäästösuodin, joka poistaa kaikki arvoa $F_s/2M$ suuremmat taajuudet. Normalisoiduissa taajuuksissa ilmaistuna alipäästösuotimen suunnitteluvaatimukset ovat seuraavat.

- Suotimen päästökaista on $[0, \frac{1}{2M} - \Delta f]$.
- Suotimen estokaista on $[\frac{1}{2M}, \frac{1}{2}]$.

Siirtymäkaistan leveys, Δf, määräytyy sovelluksen mukaan. Mitä kapeammaksi siirtymäkaista halutaan, sitä enemmän kertoimia suotimessa tulee olla. Myös vaimennusvaatimukset riippuvat sovelluksesta.

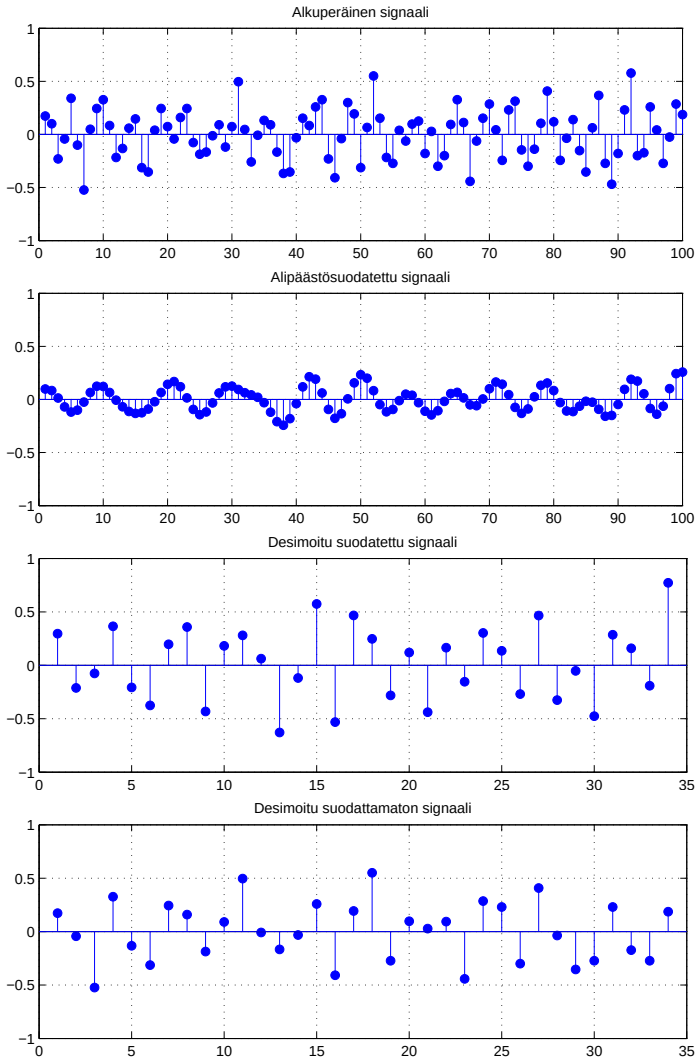
Kaavoina signaalin $x(n)$ desimointiprosessi kertoimella M signaaliksi $y(n)$ voidaan ilmaista seuraavasti:

$$\begin{aligned} w(n) &= \sum_{k=-\infty}^{\infty} h(k)x(n-k), \\ y(m) &= w(mM) = \sum_{k=-\infty}^{\infty} h(k)x(mM-k). \end{aligned}$$

Alla olevat kuvat esittävät desimoinnin vaikutusta aikatasossa. Alkuperäinen signaali $x(n)$ suodatetaan ensin alipäästösuotimella, jolloin saadaan signaali $w(n)$, joka on valmis

desimoitavaksi. Desimoitaessa kertoimella 3 otetaan uuteen signaaliin mukaan ainoastaan joka kolmas arvo. Vertailun vuoksi mukana on myös signaali, joka on saatu desimoimalla ilman alipäästösuodatusta.

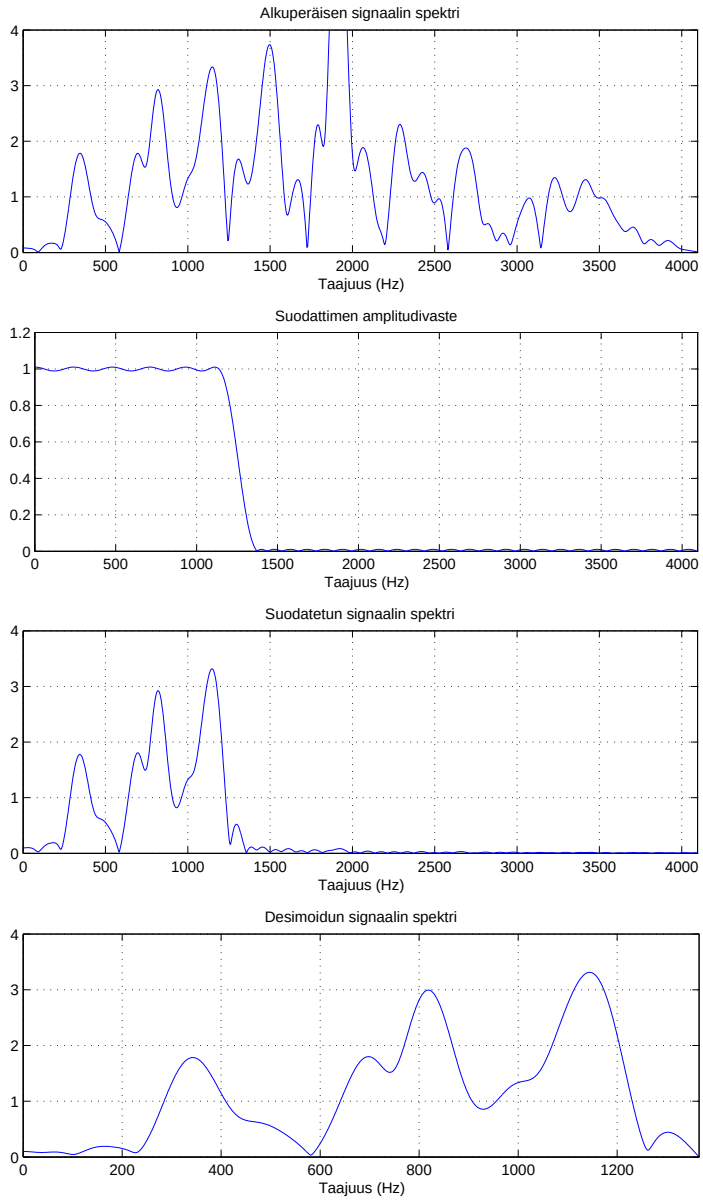
Toiseksi alimmassa kuvassa näytteet on kerrottu desimointikertoimella $M = 3$, jotta näytteiden suuruusluokka olisi sama kuin alkuperäisellä signaalilla. Ilman tätä kertolaskua näytteiden lukuarvot olisivat selvästi alkuperäisiä pienemmät, koska alipäästösuodatuksessa signaalin energia putoaa noin kolmannekseen. Yleensä tämä kertolasku jätetään pois kaavoista, koska se on helpointa toteuttaa suunnittelemalla suodin, jonka amplitudivaste päästökaistalla on M. Tällainen suodin saadaan normaalista alipäästösuotimesta yksinkertaisesti kertomalla sen jokainen kerroin luvulla M.

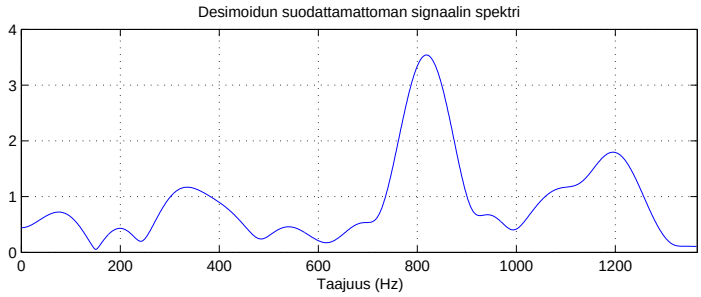


Desimointiprosessi selvinnee helpommin taajuustason kuvaajista. Ensimmäinen kuvaaja esittää alkuperäisen signaalin $x(n)$ spektriä (diskreetin Fourier-muunnoksen itseisarvoa sopivasti ikkunoituna ja interpoloituna) $|X(n)|$. Esimerkin tapauksessa näytteenottotaajuus on 8192 Hz. Tällöin siis suurin signaalin sisältämä taajuus on 4096 Hz. Kun tätä signaalia halutaan desimoida kertoimella 3, täytyy ensin poistaa $1365\frac{1}{3}$ Hz suuremmat taajuudet. Tätä varten suunnitellaan alipäästösuodin, jonka estokaista on väli $[\frac{1}{6}, \frac{1}{2}]$ (näytteenottotaajuuden suhteen normalisoituina taajuuksina) ja päästökaista nollasta johonkin

lukua $1/6$ pienempään lukuun, riippuen kuinka paljon kertoimia on varaa käyttää. Oheisessa esimerkissä päästökaistan rajataajuudeksi valittiin 0.14, päästökaistan maksimivärähtelyksi 0.09dB (0.01 lineaarisella asteikolla) ja estokaistan minimivaimennukseksi 40dB (0.01 lineaarisella asteikolla). Tämän suotimen amplitudivaste on kuvattu oheisessa kuvassa. Tällä suotimella suodatettaessa saadaan signaali $w(n)$, jonka spektri on suotimen amplitudivasteen alla olevassa kuvassa. Kun signaalista $w(n)$ jätetään jäljelle vain joka kolmas arvo (ja kerrotaan näytteet luvulla 3), tuloksena on signaali $y(n)$, jonka spektri on seuraavassa kuvassa. Nyt siis uusi näytteenottotaajuus on $8192/3$Hz.

Viimeinen kuvaaja esittää suodattamattoman desimoidun signaalin spektriä, jossa laskostumisilmiö on selvästi havaittavissa. Siinä oleva spektri poikkeaa alkuperäisen signaalin vastaavasta kaistasta, vaikka ne halutaan mahdollisimman lähelle toisiaan.





8.2 Näytteenottotaajuuden pienentäminen useassa vaiheessa

Desimaattoreiden toteutusta voidaan nopeuttaa pudottamalla näytteenottotaajuutta useammassa vaiheessa. Esimerkiksi näytteenottotaajuuden pudotus yhteen kymmenesosaan alkuperäisestä voidaan tehdä yhdessä vaiheessa kertoimella kymmenen tai kahdessa vaiheessa ensin kertoimella 5 ja sitten kertoimella 2. Kolmas mahdollisuus on pudottaa ensin kertoimella 2 ja sitten kertoimella 5. Kaikilla näillä menetelmillä tarvitaan eri määrä kertoimia, ja useimmiten suora pudotus kertoimella 10 tarvitsee enemmän kuin useammassa vaiheessa tehdyt operaatiot. Alla on esitetty kaaviot eri menetelmistä, kun lähtötaajuus on 20 kHz ja tavoite on 2 kHz. Oletetaan lisäksi, että signaalista täytyy säilyttää taajuudet 900 Hertsiin asti ja että suunnittelussa käytetään FIR-suodatinta ja Hamming-ikkunaa.

- Yhdessä vaiheessa: $x(n) \rightarrow \boxed{H(z)} \rightarrow \boxed{\downarrow 10} \rightarrow y(n)$
  Nyt päästökaista on väli $[0, 0.9]$ kHz ja estokaista $[1, 10]$ kHz. Siirtymäkaista normalisoituna on näin ollen väli $[0.045, 0.05]$, joten kertoimia tarvitaan $N = 3.3/\Delta f = 3.3/0.005 \approx \mathbf{661}$.

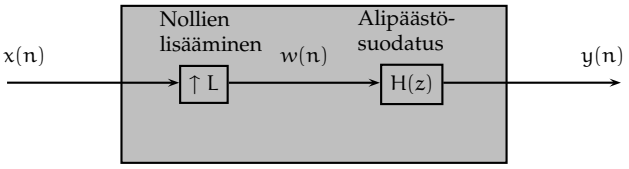
- Kahdessa vaiheessa: $x(n) \rightarrow \boxed{H_1(z)} \rightarrow \boxed{\downarrow 5} \rightarrow \boxed{H_2(z)} \rightarrow \boxed{\downarrow 2} \rightarrow y(n)$
  Nyt suotimen $H_1(z)$ päästökaista on väli $[0, 0.9]$ kHz ja estokaista $[2, 10]$ kHz. Siirtymäkaista normalisoituna on näin ollen väli $[0.045, 0.1]$, joten kertoimia tarvitaan $N_1 = 3.3/0.055 \approx 61$. Suotimen $H_2(z)$ päästökaista puolestaan on väli $[0, 0.9]$ kHz ja estokaista $[1, 2]$ kHz. Siirtymäkaista normalisoituna näytteenottotaajuudella 4 kHz on näin ollen väli $[0.225, 0.25]$, joten kertoimia tarvitaan $N_2 = 3.3/0.0250 \approx 133$. Yhteensä kertoimia tarvitaan siis **194**.

- Kahdessa vaiheessa: $x(n) \rightarrow \boxed{H_1(z)} \rightarrow \boxed{\downarrow 2} \rightarrow \boxed{H_2(z)} \rightarrow \boxed{\downarrow 5} \rightarrow y(n)$
  Suotimen $H_1(z)$ päästökaista on väli $[0, 0.9]$ kHz ja estokaista $[5, 10]$ kHz. Siirtymäkaista normalisoituna on näin ollen väli $[0.045, 0.25]$, joten kertoimia tarvitaan $N_1 = 3.3/0.205 \approx 17$. Suotimen $H_2(z)$ päästökaista puolestaan on väli $[0, 0.9]$ kHz ja estokaista $[1, 5]$ kHz. Siirtymäkaista normalisoituna näytteenottotaajuudella 10 kHz on näin ollen väli $[0.09, 0.1]$, joten kertoimia tarvitaan $N_2 = 3.3/0.01 \approx 331$. Yhteensä kertoimia tarvitaan siis **348**.

Näin ollen tehtävä on viisainta suorittaa kahdessa vaiheessa kertoimilla 5 ja 2 (tässä järjestyksessä). Yleisesti pitää paikkansa, että desimointikertoimet kannattaa sijoittaa laskevaan järjestykseen. Siksi järjestys $x(n) \rightarrow \boxed{H_1(z)} \rightarrow \boxed{\downarrow 2} \rightarrow \boxed{H_2(z)} \rightarrow \boxed{\downarrow 5} \rightarrow y(n)$ tuottaa aina enemmän kertoimia kuin $x(n) \rightarrow \boxed{H_1(z)} \rightarrow \boxed{\downarrow 5} \rightarrow \boxed{H_2(z)} \rightarrow \boxed{\downarrow 2} \rightarrow y(n)$, ja se olisi voitu alun perinkin jättää tarkastelematta.

8.3 Interpolointi eli näytteenottotaajuuden nostaminen

Signaalin interpoloinnin tarkoituksena on saada aikaiseksi signaali, joka vastaa alkuperäistä, mutta jonka näytteenottotaajuus on suurempi. Oheinen kaavio esittää lohkokaaviota interpolointioperaatiosta. Toisin kuin desimoinnissa, nyt ei tarvita esisuodatusta, koska näytteenottotaajuutta nostettaessa kaikki alkuperäiset taajuudet voidaan toki esittää.



Ensimmäinen operaatio on näytteenottotaajuuden nostaminen L-kertaiseksi. Tätä operaatiota merkitään nuolella ylöspäin ja interpolointikertoimella L, siis $\boxed{\uparrow L}$. Tämän operaation toteuttamisessa on lukuisia vaihtoehtoja; kuinka määritetään uudet ylimääräiset arvot? Matemaatikko alkaisi tässä tapauksessa luultavasti sovittaa polynomeja tai splinejä saadakseen uusia arvoja olemassaolevien arvojen välille. Signaalinkäsittelyssä tilanne hoidetaan kuitenkin toisin: tuntemattomien arvojen tilalle sijoitetaan nollat ja näin saatu signaali suodatetaan alipäästösuotimella. Nollia lisättäessä mukaan tulee hyvin suuria taajuuksia.

Suurten taajuuksien lisäämisen vaikutus on poistettava, ja se luonnollisesti tapahtuu alipäästösuodatuksella. Alkuperäisessä signaalissa suurin taajuus on $F_s/2$. Interpoloitaessa näytteenottotaajuuteen LF_s, suurin esitettävissä oleva taajuus on $LF_s/2$. Taajuutta $F_s/2$ suuremmat taajuudet on poistettava, koska ne ovat nollien lisäämisen tulosta.

Nollien lisäämisen jälkeen saatu signaali (näytteenottotaajuus LF_s) suodatetaan siis alipäästösuotimella, joka säilyttää taajuudet väliltä $[0, F_s/2]$ ja poistaa tätä suuremmat taajuudet. Koska nyt näytteenottotaajuus on interpoloinnin seurauksena LF_s, niin suotimen vaatimukset on normalisoitava tämän luvun suhteen. On siis suunniteltava alipäästösuodin, jonka vaatimukset ovat:

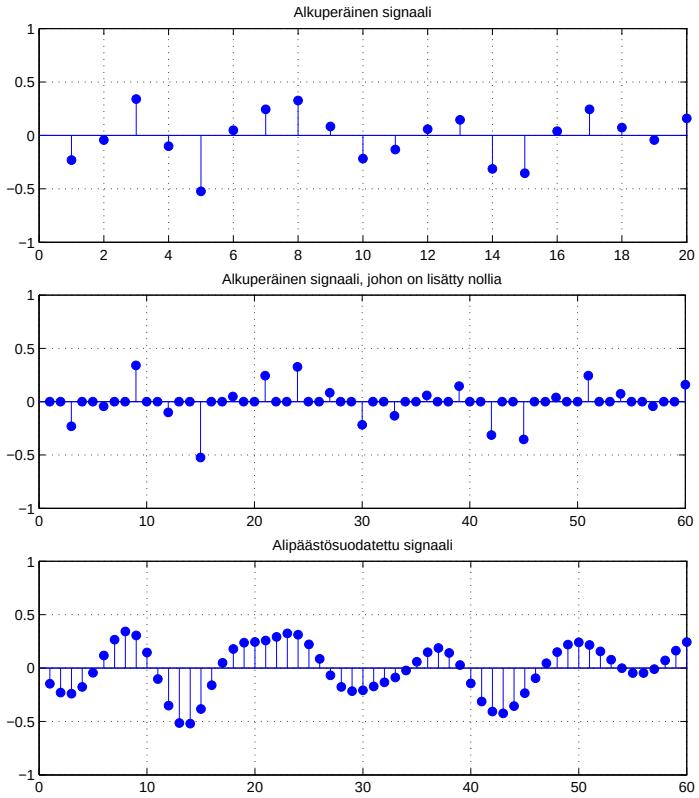
- päästökaista on normalisoituina taajuuksina ilmaistuna väli $[0, (F_s/2)/(LF_s) - \Delta f] = \left[0, \frac{1}{2L} - \Delta f\right]$. Siirtymäkaistan leveys Δf riippuu jälleen sovellutuksesta jossa interpolointia on tarkoitus soveltaa (Δf määrää osaltaan kertoimien määrän).
- estokaista on väli $[(F_s/2)/(LF_s), 1/2] = \left[\frac{1}{2L}, \frac{1}{2}\right]$.

Myös vaimennusvaatimukset määräytyvät enimmäkseen sovellutuksen mukaan, eikä niiden valinnasta voida antaa mitään yleistä ohjenuoraa. Koska alkuperäiseen signaaliin sijoitetaan $L - 1$ nollaa jokaista signaalin arvoa kohden, signaalin amplitudi putoaa suodatettaessa yhteen L:n osaan. Siksi suodatuksen jälkeen (tai sitä ennen) signaalin arvot on syytä kertoa luvulla L.

Kaavoilla esitettynä signaalin $x(n)$ interpolointiprosessi kertoimella L signaaliksi $y(n)$ on seuraava:

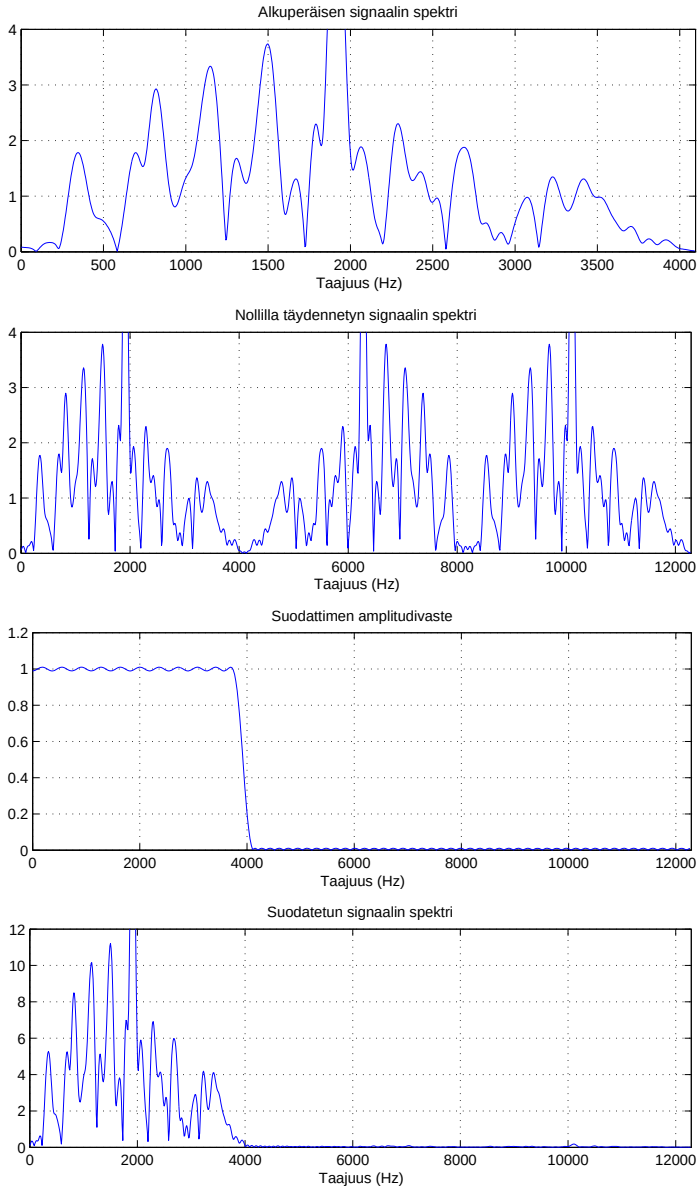
$$w(m) = \begin{cases} x(m/L), & \text{kun } m = 0, \pm L, \pm 2L, \ldots \\ 0, & \text{muulloin,} \end{cases}$$

$$y(n) = \sum_{k=-\infty}^{\infty} h(k)w(n-k).$$



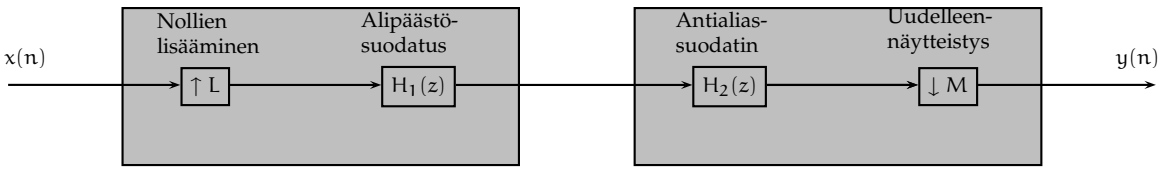
Yllä oleva kuva esittää esimerkkitapausta, jossa signaalin näytteenottotaajuus interpoloidaan kolminkertaiseksi $(L = 3)$. Ensin signaaliin $x(n)$ jokaisen kahden peräkkäisen arvon väliin sijoitetaan $L - 1 = 2$ nollaa, jolloin saadaan signaali $w(m)$. Tämä suodatetaan alipäästösuotimella, jonka estokaista on $[\frac{1}{6}, 0.5]$, päästökaista $[0, 0.1526]$ ja maksimivärähtelyarvot samat kuin desimointiesimerkissä. Tuloksena saadaan ulostulosignaali $y(m)$. Kuten desimoinnin tapauksessakin, tämä signaali on lopuksi vielä kerrottava luvulla $L = 3$, jotta signaalin energia säilyisi. Vaihtoehtoisesti käytetyn alipäästösuotimen kertoimet voidaan kertoa luvulla L, jolloin päästökaistan taajuudet vahvistuvat kolminkertaisiksi. Tulos on alimmassa edellä olleista kuvista.

Taajuustasossa tilanne on oheisen kuvan mukainen. Ylin kuva esittää alkuperäistä signaalia, jonka näytteenottotaajuus on esimerkissämme 8192 Hz. Lisättäessä signaaliin nollia, saavutetaan signaali, jonka spektri $|W(e^{i\omega})|$ on seuraavassa kuvassa. Tästä on poistettava välillä 4096 Hz – 12288 Hz olevat taajuudet esimerkiksi suotimella, jonka amplitudivaste $|H(e^{i\omega})|$ on seuraavassa kuvassa. Tuloksena on signaali $y(n)$, jonka spektri on alimmassa kuvassa.

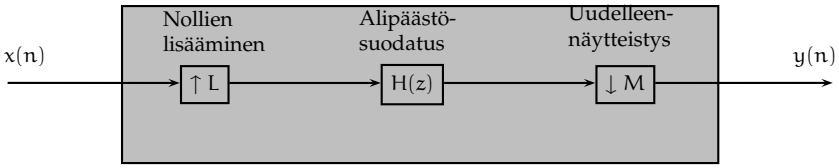


8.4 Näytteenottotaajuuden muunnos rationaalikertoimella

Kuten aiemmin mainittiin, rationaalikertoiminen näytteenottotaajuuden muunnos saadaan yhdistämällä interpolointi ja desimointi, tässä järjestyksessä. Jos siis näytteenottotaajuus halutaan $\frac{L}{M}$-kertaiseksi, interpoloidaan signaali ensin L-kertaiseksi ja desimoidaan tämän jälkeen kertoimella M. Tässä kannattaa laskennan säästämiseksi supistaa murtoluku $\frac{L}{M}$ niin pitkälle kuin mahdollista. Järjestelmän lohkokaavio on alla olevan kuvan mukainen.



Kaaviosta käy ilmi, että järjestelmässä on kaksi alipäästösuodatusta peräkkäin. Toinen näistä voidaan poistaa ja jättää jäljelle se, jonka suodatusvaatimukset ovat tiukemmat. Jos esimerkiksi kerroin $\frac{L}{M} = \frac{4}{3}$, tulee suotimen $H_1(z)$ päästökaistaksi $[0, \frac{1}{2L} - \Delta f] = [0, \frac{1}{8} - \Delta f]$ ja estokaistaksi $[\frac{1}{8}, \frac{1}{2}]$. Suotimen $H_2(z)$ vastaavat arvot ovat $[0, \frac{1}{2M} - \Delta f] = [0, \frac{1}{6} - \Delta f]$ ja $[\frac{1}{6}, \frac{1}{2}]$. Suodin $H_1(z)$ poistaa siis laajemman taajuusalueen kuin $H_2(z)$, joten pelkkä $H_1(z)$ riittää. Tässä yhteydessä täytyy luonnollisesti huomioida myös vaimennusvaatimusten toteutuminen. Alla oleva kaavio esittää näin saatavaa yksinkertaistettua järjestelmää.



8.5 Interpoloinnin käyttö D/A-muunnoksessa

Seitsemänkymmentäluvun puolivälin jälkeen alettiin tutkia mahdollisuuksia musiikin ja muun äänimateriaalin digitaaliseen tallentamiseen. Lopputuloksena syntyi CD-soitin, joka on nykyään yleisesti käytössä ympäri maailmaa. Seuraavassa tarkastellaan sen joitakin teknisiä yksityiskohtia ja erityisesti sen D/A-muunnosvaihetta, jossa digitaalinen signaali muunnetaan analogiseksi. Lisätietoja löytyy esimerkiksi Steiglitzin kirjasta[1] sekä Philipsin julkaisusta[2]. Kohinanmuokkausta tarkastellaan yksityiskohtaisemmin IEEE:n artikkelikokoelmassa[3].

CD-levyn halkaisija on 12 cm ja raidat ovat 1.6 mikrometrin etäisyydellä toisistaan. Audio-CD:n sisältämä informaatio koostuu 16 bitin näytteistä, joita on otettu 44100 Hertzin näytteenottotaajuudella. Koska stereoääni tarvitsee kaksi kanavaa, saadaan bittimääräksi $2 \times 16 \times 44100 \approx 1.41$ miljoonaa bittiä sekunnissa. Todellinen levyllä oleva bittimäärä on kuitenkin noin kolminkertainen. Ylimääräiset bitit ovat enimmäkseen virheenkorjausta varten. Pieni osa biteistä tarvitaan myös kappaleiden pituuksien ja mahdollisesti myös nimien tallentamiseen. Multirate-tekniikan yhteydessä mielenkiintoisin vaihe on CD-soittimen D/A-muunnos.

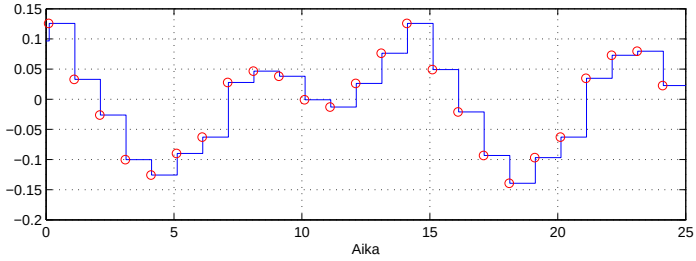
8.5.1 Nollannen asteen pitopiiri

Yksinkertaisimmillaan muunnos digitaalisesta signaalista analogiseksi tapahtuu nollannen kertaluvun pitopiirillä (engl. zero-order hold; ZOH tai sample-and-hold; S/H). Tällöin analogisen signaalin arvoksi asetetaan viimeksi tullut digitaalisen signaalin arvo. Oheisessa kuvassa esitetään erään digitaalisen signaalin muunnos analogiseksi nollannen kertaluvun pitopiirillä. Ympyrät kuvaavat digitaalisen signaalin arvoja ja viiva on analoginen approksimaatio.

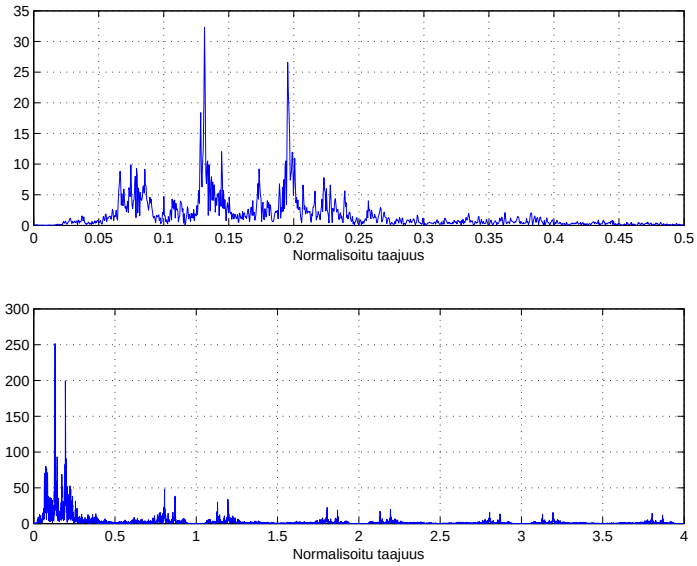
[1] Ken Steiglitz, *A Digital Signal Processing Primer, with Applications to Digital Audio and Computer Music*, Addison-Wesley, Menlo Park, CA, 1996.

[2] "Compact Disc Digital Audio,"*Philips Technical Review*, vol. 40, no. 6, 1982.

[3] *Oversampling Delta-Sigma Data Converters: Theory, Design and Simulation*, toim. J. Candy ja G. Temes, IEEE Press, New York, 1992.



Äänisignaalin ollessa kyseessä mielenkiintoisinta on kuinka analogisen signaalin spektri vastaa alkuperäisen digitaalisen signaalin spektriä. Vastaus löytyy seuraavan kuvan simuloiduista spektreistä.



Ylemmässä kuvassa on digitaalisen signaalin spektri. Vaaka-akseli esittää taajuuksia näytteenottotaajuuden suhteen normalisoituina. Alempi kuvaaja esittää analogisen signaalin spektriä nollannen asteen pitopiirin jälkeen. Nyt signaalin energia jakautuu paljon laajemmalle alueelle siten, että alkuperäinen spektri (välillä $[0, 0.5]$) monistuu suuremmille taajuuksille. Välillä $[0.5, 1]$ on vaimentunut peilikuva alkuperäisestä spektristä, välillä $[1, 1.5]$ on alkuperäisen spektrin vaimennettu kopio, välillä $[1.5, 2]$ taas peilikuva vaimennettuna, jne. Kuvassa on esitetty ainoastaan alkuperäinen spektri ja sen seitsemän kopiota, mutta itse asiassa kopioita on äärettömän monta (pienemmillä ja pienemmillä energioilla).

Nollannen asteen pitopiirin käyttäytyminen taajuustasossa voidaan esittää analyyttisesti. Koska kyseessä on jatkuva-aikainen suodin, analyysimenetelmä poikkeaa hieman johdatuskursseilla tarkastelluista. Pitopiiri voidaan ajatella suotimeksi, jonka impulssivaste on jatkuva funktio

$$h(t) = \begin{cases} 1, & \text{kun } 0 \leq t < T, \\ 0, & \text{muulloin,} \end{cases}$$

missä T on kahden näytteenottohetken välinen aikaero (CD-soittimella $1/44100$ s). Analogisen suotimen tapauksessa konvoluutio määritellään kaavalla

$$y(t) = \int_{-\infty}^{\infty} h(u)x(t-u)du$$

eli tässä tapauksessa

$$y(t) = \int_0^T x(t-u)du.$$

Tällaisen suotimen taajuusvaste on

$$\begin{aligned} H(e^{i\omega}) &= \int_{-\infty}^{\infty} h(t)e^{-i\omega t}\,dt \\ &= \int_0^T e^{-i\omega t}\,dt \\ &= \Big/_0^T \frac{1}{-i\omega}e^{-i\omega t} \\ &= \frac{e^{-i\omega T}-1}{-i\omega}. \end{aligned}$$

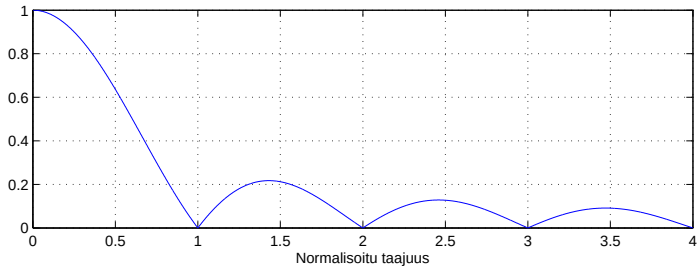
Viimeisin lauseke voidaan sieventää muotoon

$$H(e^{i\omega}) = Te^{-i\omega T/2}\,\mathrm{sinc}(\omega T/2),$$

jolloin amplitudivasteeksi tulee

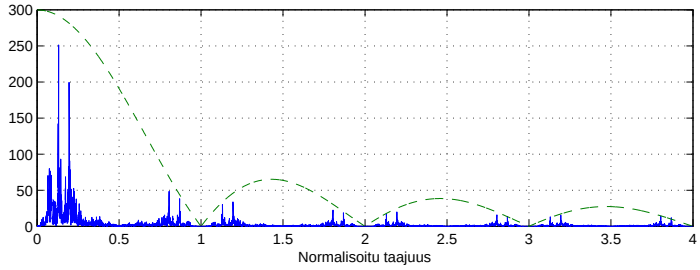
$$|H(e^{i\omega})| = T|\,\mathrm{sinc}(\omega T/2)|.$$

Tämän funktion kuvaaja on alla (nyt $T = 1$).



Digitaalisen signaalin näytteenottotaajuus on yo. asteikolla yksi ja Nyquistin rajataajuus näinollen kohdassa 1/2. Jokainen kuvaajan huippukohta tuottaa yhden vaimennetun kopion digitaalisen signaalin spektristä.

Amplitudivasteen kuvaaja sovitettuna signaalin spektriin on alla. Visualisointisyistä amplitudivaste on nyt kerrottu kolmellasadalla.



Näin havaittiin, että pelkkä pitopiiri tuottaa ylimääräisiä taajuuksia analogiseen signaaliin. CD-soittimen tapauksessa ylimääräinen energia esiintyy yli 22.05 kilohertsin taajuuksilla, joita ihmiskorva ei kuule. Ylimääräinen energia korkeilla taajuuksilla saattaa kuitenkin rasittaa vahvistinlaitteistoa, joten se on hyvä poistaa.

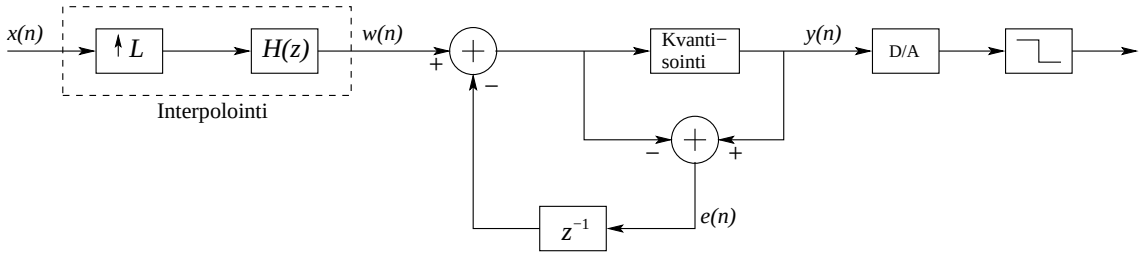
Korkeat taajuudet voitaisiin poistaa analogisella alipäästösuotimella, jonka päästökaista on esimerkiksi väli $[0, 20]$ kHz ja estokaista $[22.05, \infty)$ kHz (analogisella suotimilla käsiteltävillä taajuuksilla ei ole ylärajaa). Tällöin siirtymäkaistasta tulee kuitenkin verraten kapea, jolloin analogisesta suotimesta tulee hankala suunniteltava ja helposti kallis.

Multirate-menetelmiä käyttäen suodatus voidaan tehdä kahdessa vaiheessa, jolloin suotimista tulee yksinkertaisempia. Ensimmäisessä vaiheessa nostetaan digitaalisen signaalin näytteenottotaajuus nelinkertaiseksi (176.4 kHz). Tämä tapahtuu lisäämällä kahden näytearvon väliin kolme nollaa ja suodattamalla tulos alipäästösuotimella, jonka päästökaista on väli $[0, 20]$ kHz ja estokaista $[22.05, 88.2]$ kHz. Tämän suotimen toteutus riippuu halutusta tarkkuudesta, mutta esimerkiksi Philipsin kuvauksessa käytetään FIR-suodinta, jossa on 96 kerrointa. Kukin kerroin esitetään kahdentoista bitin tarkkuudella. Päästökaistalla suotimen amplitudivaste on nouseva niin, että nollataajuuden lähellä se on hieman alle nollan desibelin ja 20 kHz:n lähellä hieman yli nollan desibelin. Näin pyritään kompensoimaan korkeampien taajuuksien pientä vaimenemista nollannen asteen pitopiirissä.

Toisessa vaiheessa digitaalinen signaali muunnetaan analogiseksi nollannen asteen pitopiirillä ja suodatetaan lopuksi analogisella suotimella, joka poistaa suurille taajuuksille tulevan ylimääräisen energian. Nyt analogisen suotimen vaatimukset on helppo toteuttaa: päästökaista $[0, 20]$ kHz, estokaista $[88.2, \infty)$ kHz. Siirtymäkaistan leveys on nyt yli 30-kertainen aikaisempaan verrattuna. Itse asiassa siirtokaista voitaisiin venyttää yli 150:een kilohertsiin ilman ongelmia (miksi?).

8.5.2 Kohinanmuokkaus

Digitaalilaitteita käytettäessä informaation kaikkein luonnollisin esitysmuoto on binäärinen. Binääridatalla tehtävät operaatiot ovat nopeita ja niitä käyttävät laitteet yksinkertaisia suunnitella. Myös CD-soittimen alunperin 16-bittinen data muunnetaan nykyisissä soittimissa ensin binääriseksi ja vasta tämän jälkeen analogiseksi. Tällöin D/A-muunnin tuottaa vain kahta eri jännitetasoa ja se saadaan rakenteeltaan yksinkertaiseksi. Normaalisti signaalin muunnos binääriseksi aiheuttaisi voimakkaan kvantisointikohinan signaaliin, mutta siitä päästään eroon nostamalla signaalin näytteenottotaajuutta sekä siirtämällä kvantisointikohinaa korkeammille taajuuksille. Tällaisista menetelmistä käytetään nimeä *kohinanmuokkaus* (*engl. noise shaping*), ja ideana on nimenomaan siirtää kohinan energiaa suuremmille taajuuksille, josta se on helppo poistaa alipäästösuodatuksella.



Yksinkertaisin kohinanmuokkausmenettely on yllä olevan lohkokaavion mukainen. Ensimmäisessä vaiheessa signaali $x(n)$ interpoloidaan näytteenottotaajuudeltaan moninkertaiseksi. Tulos $w(n)$ kvantisoidaan ja järjestelmä laittaa muistiin syntyneen kvantisointivirheen $e(n)$. Kvantisointivirheen tallentaminen auttaa tasapainottamaan virhettä seuraavilla kierroksilla.

Yleensä kvantisointia mallinnetaan lisäämällä kvantisoitavaan signaaliin virhesignaali $e(n)$. Yllä olevan kuvan tapauksessa lohkokaaviosta saadaan yhtälö

$$y(n) = w(n) + e(n) - e(n-1).$$

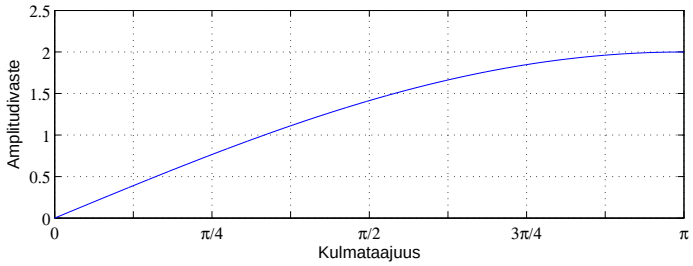
Ottamalla z-muunnokset puolittain saadaan yhtälö

$$Y(z) = W(z) + E(z)(1 - z^{-1}).$$

Näin ollen signaali $w(n)$ menee sellaisenaan järjestelmän läpi (ei suodatusta). Sen sijaan virhesignaali $e(n)$ kulkee lineaarisen järjestelmän $H(z) = 1 - z^{-1}$ läpi. Käytetään tästä ulostulosignaalista jatkossa merkintää $d(n)$:

$$d(n) = e(n) - e(n-1).$$

Järjestelmän $H(z)$ taajuusvaste on $H(e^{i\omega}) = 1 - e^{-i\omega}$, jonka itseisarvon kuvaaja on alla. Melko helposti voidaan osoittaa, että $|1 - e^{-i\omega}| = 2\sin(\frac{\omega}{2})$.



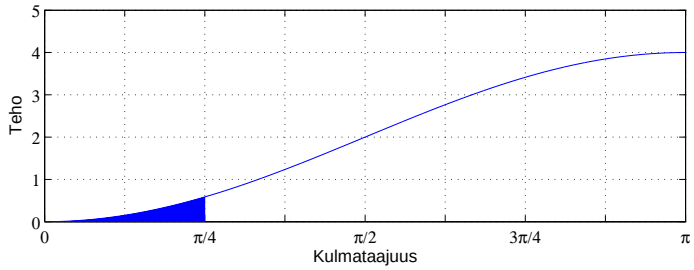
Kuviosta nähdään, että kohinan pienet taajuudet vaimenevat ja suuret taajuudet vahvistuvat. Koska signaali $w(n)$ on interpoloitu versio alkuperäisestä signaalista, sijaitsevat tärkeät taajuudet nimenomaan pienillä taajuuksilla, jossa kohina on vaimentunut. Signaalin $y(n)$ SNR riippuu nyt interpolointikertoimesta L ja siitä, kuinka moneen bittiin signaali kvantisoidaan.

Kohinan määrä järjestelmän $H(z)$ jälkeen voidaan laskea analyyttisesti käyttämällä hyväksi tietoa, että sen tehospektri riippuu valkoisen kohinan tapauksessa suoraan järjestelmän $H(z)$ taajuusvasteesta:

$$\begin{aligned} P_{dd}(e^{i\omega}) &= |H(e^{i\omega})|^2 \sigma_e^2 \\ &= \left(2\sin\left(\frac{\omega}{2}\right)\right)^2 \sigma_e^2 \\ &= 4\sigma_e^2 \sin^2\left(\frac{\omega}{2}\right), \end{aligned}$$

missä $\sigma_e^2 = 2^{-2b}/12$ ja b on bittien määrä kvantisoinnin jälkeen. Koska signaali sijaitsee interpoloinnin jälkeen taajuuksilla $\omega \in [0, \pi/L]$, meidän tarvitsee tietää tällä taajuusalueella olevan kohinan teho.

Esimerkiksi tapauksessa $L = 4$ tilanne olisi alla olevan kuvan mukainen. Käyrän alle jäävä merkitty alue vastaa signaalin kanssa samalla taajuusalueella olevan kohinan tehoa.



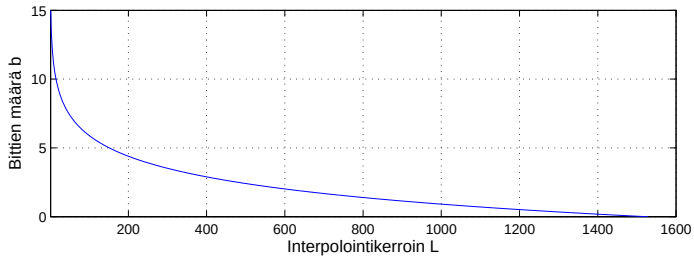
Se saadaan integroimalla,

$$\begin{aligned}\frac{1}{\pi}\int_0^{\pi/L} 4\sigma_e^2 \sin^2(\frac{\omega}{2})\,d\omega &= 4\cdot\frac{2^{-2b}}{12\pi}\int_0^{\pi/L}\sin^2(\frac{\omega}{2})\,d\omega \\ &= \frac{2^{-2b}}{3\pi}\left(\frac{\pi}{2L}-\frac{1}{2}\sin(\frac{\pi}{L})\right) \\ &= \frac{2^{-2b}}{6}\left(\frac{1}{L}-\frac{\sin(\pi/L)}{\pi}\right).\end{aligned}$$

Nyt voidaan kysyä, montako bittiä säästetään kun interpoloidaan tietyllä kertoimella L. Esimerkiksi CD-soittimen tapauksessa bittimäärä on $15+1$, ja kvantisointikohinan teho näin ollen $\frac{2^{-30}}{12}$. Samaan kvantisointikohinaan päästään interpoloinnin ja kohinanmuokkauksen avulla interpolointikertoimella L ratkaisemalla tarvittava bittimäärä b yhtälöstä

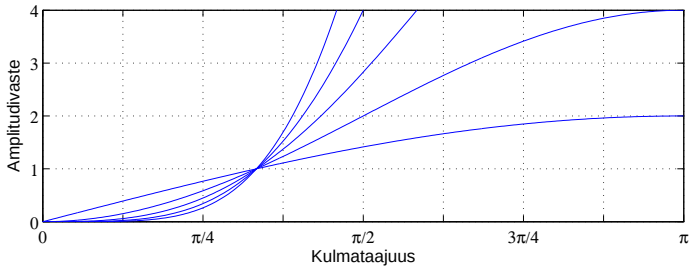
$$\frac{2^{-2b}}{6}\left(\frac{1}{L}-\frac{\sin(\pi/L)}{\pi}\right)=\frac{2^{-30}}{12}.$$

Bittien tarve eri kertoimilla on esitetty alla olevassa kuvassa. Arvoissa ei ole mukana merkkibittiä.



Kuviosta nähdään interpolointikertoimien olevan edelleenkin varsin suuria. Täysin binääriseen tilanteeseen ($b=0$) tarvitaan nyt 1523-kertainen interpolointi, mikä on liikaa. Järjestelmää on siis kehitettävä edelleen.

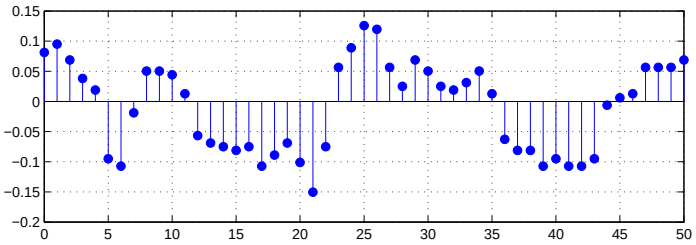
Kohinanmuokkaus tapahtuu siis yksinkertaisimmillaan viemällä kvantisointikohina järjestelmän $H(z)=1-z^{-1}$ läpi. Tulosta voidaan parantaa korottamalla siirtofunktio toiseen tai suurempaan potenssiin. Toisen asteen kohinanmuokkaus käyttää siis järjestelmää $H(z)=(1-z^{-1})^2=1-2z^{-1}+z^{-2}$, joka on melko helppo liittää aiemmin esillä olleeseen lohkokaavioon. Yleisemmin p:nnen asteen kohinanmuokkaus vie kohinan järjestelmän $H(z)=(1-z^{-1})^p$ läpi. Tällöin amplitudivasteeksi tulee $|H(e^{i\omega})|=(2\sin(\frac{\omega}{2}))^p$ ja amplitudivasteen kuvaajat arvoilla $p=1,2,\ldots,5$ on esitetty alla. Käyrät kohtaavat pisteessä ω, jossa on voimassa yhtälö $2\sin(\omega/2)=1$, eli pisteessä $\omega=\pi/3$, joka on normalisoituina taajuuksina $f=\frac{\omega}{2\pi}=\frac{1}{6}$.



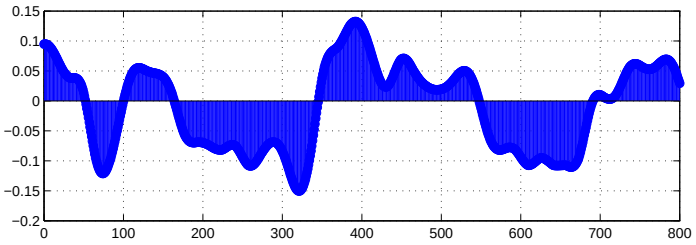
Näin ollen p:nnen asteen kohinanmuokkaimen kvantisointikohinan varianssi on

$$\frac{2^{2(p-b)}}{12\pi}\int_0^{\pi/L}\sin^{2p}\left(\frac{\omega}{2}\right)\,d\omega.$$

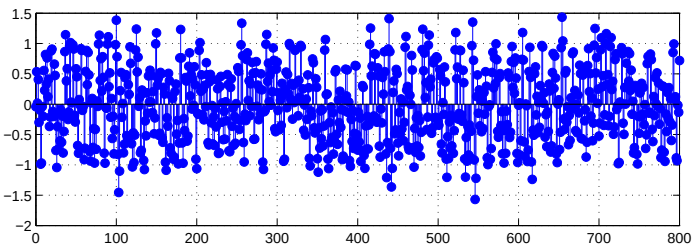
Tutkitaan lopuksi toisen asteen kohinanmuokkaajaa, kun testisignaalin alkuperäinen näytteenottotaajuus on 8192 Hz, $L = 4$ ja $b = 0$ eli kvantisoinnin tulos on binäärinen. Interpolointikerroin L on visualisointisyistä melko pieni. Alla olevassa kuvassa on pätkä testisignaalia.



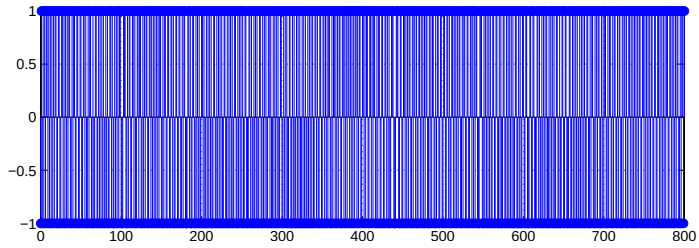
Ensimmäisessä vaiheessa signaali interpoloidaan; tulos on alla.



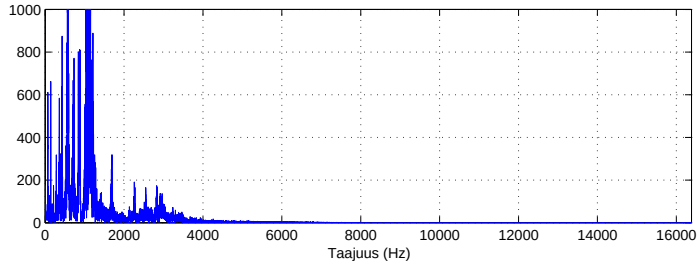
Interpoloinnin jälkeen signaali kvantisoidaan yhteen bittiin, eli käytännössä jäljelle jää vain näytteen merkki. Tällöin syntyy alla oleva virhesignaali. Huomaa, että kyseessä ei ole alkuperäisen signaalin merkki vaan mukana on myös edellisen askeleen virhesignaali.



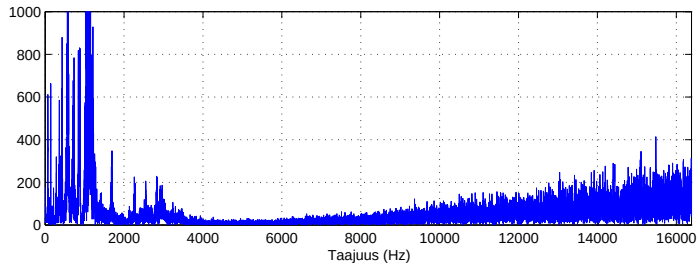
Järjestelmän ulostulosignaali näyttää seuraavalta.



Signaali ei näytä aikatasossa juurikaan samalta kuin alkuperäinen, mutta taajuustasossa yhtäläisyydet silti löytyvät. Ensin alkuperäisen interpoloidun signaalin spektri.



Kun tätä verrataan alla olevaan tulossignaalin spektriin, nähdään, että pienillä taajuuksilla spektrit ovat kokolailla samanlaisia. Lisäksi havaitaan suuremmilla taajuuksilla spektrin noudattavan amplitudivasteen $|H(e^{i\omega})| = 16\sin^4(\omega)$ muotoa. Nyt interpolointikerroin L ei ollut riittävän suuri, joten signaalin ja kohinan spektrit ovat osittain samalla taajuusalueella.



Seuraavaksi kvantisoitu signaali muunnetaan analogiseksi yksinkertaisella yksibittisellä D/A-muuntimella, jonka jälkeen analoginen signaali suodatetaan analogisella alipäästösuotimella. Analoginen suodin poistaa signaalista kohinan taajuudet. Jos interpolointikerroin ja järjestelmän aste olisivat olleet riittävän suuria, signaalin ja kohinan spektrit olisivat melko selkeästi erillään. Tällöin analogisen alipäästösuotimen suunnittelu olisi melko helppoa, koska siirtymäkaista saataisiin riittävän leveäksi.

Harjoitustehtäviä

8.1. Kuvitellun digitaalisen järjestelmän näytteenottotaajuus on 42 kHz. Jatkokäsittelyä varten signaali pitää muuntaa näytteenottotaajuuteen 6 kHz. Millaiset taajuusvaatimukset on asetettava antialias-suotimelle, jonka läpi signaali suodatetaan ennen varsinaista uudelleennäytteistystä? Signaalista tiedetään, että taajuuksien $0 - 2.5$ kHz tulee ehdottomasti säilyä desimoidussa signaalissa. (Siis missä on päästökaista ja estokaista?)

8.2. Signaali $x(n)$, jonka näytteenottotaajuus on 12 kHz pitää muuntaa signaaliksi, jonka näytteenottotaajuus on 15 kHz. Selvitä muunnoksen vaiheet lohkokaaviona käyttäen uudelleennäytteistystä ($\boxed{\uparrow L}$ ja $\boxed{\downarrow M}$) ja alipäästösuodatusta ($\boxed{H(z)}$). Esitä tarvittavien alipäästösuodinten päästö- ja estokaistojen sijainti, kun taajuudet väliltä $0 - 5.5$ kHz halutaan säilyttää.

8.3. Tavoitteena on pudottaa 12 kHz signaalin näytteenottotaajuus yhteen kilohertsiin useassa vaiheessa. Signaalin olennaisin informaatio sijaitsee taajuuskaistalla 0-450 Hz, joka tulee siis ehdottomasti säilyttää. Selvitä tarvittavien suodinten kertoimien yhteismäärä eri multistage-toteutuksissa (4 kpl riittää, jos tutkitaan vain ne tapaukset, joissa desimointikertoimet ovat laskevassa järjestyksessä), kun alipäästösuotimet suunnitellaan Hamming-ikkunalla (jolloin $N = 3.3/\Delta f$).

8.4. Kuinka monta kertolaskua kukin edellisistä toteutuksista tarvitsee sekunnissa (MPS, multiplications per second)? Tämä saadaan laskettua kaavasta

$$\mathrm{MPS} = \sum_{i=1}^{I} N_i F_i,$$

missä I on desimointilohkojen lukumäärä, N_i on i:nnen lohkon kertoimien määrä ja F_i on i:nnen lohkon sisääntulevan signaalin näytteenottotaajuus.

8.5. *(Matlab)* Toteutetaan tässä ja seuraavassa tehtävässä Matlabilla näytteenottotaajuuden muunnos kertoimella $\frac{2}{3}$. Lataa ensin testisignaali `laughter` (latautuu komennolla `load laughter` muuttujaan `y`). Signaalin näytteenottotaajuus on 8192 Hz.

Nostetaan signaalin näytteenottotaajuus ensin kaksinkertaiseksi. Tämä vaatii seuraavat vaiheet:

- Muodosta nollasignaali `z`, jonka pituus on kaksinkertainen alkuperäiseen nähden.
- Sijoita nollasignaalin joka toiseen paikkaan alkuperäisen signaalin arvot komennolla `z(1:2:end) = y;` Kuuntele tulos. Huomaa, että komennolle `soundsc` täytyy antaa myös näytteenottotaajuus; muutoin tulos kuulostaa pikkuoravilta.
- Suunnittele esim. astetta 100 oleva FIR-suodin (komento `fir1`) ja suodata syntyneet häiriöt pois.
- Kuuntele tulos ja totea, että se kuulostaa samalta kuin alkuperäinen.
- Tulosta kaikkien kolmen signaalin spektrogrammit komennolla `specgram`.

8.6. *(Matlab)* Pudota edellisen tehtävän tulossignaalin näytteenottotaajuus kolmasosaan seuraavasti.

- Suunnittele antialias-suodin, jonka aste on myös 100.
- Suodata signaali `z` kyseisellä suotimella.
- Ota tuloksesta talteen joka kolmas näyte.
- Kuuntele tulos näytteenottotaajuudella 5461 Hz, mikä on noin 2/3 alkuperäisestä.

- Tulosta kaikkien kolmen signaalin spektrogrammit komennolla `specgram`.

8.7. *(Matlab)* Toteutetaan Matlabilla ensimmäisen asteen kohinanmuokkain. Aloita Matlab-skripti lataamalla testisignaali Handel muuttujaan `y`. Sijoita `x=y;`, koska muuttujaa `y` tarvitaan myöhemmin. Interpoloi signaali kertoimella 4 käyttäen komentoa `interp`. Interpoloinnin tulos on lohkokaaviossa oleva signaali $w(n)$. Tätä signaalia lähdetään käsittelemään lohkokaavion mukaisesti. Alusta tätä varten muuttujat `e` ja `y` nolliksi: `e=zeros(1,length(w));`. Lohkokaavion mukainen kohinanmuokkaus voidaan esittää pseudokoodina seuraavasti:

```
for n := 2 to length(w)
    tmp  := w(n)-e(n-1);
    y(n) := sign(tmp);
    e(n) := y(n)-tmp;
end
```

Tulosta ruudulle ulostulosignaali ja sen spektri (`fft`:llä laskettuna) sekä virhesignaali. Desimoi signaali takaisin 8192 Hertsiin ja kuuntele tulos. Kvantisointikohina on selvästi kuultavissa signaalin taajuuskaistalla johtuen vain nelinkertaisesta interpoloinnista ja ensimmäisen asteen kohinanmuokkauksesta.

8.8. *(Matlab)* Muunna edellisen tehtävän kohinanmuokkaus toisen asteen versioksi, jossa pseudokoodi on seuraava.

```
for n := 3 to length(w)
    tmp  := w(n)-2*e(n-1)+e(n-2);
    y(n) := sign(tmp);
    e(n) := y(n)-tmp;
end
```

Tee vastaavat testit kuin edellisessä tehtävässä ja vertaa tulosta.

8.9. *(Matlab)* Muunna tehtävän 8.8 kohinanmuokkaus kolmannen asteen versioksi. Tee vastaavat testit kuin tehtävissä 8.7 ja 8.8 ja vertaa tulosta. Kuitenkin niin, että kvantisoit tuloksen $(1 + 1)$ bittiin. $(1 + 0)$-bittinen ratkaisu on numeerisesti epästabiili korkeamman asteen vuoksi. Siis ei

```
y(n) := sign(tmp);
```

vaan

```
y(n) := quant(tmp, 2^(-1));
```

8.10. Täydennä oheinen kohinanmuokkauksen lohkokaavio niin, että se esittää kolmannen asteen kohinanmuokkainta.


x(n)
L
H(z)
w(n)
Interpolointi
+
−
Kvanti−
sointi
y(n)
D/A
−
+

Luku 9

Digitaaliset signaaliprosessorit

Tässä kappaleessa tutustutaan digitaalisiin signaaliprosessoihin (engl. *digital signal processor; DSP*), eli prosessoreihin, jotka on suunniteltu nimenomaisesti digitaalisen signaalinkäsittelyn tarpeisiin.

Tietokoneiden suorittamat tehtävät voidaan jakaa kahteen luokkaan: yleiseen tiedon käsittelyyn (järjestäminen, haku, jne.) sekä aritmeettisiin operaatioihin. Perinteisesti työasemissa käytettävät yleiskäyttöiset prosessorit on suunniteltu tehokkaiksi ensisijaisesti ensimmäisessä tehtävissä. Signaaliprosessoreissa sen sijaan on erityisesti panostettu aritmeettisten tehtävien nopeaan suoritukseen. Tämä saadaan aikaiseksi toteuttamalla piiritasolla tiettyjä käskyjä, jotka tulevat usein vastaan DSP-intensiivisissä sovelluksissa. Prosessorin tulee nimittäin olla nopea kertolaskuissa ja summauksissa, sillä esimerkiksi FIR-suodatuksessa konvoluutio lasketaan tulojen summana. Usein signaaliprosessorit on optimoitu myös kompleksilukulaskentaa edellyttävän FFT:n laskemista varten.

Lisäksi signaaliprosessorien arkkitehtuuri on suunniteltu tehokkaaksi suurien datamäärien jatkuvaan käsittelyyn. Signaaliprosessorilla tiettyyn tehtävään liittyvän suoritusajan tulee nimittäin olla ennustettavissa ja mieluiten kiinteä datasta riippumatta. Prosessointi onkin signaaliprosessoreissa yleensä reaaliaikaista ja jatkuvaa. Jos prosessori on esimerkiksi osana kuulolaitetta, prosessorin pitää suoriutua tehtävästään pitkiä aikoja yhtäjaksoisesti ilman katkoksia, jotka aiheutuvat siitä, ettei prosessori pysy mukana AD-muuntimen vauhdissa.

Perinteiset yleiskäyttöiset prosessorit noudattavat von Neumannin 1940-luvulla ehdottamaa *von Neumann -arkkitehtuuria*, jossa suoritettava ohjelma ja käsiteltävä data ovat samassa muistilaitteessa. Esimerkiksi konvoluution perusoperaatio, eli kahden luvun kertominen keskenään vaatii tällöin kolme kellojaksoa (kahden luvun haku muistista ja tuloksen siirto muistiin). Koska suotimen kertoimet pysyvät kuitenkin vakioina, saadaan laskusta helposti nopeampi käyttämällä erillisiä muisteja datalle ja ohjelmalle. Tällöin suotimen kertoimet sijaitsevat ohjelmamuistissa. Prosessori voi yhden kellojakson aikana hakea sekä datamuistista että ohjelmamuistista, jolloin kerrottavien lukujen hakuaika pienenee puoleen. Erillisten muistien arkkitehtuuria kutsutaan *Harvard-arkkitehtuuriksi*, koska rakenne esiteltiin ensimmäisen kerran Harvardin yliopistossa Kaliforniassa.

Viime vuosikymmeninä signaaliprosessoreissa huomio on kiinnittynyt entistä enemmän signaaliprosessoreiden mobiilikäyttöön. Prosessoreiden täytyy olla pieniä ja kevyitä, ja niiden täytyy tuottaa vähän lämpöä ja kuluttaa vähän virtaa. Osittain tästä syystä sig-

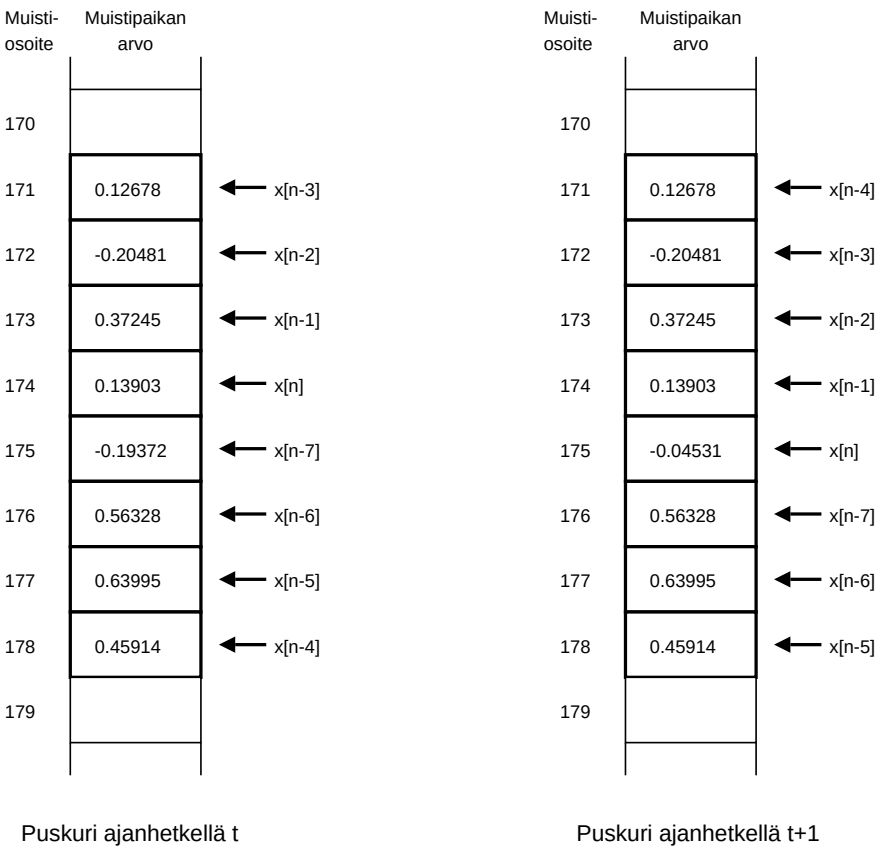
naaliprosessoreista on tullut hyvin halpoja yleiskäyttöisiin prosessoreihin nähden[1]. Halpaan hintaan vaikuttaa myös se, että tiettyä tehtävää varten valmistaja valitsee halvimman ja yksinkertaisimman tehtävään sopivan prosessorityypin, eikä tehokkainta markkinoilta löytyvää mallia. Pöytäkoneissahan tehoa ei koskaan ole liikaa, koska maksimitehoa tarvitaan aina jossain tilanteessa. Signaaliprosessorissa käsiteltävää dataa tulee sen sijaan yleensä vakiotahdilla, joten ylitehokkaan prosessorin käytössä ei ole järkeä.

9.1 Circular buffering

Signaalinkäsittelyä käyttävissä laitteissa prosessointi on yleensä siis reaaliaikaista. Toisin sanoen ulostulosignaali tuotetaan samaan aikaan kuin sisääntulosignaali saapuu laitteeseen. Ulostulo on siis saatava tuotettua lähes välittömästi herätteen tultua laitteeseen. Esimerkiksi puhelimessa viiveen tulee olla alle 10 millisekuntia, jotta puhuja tai kuuntelija eivät huomaa viivettä.

Toteutettaessa reaaliaikainen FIR-suodatus, muistissa täytyy olla kerrallaan yhtä monta signaalin näytettä kuin on suotimen kertoimien määrä. Näitä näytteitä päivitetään jatkuvasti kun uusia näytteitä saapuu. Koska näytteenottotaajuudet voivat olla suuria (esim. 44.1 kHz), tarvitaan tehokas rakenne näytteiden tallettamista ja päivittämistä varten.

Yleisesti käytössä on niin sanottu *circular buffering* -rakenne. Periaate näkyy alla olevasta kuvasta, jossa on esitetty kahdeksan näytteen puskurirakenne. Puskurin sisältö on esitetty kahdella peräkkäisellä ajanhetkellä.



[1]Kirjoitushetkellä tyypillinen listahinta on parinkymmenen dollarin luokkaa.

Rakenteen ideana on, että puskurin ensimmäisen muistipaikan ajatellaan olevan kiinnitetyn puskurin viimeiseen muistipaikkaan. Toisin sanoen kuvan esimerkissä muistipaikat 171 ja 178 ovat peräkkäisiä aivan kuten esimerkiksi muistipaikat 174 ja 175 ovat peräkkäisiä. Tietoa siitä, missä viimeisin näyte on, ylläpidetään osoittimen avulla. Kun saadaan uusi näyte, osoitin siirretään seuraavaan muistipaikkaan. Puskurirakenteen tehokkuus perustuu siihen, että vain yhtä näytettä tarvitsee päivittää uuden näytteen saapuessa.

Esimerkki: Tarkastellaan FIR-suodatuksen toteuttamista circular buffering -rakennetta käyttäen.

1. Lue arvo A/D-muuntimelta ja generoi keskeytys;
2. Käsittele keskeytys ja siirrä signaalin näyte sisääntulosignaalin puskuriin;
3. Päivitä puskurin osoitinta;
4. Käy läpi kaikki suodattimen kertoimet:
   (a) lue kertoimen arvo muistista
   (b) päivitä kertoimiin osoittavan osoittimen arvo seuraavaan kertoimeen
   (c) lue näytteen arvo puskurista
   (d) päivitä näytteisiin osoittavan osoittimen arvo seuraavaan näytteeseen
   (e) kerro näytteen arvo kertoimen arvolla
   (f) lisää tulos summa-rekisteriin (engl. *accumulator*)
   (g) toista kunnes kaikki suotimen kertoimet on käyty läpi
5. Siirrä tulos ulostulorekisteriin ja edelleen D/A-muuntimeen.

Signaaliprosessoreissa on olennaista, että esimerkin algoritmissa toistuvat operaatiot pystytään toteuttamaan nopeasti. Tärkeitä ovat erityisesti kohtaan 4 kuuluvat vaiheet, eli muistista lukeminen, osoittimen päivittäminen, lukujen kertominen ja yhteenlasku. Useissa prosessoreissa arkkitehtuuri onkin suunniteltu niin, että kaikki nämä toiminnot saadaan suoritettua yhden kellojakson aikana eli käytännössä siis yhdellä konekäskyllä. Tämä konekäsky on nimeltään MAC-operaatio (Multiply and ACcumulate), ja se kertoo kaksi lukua keskenään ja lisää tuloksen summamuuttujaan yhden kellojakson aikana. Perinteissä mikroprosessoreissa jokainen vaihe vaatisi oman käskynsä. Nykyään MAC-operaatioita voidaan suorittaa useampia yhdessä kellojaksossa, jolloin suodatus nopeutuu entisestään.

Vastaavanlaisia operaatioita on viime aikoina tullut myös yleiskäyttöisiin prosessoreihin. Intel esitteli vuonna 1997 laajennetun käskykannan MMX-prosessorit, joissa on MAC-operaatiota vastaava komento kokonaisluvuille. Lisäksi MMX-laajennuksen avulla yhden kellojakson aikana voidaan suorittaa useita kokonaislukukertolaskuja. Sittemmin Intel on laajentanut käskykantaa myös liukuluvuille SSE, SSE2 ja SSE3 -laajennusten myötä. Myös muilla prosessorivalmistajilla on nykyisin vastaavia tekniikoita käytössä.

9.2 Kiinteän vai liukuvan pilkun aritmetiikka?

Digitaalinen signaaliprosessori voidaan toteuttaa käyttäen joko kiinteän pilkun (engl. *fixed point*) tai liukuvan pilkun (engl. *floating point*) aritmetiikkaa. Molempia esitystapoja käyttäviä prosessoreja on yleisesti saatavilla.

Perinteisesti liukulukulaskenta on ollut hitaampaa kuin kokonaislukulaskenta, mutta nykyaikaisissa signaaliprosessoreissa ero on merkityksetön. Niinpä kiinteän pilkun aritmetiikan käytöstä saatava merkittävin hyöty on laitteen halvempi hinta sekä pienempi koko ja virrankulutus. Liukuvan pilkun järjestelmien etuja ovat laskennan suurempi tarkkuus ja suurempi käytössä oleva lukualue. Lisäksi näiden laitteiden ohjelmointi on helpompaa, sillä ohjelmoijan ei yleensä tarvitse huolehtia ylivuotojen ja alivuotojen tai pyöristysvirheiden välttämisestä.

Usein kiinteän pilkun aritmetiikkaa kannattaa kuitenkin käyttää. Jos esimerkiksi A/D-muunnoksessa käytetään joka tapauksessa pientä bittimäärää, ei ole välttämättä järkevää käyttää liukulukuja. Esimerkiksi usein videosignaali esitetään kahdeksalla bitillä. Vastaavasti jos toteutettavat algoritmit ovat melko yksinkertaisia, kannattaa harkita kiinteän pilkun aritmetiikkaa. Jos taas algoritmit ovat monimutkaisia, ne on paljon helpompi toteuttaa liukuvan pilkun aritmetiikkaa käyttäen. Yleisesti taajuustason algoritmit (kuten FFT) ovat melko monimutkaisia, ja ne kannattaa toteuttaa liukulukuja käyttäen.

Valintaan vaikuttaa useimmiten myös taloudellinen näkökohta. Käytettäessä kiinteän pilkun aritmetiikkaa, laitteen tuotantokustannukset saadaan painettua alas, mutta vastaavasti laitteen suunnittelukustannukset saattavat nousta korkeiksi. Liukuvan pilkun aritmetiikkaa käytettäessä tilanne on päinvastainen.

Myös prosessorin tehonkulutus on usein ratkaiseva valintaperuste. Monissa sovelluksissa laitteen on toimittava pitkiä aikoja samalla virtalähteellä (akut, paristot, tms.), jolloin tehon kulutus on ratkaiseva tekijä. Kiinteän pilkun prosessorien tehonkulutus on huomattavasti liukuvan pilkun prosessoreita vähäisempi. Tällaisissa sovelluksissa voi joskus jopa olla kannattavaa simuloida liukulukulaskentaa ohjelmallisesti kiinteän pilkun prosessorilla.

9.3 C:llä vai konekielellä?

Yleisesti signaaliprosessoreja ohjelmoidaan joko C-kielellä tai suoraan symbolisella konekielellä (assembler). Nyrkkisääntönä on, että C-kielellä ohjelmointi on helpompaa ja nopeampaa kuin konekielellä, kun taas taitava ohjelmoija saa tuotettua konekielellä nopeampaa koodia kuin C-kielellä.

Yleisprosessoreja ohjelmoitaessa assemblerin käyttö on nykyisin melko harvinaista. Signaaliprosessoreissa sen käyttö on kuitenkin edelleen hyvin yleistä, koska tyypillisesti ohjelmat ovat lyhyitä (helppoja hallita), ja nopeus on kriittinen vaatimus. Kääntäjät eivät nimittäin ole täydellisiä, eivätkä ne välttämättä pysty hahmottamaan C-kielisen ohjelman perimmäistä tarkoitusta ja tehokkainta käännöstä. Esimerkiksi FIR-suodin toteutetaan C-kielellä for-silmukan sisällä. C-kielisessä koodissa ei kuitenkaan ole enää tietoa siitä, että kyseessä on konvoluutio, joka voidaan toteuttaa tehokkaasti MAC-operaatioiden avulla. Kääntäjä joutuukin tutkimaan silmukan alku- ja loppuehtoja sekä sen sisällä käytettävien muuttujien riippuvuuksia toisistaan. Aina kääntäjä ei löydä summausoperaatiota, joka oli-

si tehokas toteuttaa MAC-operaatiolla tai pysty löytämään riippumattomia lohkoja, jotka olisi mahdollista suorittaa rinnakkain.

Useat kääntäjät muuntavat C-koodin ensin Assembler-kieliseksi, josta on mahdollista tarkastaa kuinka kääntäjä tulkitsi C-kielisen koodin.

Harjoitustehtäviä

9.1. Toteutetaan reaaliaikainen FIR-suodatin TI:n DSP Starter Kitillä C-kielisenä rutiinina. Tutustutaan tässä tehtävässä kortin käyttöönottoon.

Alkuvalmistelut. Tarkista ensin kaikki kytkennät. Tietokoneen takana olevasta äänikortin vihreästä pistokkeesta kytketään johto DSP-kortin sisäänmenoon (kotelossa lukee IN, tms.) Kotelossa on myös toinen 3,5 mm:n pistoke (OUT), johon liitetään kaiuttimien johto. Näin ollen tietokoneen tuottama ääni menee DSP-kortille, joka käsittelee sen ja lähettää edelleen ulostulon kautta kaiuttimiin. Tarkista myös, että tietokoneen takaa USB-väylästä menee vaaleanharmaa johto kortille. Tarkista vielä, että kortti saa virtaa (vihreä ledi palaa). Käytä ohutta mustaa virtajohtoa varmuuden vuoksi irti, jotta kortti on perustilassa, eikä esimerkiksi edellisen ryhmän ohjelma jää sinne pyörimään.

Kortin toimintaa testataan jatkossa soittamalla Windows Media Playerillä jompi kumpi esimerkkikappaleista (tai nettiradio tms.). Myös Matlabissa on testisignaaleita (esimerkiksi komennot `load handel.mat; soundsc(y, 8192);`). Testisignaali prosessoinnin jälkeen pitäisi saada kuulumaan kaiuttimista. Tietokoneissa on sisäiset kaiuttimet, joten voit kuunnella alkuperäisen signaalin irrottamalla audiokaapelin koneen takaa. Tällöin kone käyttää sisäisiä kaiuttimia.

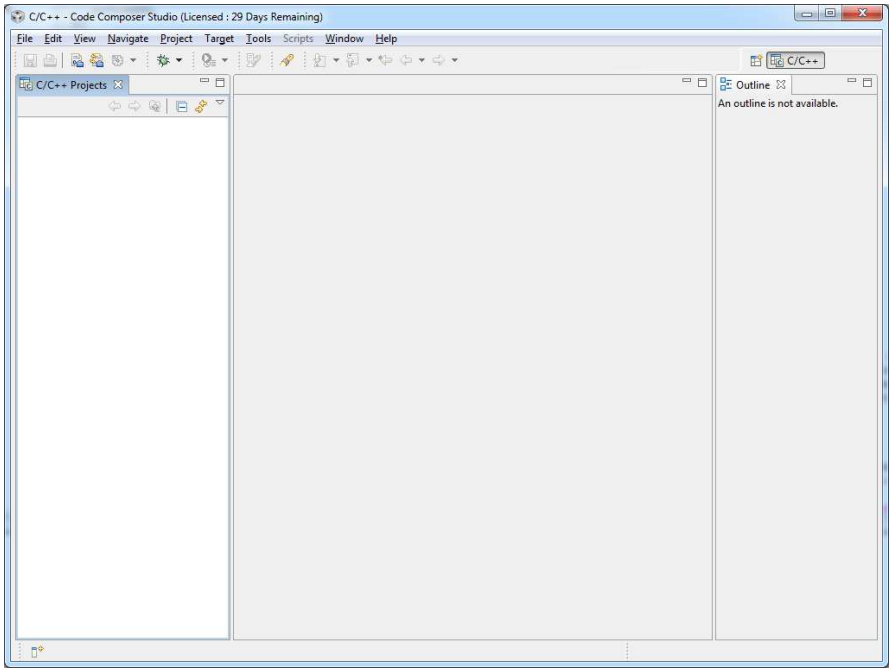
Ohjelmisto. Ohjelmointityökalu (Code Composer Studio 4.2; CCS) on asennettu, ja kuvake löytyy työpöydältä:

Ohjelman pohjana käytetään esimerkkiohjelmaa, joka löytyy osoitteesta

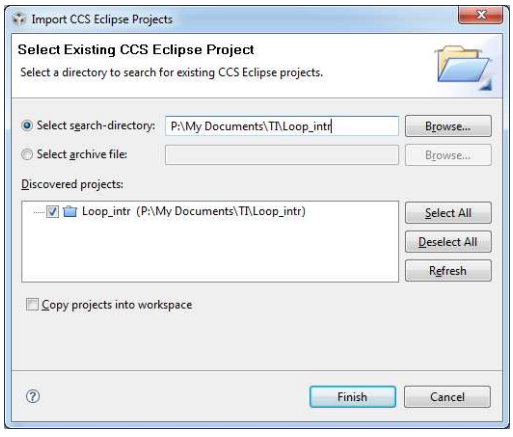
`http://www.cs.tut.fi/kurssit/SGN-1251/TI-tyo.zip`

Pura tiedosto esim. hakemistoon `P:\My Documents\TI\`.

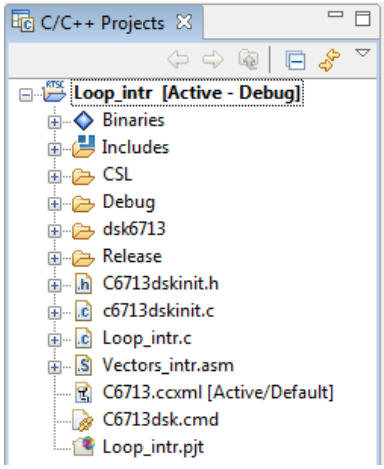
Laitteiston testaus. Käynnistä Code Composer Studio. Ensimmäisellä käynnistyskerralla ohjelma kysyy oletushakemistoa uusille projekteille, mihin kannattaa antaa oma kotihakemistosi tai sen alikansio (esim. `P:\My Documents\TI\`). Tämän jälkeen aukeaa musta Welcome-screen, jonka oikeassa ylälaidassa on CCS:n ikoni ja teksti "start using CCS". Klikkaa ikonia ja alla olevan kuvan mukainen työpöytänäkymä aukeaa.



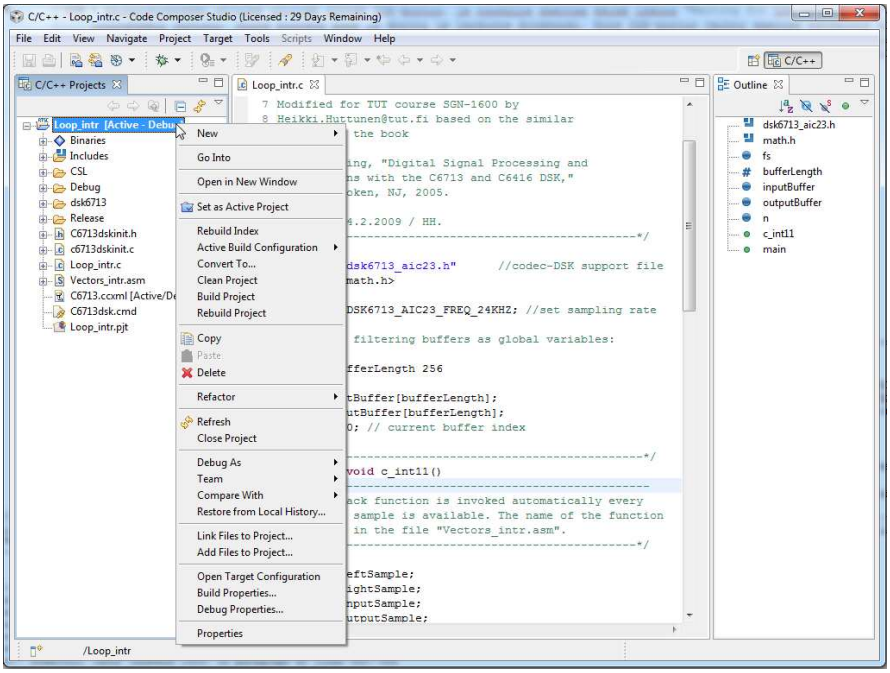
Avaa seuraavaksi lataamasi ja purkamasi projektipohja jonka purit hakemistoon `P:\My Documents\TI\`. Tämä onnistuu valikoista: Project → Import Existing CCS/CCE Eclipse Project. Aukeaa dialogi, jolle täytyy antaa hakemisto, josta projekti löytyy (siis `P:\My Documents\TI\Loop_Intr\` kohtaan Select search-directory). Jos kaikki meni oikein, CCS löytää valmiin projektipohjan ja dialogi näyttää seuraavalta.



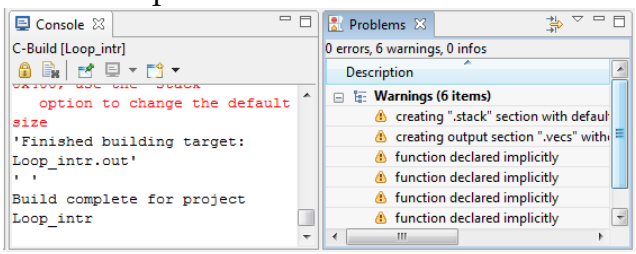
Ikkunan vasemmassa laidassa on tämän jälkeen alla olevan kuvan mukainen puu. Muokattava C-ohjelmakoodi on nyt tiedostossa `Loop_intr.c`. Avaa tiedosto ja tarkista että sieltä löytyy funktio `interrupt void c_int11()`. Laite kutsuu tätä funktiota joka kerta kun uusi näyte saapuu A/D-muuntimelta, eli 24000 kertaa sekunnissa (oletusasetuksilla näytteenottotaajuus on 24 kHz).



Ohjelman raakaversio kopioi järjestelmään tulevan signaalin sellaisenaan ulostuloon, joten on hyvä testata sillä että kaikki toimii kuten pitääkin. Ohjelman voi kääntää esim. klikkaamalla hiiren oikealla napilla avointa projektia (`Loop_intr`), jolloin aukeaa alla olevan kuvan mukainen valikko.



Ohjelma käännetään ajettavaan muotoon valitsemalla yo. valikosta "Build Project"(tai näppäinyhdistelmällä Ctrl-B). Jos kaikki meni oikein, ruudun alalaidassa olevissa ikkunoissa pitäisi olla alla olevan mukainen näkymä.



Jos Problems-ikkunassa on virheitä (errors), tulee ne korjata ennen kuin ohjelma kääntyy. Ohjelma voidaan siirtää prosessorille ja laittaa ajoon klikkaamalla alla

olevan kuvan mukaista Debug-nappia ja sieltä valintaa Debug Active Project.

9.2. Jatkoa edelliseen tehtävään. Edellinen tehtävä oli testi, jolla testattiin laitteiston toimintaa. Seuraavaksi toteutetaan FIR-ylipäästösuodin muokkaamalla ohjelmakoodia tiedostossa `Loop_intr.c`. Avaa kyseinen tiedosto.

Tiedoston alussa on määrittely

```
interrupt void c_int11()
```

Tämä on funktio, jota järjestelmä kutsuu aina kun uusi näyte on valmis käsiteltäväksi (siis 24000 kertaa sekunnissa). Funktio saa uusimmat näytteet riveillä

```
leftSample = input_left_sample();
rightSample = input_right_sample();
```

Funktio pitää itse muistissaan aiemmin tulleet näytteet taulukossa

```
short inputBuffer[bufferLength];
```

Funktio tallentaa myös vanhat ulostulonäytteet, mutta niitä ei tässä tehtävässä käytetä (koska suodin of FIR eikä IIR).

Funktion keskivaiheilla on rivi:

```
outputSample = inputSample;
```

Tämän rivin tilalle tulee FIR-suotimen toteuttava ohjelmalohko, jonka ulostulo sijoitetaan muuttujaan `Loop_intr.c`. Ohjelmalohko on yksinkertainen for-silmukka, jonka sisällä lasketaan konvoluutiosumma. Konvoluutio lasketaan ylipäästösuotimen impulssivasteen kanssa, joka suunnitellaan Matlabissa.

Tee siis seuraavat vaiheet.

(a) Suunnittele Matlabissa FIR-ylipäästösuodin ikkunamenetelmällä eli komennolla `fir1`. Suotimen rajataajuus on 3000 Hz ja laitteen näytteenottotaajuus on 24000 Hertsiä. Suotimessa tulee olla 31 kerrointa. Ennen kertoimien laskentaa anna komento `format long`, jotta tarkkuus olisi riittävä.

(b) Kopioi kertoimet[2] tiedoston `Loop_intr.c` alkuun, jossa määrittelet taulukon

```
float h[] = {0.001201261290430, 0.002048894418557,... jne. };
```

(c) Poista yllä mainittu rivi (`outputSample = inputSample`), ja laita sen tilalle for-silmukka, esimerkiksi näin:

[2]Copy-paste operaatio voi helpottua tallentamalla kertoimet ensin tekstitiedostoon (`save -ascii`) ja kopioimalla ne Notepadista. Matlabissa on myös komennot `fopen`, `fprintf`, jne., jolla tiedoston saa muotoiltua miten tahansa.

```
sum = 0.0;                  /* Tyyppi float */
for (k = 0; k < N; k++)     /* indeksin k tyyppi int */
{
   sum = sum + jotain;
}

outputSample = (short) sum;
```

For-silmukan sisällä oleva termi "`jotain`" täytyy korvata konvoluutiosummassa olevalla tulolla. Konvoluution määritelmähän kuuluu 31 näytteen mittaisen FIR-suotimen tapauksessa seuraavasti:

$$y(n) = \sum_{k=0}^{30} h(k)x(n-k).$$

Huomaa kuitenkin, että muoto $x(n-k)$ ei kelpaa sellaisenaan, koska käytössä on circular buffering, ja indeksi voi mennä negatiiviseksi. Jos näin käy, oikea arvo löytyykin indeksillä `n - k + bufferLength`.

(d) Käännä ja aja ohjelma. Soita Windows Media Playerin esimerkkitiedosto. Kaiuttimista pitäisi nyt kuulua ylipäästösuodatettu versio kappaleesta.

Yleisiä ongelmia

- Kirjoita C-kieltä, älä C++:aa. Esim. muuttujat saa määritellä vain lohkon alussa, s.o. aaltosulun jälkeen.
- Muista joka kerralla kääntää ja ladata ohjelma uudelleen laudalle.
- Indeksointi for-silmukan sisällä menee helposti väärin. Muista, että C-kielen taulukon indeksointi alkaa nollasta eikä ykkösestä kuten Matlabissa. Voit siis esim. tässä tapauksessa viitata taulukkoon `inputBuffer` indekseillä $0, 1, \ldots, 255$. Jos viittaat negatiivisella indeksillä, ohjelma luultavasti silti kääntyy ja toimii. Virhe ilmenee ainoastaan esim. napsahduksina kun ulostuloon pääsee vääriä arvoja silloin tällöin.
- Jos kertoimet ovat väärin, voi tuloksena olla ylivuoto.

Tampereen teknillinen yliopisto
PL 527
33101 Tampere

Tampere University of Technology
P.O.B. 527
FI-33101 Tampere, Finland