Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Méréstechnika és Információs Rendszerek Tanszék

Adaptív jelfeldolgozás
– hallgatói mérés tervezése –

Készítette: **Balogh Tibor Csaba**

E-mail: `tibee@tibee.hu`

Konzulens: dr. Sujbert László

2008

Nyilatkozat

Alulírott, **Balogh Tibor Csaba**, a Budapesti Műszaki és Gazdaságtudományi Egyetem hallgatója kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, és a diplomatervben csak a megadott forrásokat használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Budapest, 2008. május 16.

Balogh Tibor Csaba

Tartalomjegyzék

Ábrák jegyzéke

Kivonat

Manapság a gyakorlati alkalmazások körében egyre inkább terjed az adaptív jelfeldolgozás. Ezen alkalmazások rendszerint nagy komplexitásúak, megvalósításuk nagy számításigényt támaszt, miközben a valósidejű feldolgozás is a teljesítendő követelmények sorában áll. Napjainkban a gyártók termékpalettáján kínált korszerű jelfeldolgozó processzorok számítási teljesítménye egyre növekszik, ezzel párhuzamosan áruk egyre alacsonyabb, így elterjedtségük is növekvő tendenciát mutat a beágyazott rendszerek területén. Ez olyan célterületekre is fennáll, ahol eddig magas áruk vagy nem elégséges komplexitásuk miatt nem alkalmazták őket.

A diplomaterv célja a mesterképzés (MSc) keretében, a villamosmérnöki szak Beágyazott Információs Rendszerek szakirányán sorra kerülő *Információfeldolgozás laboratórium* című tárgya számára olyan mérés kidolgozása, mely bemutatja az adaptív jelfeldolgozási módszerek bizonyos sajátosságait. Ezen laboratórium mérései által kitűzött cél a beágyazott rendszerekben előforduló információ-feldolgozási algoritmusokban való elmélyülés segítése és a hozzájuk kapcsolódó illetve azokat kiegészítő szoftvereszközök ismertetése. A mérések során a hallgatók már az elemi digitális jelfeldolgozási ismeretek birtokában vannak, és azokat alkalmazni tudják, így a feladatok összetettebb rendszerek létrehozásából és vizsgálatából állnak. Az elvégzendő feladatok többségét valamilyen valóságos fizikai rendszer, vagy annak modellje mint mintarendszer kell, hogy támogassa.

A kidolgozott mérés címe *Adaptív szűrők vizsgálata*, és a következő témaköröket érinti:

- az LMS algoritmus megvalósítása
- adaptív nemrekurzív (FIR) szűrők vizsgálata
- identifikáció LMS algoritmussal
- adaptív vonaljavító rendszer (ALE) vizsgálata
- az LMS algoritmus változatai: az NLMS és FxLMS algoritmus megvalósítása
- adaptív visszhangcsökkentő rendszer (AEC) vizsgálata
- aktív zajcsökkentő rendszer (ANC) vizsgálata.

A diplomatervezés során sor került a mérés részletes kidolgozására, azaz az egyes mérési feladatok kijelölésére, a hozzájuk kapcsolódó fejlesztések elvégzésére, valamint mindezek alapján a részletes mérési és mérésvezetői útmutatók elkészítésére. A mérési útmutató ismerteti a téma elméleti alapjait, valamint kitér az alkalmazott hardvereszközök bemutatására is.

A mérési feladatokban kijelölt valóságos fizikai rendszerek az Analog Devices által gyártott ADSP-21364 lebegőpontos jelfeldolgozó processzoron kerültek implementálásra az ADSP-21364 EZ-KIT Lite fejlesztőkártya segítségével. A szimulációk MATLAB-ban kerültek megvalósításra.

Abstract

Nowadays the adaptive signal processing is increasingly spreading in the field of practical applications. These applications characterize high complexity, their implementation requires high computing performance and also real-time processing. Recently, the modern signal processors being offered on the manufacturers' product lists, feature high computing performance and even lower price. Therefore these devices are more and more popular in the embedded systems. This statement also stands on such application filelds where these processors couldn't be used before because of their high price or insufficent complexity.

The goal of this work is to develop a plan of laboratory measurements and exercises for the Electrical Engineering MSc course, as the part of the Embedded Systems specialization. The laboratory is ordered to demonstrate the nature of some methods of adaptive signal processing. The goal of these exercises is to help the understanding of the information-processing methods found in embedded systems and to give a review of related and complementary software components. One can presume that the students have precognition of the basic theory and application of digital signal processing. The exercises are therefore formed of synthesis and analysis of complex systems. The exercises to be performed are based on real physical systems or models as sample-systems.

The laboratory's title is *Analysis of adaptive filters*, and deals with the following topics:

- implementation of the LMS algorithm
- analysis of adaptive nonrecursive (FIR) filters
- identification with LMS algorithm
- analysis of an Adaptive Line Enhancer (ALE) system
- forms of LMS algorithm: implementation of NLMS and FxLMS algorithms
- analysis of an Adaptive Echo Cancellation (AEC) system
- analysis of an Active Noise Control (ANC) system.

During the work the detailed development of laboratory exercises has been completed, including the assignment of the measurement exercises, working out the corresponding related developments, and preparing the syllabi for students and lecturers. These documents review the theoretical background of the topic and also deal with the introducement to the hardware components being used.

The physical systems that have been assigned in some exercises, have been implemented on the Analog Devices ADSP-21364 floating point digital signal processor with the use of the ADSP-21364 EZ-KIT Lite development board. The simulations have been implemented using MATLAB.

1. fejezet

Bevezetés

1.1. A feladatkiírás értelmezése

Diplomatervem célja a mesterképzés (MSc) keretében, a villamosmérnöki szak Beágyazott Információs Rendszerek szakirányán sorra kerülő *Információfeldolgozás laboratórium* című tárgya számára mérés kidolgozása. A laboratóriumi mérések által kitűzött cél a beágyazott rendszerekben előforduló információ-feldolgozási algoritmusokban való elmélyülés segítése és a hozzájuk kapcsolódó illetve azokat kiegészítő szoftvereszközök ismertetése. A mérések során a hallgatók már az elemi digitális jelfeldolgozási ismeretek birtokában vannak, és azokat alkalmazni tudják, így a feladatok összetettebb rendszerek létrehozásából és vizsgálatából állnak. Az elvégzendő feladatok többségét valamilyen valóságos fizikai rendszer, vagy annak modellje mint mintarendszer kell, hogy támogassa.

Feladatom egy olyan mérés kidolgozása (tervezése), mely a *Méréstechnika és Információs Rendszerek Tanszék* által erre előírt ismeretanyagot mutatja be. A mérés címe *Adaptív szűrők vizsgálata* lesz, amely a következő témaköröket érinti majd:

- az LMS algoritmus megvalósítása
- az LMS algoritmus változatai
- az FxLMS algoritmus vizsgálata
- adaptív nemrekurzív (FIR) szűrők vizsgálata
- identifikáció LMS algoritmussal
- adaptív visszhangcsökkentés.

A mérés tematikája tehát előzetesen csak vázlatpontok szintjén adott, az egyes feladatok nincsenek megtervezve, a hozzájuk kapcsolódó fejlesztés még nem kezdődött meg. A diplomaterv célja a mérés kidolgozása, a mérési feladatok kijelölése, a hozzájuk kapcsolódó fejlesztések elvégzése, valamint mindezek alapján a részletes mérési és mérésvezetői útmutatók elkészítése.

A feladatokban kijelölt valóságos fizikai rendszereket az ADSP-21364 jelfeldolgozó processzoron, a szimulációkat pedig MATLAB-ban kell megvalósítani.

1.2. A feladat célja

A manapság jellemző tendenciák szerint egyre jobban terjed az adaptív jelfeldolgozás gyakorlati alkalmazása. Az adaptív szűrők és az azokat felhasználó bonyolultabb struktúrák így mára a villamosmérnök eszköztárát már nem csak elméleti síkon, hanem gyakorlati szinten is bővítik. Napjainkban a gyártók termékpalettáján kínált korszerű jelfeldolgozó processzorok számítási teljesítménye egyre növekszik, ezzel párhuzamosan áruk egyre alacsonyabb, így elterjedtségük is növekvő tendenciát mutat. Jellemzően olyan területeken is egyre nagyobb teret hódítanak, ahol eddig magas áruk vagy nem elégséges komplexitásuk miatt nem alkalmazták őket. A mai, korszerű jelfeldolgozó processzorok tehát olyan gyakorlati problémák megoldására kínálnak lehetőséget, amelyek korábban elérhetetlenek és elképzelhetetlenek voltak – többek között – a bonyolult valósidejű jelfeldolgozás igénye miatt.

A diplomaterv célja olyan hallgatói mérés kidolgozása, mely bemutatja ezen módszerek bizonyos sajátosságait. A hallgatói mérésben nagy súlyt kell fektetni az adaptív jelfeldolgozás elméleti alapjaira és az alapstruktúrák gyakorlati megvalósítására. A bonyolultabb alkalmazások felületes ismertetése vagy működés közben történő bemutatása nem tartozik a mérés célkitűzésébe.

1.3. A dolgozat felépítése

Diplomatervem első részében a modellillesztés és az adaptív jelfeldolgozás témakörének elméleti alapjait tekintem át a későbbiek megértéséhez szükséges mértékben. Bemutatom továbbá az alapvető, de lényegesebb adaptív jelfeldolgozó algoritmusokat is. A dolgozat második részében ezen algoritmusok lehetséges gyakorlati alkalmazásait ismertetem. A harmadik részben azon struktúrák konkrét megvalósításáról van szó, melyek a mérési feladatok során előkerülnek. Ismertetést adok egyrészt az alkalmazott hardver- és szoftvereszközökről, másrészt a konkrét megvalósított struktúrákról, különös tekintettel ezek a mérésben játszott szerepére, valamint választásaim indoklására.

A dolgozat függelékében a mérési illetve mérésvezetői útmutatót készítettem el. A mérési útmutató tartalmazza a feladatok eredményes elvégzéséhez szükséges elméleti ismeretanyagot, a struktúrák ismertetését valamint a feladatok részletes leírását. A mérésvezetői útmutatóban megadom a feladatok egy lehetséges megoldását valamint azok magyarázatát egyaránt.

2. fejezet

A mérés tervezése

Jelen fejezetben ismertetni kívánom azokat a megfontolásokat valamint szakmai döntéseket, melyeket a mérés kidolgozása során a diplomaterv-kiírás alapján figyelembe kellett vennem. Emellett beszámolok az ehhez szükségesnek bizonyult és elvégzett tájékozódási tevékenységeimről és az elvégzett feladataimról egyaránt.

2.1. A mérés kidolgozásának szempontjai

A mérés kidolgozásának megkezdése előtt konzulensemmel egyeztetésre került a mérés leendő tematikája valamint az egyes témák áttekintésének irányvonala, elvárt mélysége. A mérés tervezése során mindezek alapján figyelembe vett szempontok a következők voltak:

- A tervezés során fel kellett tételezni, hogy a hallgatók a hasonló témával foglalkozó, a BSc képzés tárgyai között szereplő *Beágyazott és ambiens rendszerek laboratórium* megfelelő méréseit sikeresen végrehajtották és az ott tanult ismereteket elsajátították.
- A jelen diplomaterv keretében kidolgozandó mérést az MSc képzés *Információfeldolgozás Laboratórium* tárgya számára kell elkészíteni. Az MSc képzés jellegéből adódóan a BSc képzéssel szemben inkább elméleti, tudományos szemléletmódot követ. Ez azt jelenti, hogy a korszerű, ellenben komplex, nehezen átlátható struktúrájú és széleskörű ismereteket igénylő gyakorlati példák egyszerű bemutatásával szemben a mérés során a téma alapvető algoritmusaira és legegyszerűbb működésű elrendezéseire kell koncentrálni.
- A kiszabott feladatokban olyan hardvereszközök és szoftverek alkalmazását szabad előírni, melyek a laboratóriumban megfelelő számban vagy példányban rendelkezésre állnak. Előzetes ismeret, hogy a mérés során minden egyes mérőhelyen rendelkezésre áll egy oszcilloszkóp, függvénygenerátor, egy ADSP-21364 EZ-KIT Lite jelfeldolgozó kártya [8], valamint egy személyi számítógép Microsoft Windows XP vagy azzal egyenértékű operációs rendszerrel, MATLAB szoftvercsomaggal, valamint a jelfeldolgozó kártya működtetésére szolgáló VisualDSP++ integrált fejlesztői környezettel. A mérőhelyeken rendelkezésre állnak továbbá hangszórók is az akusztikai jelek meghallgatásához.
- A mérés során olyan feladatokat kell kialakítani, amelyek az adaptív jelfeldolgozó algoritmusok elvi működésének megértését hivatottak elősegíteni. A hallgatóknak képességet kell

szerezniük ezen algoritmusok implementálására. A komplex, de nagyobb gyakorlati jelentőségű alkalmazások bemutatása a BSc képzés megfelelő laboratóriumi méréseinek keretében előzetesen megtörtént.

- A feladatoknak szükség szerint egyaránt magukba kell foglalniuk a jelfeldolgozó kártyán megvalósított struktúrák üzemeltetését, paramétereinek megfelelő meggondolások után elvégzett hangolását, a struktúrák jellemzőinek bemérését, valamint a megfelelő számítógépes szimulációk elvégzését is.

- A hallgatóktól elvárható a MATLAB programcsomag előzetes ismerete olyan szinten, hogy egy, az útmutatóban pontosan közölt, jól definiált algoritmust minden nehézség nélkül a helyszínen meg tudjanak valósítani abban az esetben, ha az semmi különleges eszköz alkalmazását nem igényli.

- A hallgatók előzetesen rendelkeznek a digitális jelfeldolgozás valamint jel- és rendszerelmélet következő területeinek ismeretével:

  – mérési eljárások, rendszerelmélet és modellalkotás
  – determinisztikus és sztochasztikus jelek
  – idő-, frekvencia- és komplex körfrekvencia-tartomány
  – Fourier, Laplace és z-transzformáció, FFT algoritmus
  – digitális szűrők
  – modellillesztés alapjai.

A mérés a fentiek alapján a témaköröket tekintve láthatóan erősen behatárolt, de a témákon belül jelentős szabadságfokot tartalmaz a feladatok kitűzését illetően. A mérés tervezése során elvégzett feladataimat a következőkben ismertetem.

- Elsőként elengedhetetlen volt az adaptív jelfeldolgozás témaköre elméleti alapjainak részletes áttekintése annak érdekében, hogy eldönthetővé váljon, ebből mely ismeretek felidézése szükséges a mérésre való előzetes felkészülés során, valamint mely elméleti alapokra helyezett gyakorlati elrendezések kerüljenek a mérés tematikájába.

- Az elméleti áttekintés során megfelelően rövidített, de minden a későbbiek során szükséges, lényeges információt tartalmazó elméleti összefoglalót készítettem, melyet a hallgatók a mérés előtt, önállóan fel kell, hogy dolgozzanak, és tudásukat szükség szerint ki kell, hogy egészítsék.

- Fel kellett derítenem a tématerület lehetséges gyakorlati alkalmazásait, majd a kiválasztott gyakorlati struktúrákat részletesen is át kellett tekintenem annak érdekében, hogy azokat a legmegfelelőbb módon be tudjam mutatni a mérési feladatok során.

- A kiválasztott struktúrák egy része nem igényli konkrét fizikai rendszer előzetes elkészítését azok működésének megértéséhez. Számítógépes szimulációs eljárások során ellenben előzetesen ismert módon, megszabott adatokon történnek a vizsgálatok, amelyből kifolyólag több alapvető megfigyelést is lehet tenni a feladatok elvégzésekor a rendszereket és algoritmusokat illetően. Feladatom volt tehát a megfelelő alkalmazások bemutatására MATLAB szimulációk készítése.

- Vannak olyan struktúrák is, melyek működése csak valós időben megfelelően „látványos”, ugyanis az egyszerű modellen alapuló szimuláció nem tenné lehetővé a rendszerparaméterek hangolásával történő kísérletezést és a szükséges következtetések levonását. Alkalmasabb, komplexebb szimuláció megvalósítása viszont bonyolultsága miatt elterelné a figyelmet a lényegesebb kérdésekről. Feladatom részét képezte tehát bizonyos kiválasztott struktúrák jelfeldolgozó processzorra történő megvalósítása is. Mivel a mérések során rendelkezésre áll az ADSP-21364 EZ-KIT Lite fejlesztőkártya [8], a jelfeldolgozó processzoron megvalósított rendszerek nagyon egyszerűen üzembe helyezhetők és vizsgálhatók.

- A jelfeldolgozó kártyára történő fejlesztést az Analog Devices által rendelkezésre bocsátott VisualDSP++ integrált fejlesztői környezet teszi lehetővé. A környezet bonyolult, működtetése átfogó előzetes ismeretet igényel mind magáról a környezetről, mind a jelfeldolgozó processzorokról is. Ha ennek használata kiküszöbölhető, a hallgatókra nem nehezedik a mérés során irreleváns, de esetleg komoly nehézségeket jelentő feladatok elvégzése. Az önálló laboratóriumi munka során elvégzett eredményekre támaszkodva lehetővé vált egy olyan MATLAB alapú grafikus kezelői felület elkészítése a jelfeldolgozó kártya működtetésére, amely teljes mértékben kiküszöböli a fejlesztői környezet használatát, valamint lehetővé teszi a megvalósított struktúrák paramétereinek állítását is.

- A feladatok kidolgozását megelőző felkészülésem részét képezte a BSc képzés vonatkozó méréseinek áttekintése, megismerése is.

- Az elméleti ismeretek áttekintésével, valamint a fizikailag és szimulációban vizsgálandó struktúrákkal összhangban az elvégzendő legnagyobb feladat a mérési feladatok részletes kijelölése volt.

- A mérési feladatok alapján részletes útmutatót kellett készíteni a hallgatók számára, amely érthetően ismerteti az elméleti alapokat, bemutatja a mérés során vizsgált rendszereket és kijelöli az elvégzendő mérési feladatokat. A feladatokat olyan szinten kellett specifikálni, hogy kellően meghatározottak legyenek a sikeres megvalósításhoz, de lehetőséget biztosítsanak a hallgatók kreativitásának alkalmazására, mindemellett elősegítsék azon kvalitatív következtetések levonását, melyek az adott struktúrák működéséből láthatók.

- A mérési útmutató mellett a feladatok lehetséges megoldásait és az általam szükségesnek ítélt magyarázatokat a mérésvezetők számára készített útmutatóban adom meg. Ez az útmutató leírja a feladatok elvégzése, a rendszerek kipróbálása után levont tapasztalatokat is.

A mérés alaptémája, a modellillesztés és az adaptív algoritmusok átfogó áttekintése után nyilvánvalóvá vált, hogy az elméleti anyag ismeretét még abban az esetben sem szabad fenntartások nélkül ismertnek feltételezni, és a feladatokat közvetlenül az előzetes tanulmányokra támaszkodva felépíteni, ha a képzés tematikája alapján valamely tárgyban a téma előzetesen szerepelt. Hosszú távon fontosabb az alapismeretek elmélyítése, mint bonyolultabb alkalmazások felületes és múlandó ismeretanyagot nyújtó vizsgálata.

Ezen megfontolásokból a mérési feladatok kialakítása során azt az elvet követtem, hogy az elméleti áttekintésben szereplő anyag megértését a legjobban az teszi lehetővé, ha az ismertetett algoritmusokat a hallgatók MATLAB-ban megvalósítják. Ily módon a feladat megoldásához elengedhetetlenül szükségessé válik az elméleti anyag készség szintű ismerete. Természetesen ehhez minden lehetséges segítséget meg kell adni mind a helyszínen, mind az otthoni felkészüléshez. Az

elvárt feladat komplexitása ezzel együtt sem becsüli túl a mérésre fordítható időkeretet, mert az egyes algoritmusok az elsőként elkészített, megfelelően működő kód kiegészítésével, alkalmas módosításával egyszerűen megvalósíthatók. Egy esetlegesen adódó probléma vagy félreértés esetén pedig a helyesen működő kód a mérésvezető rendelkezésére áll, és a hallgatókat segíteni tudja.

A szimulációk sikeres elvégzése után következik a jelfeldolgozó kártyán működő struktúrák vizsgálata. A szűkös idő miatt ez a művelet is teljes egészében MATLAB alatt történik, minden háttérművelet a hallgatók elől rejtve marad annak érdekében, hogy a jelfeldolgozási problémákra lehessen koncentrálni a technikai problémák leküzdése, vagy a bonyolult fejlesztőkörnyezetek megismerése helyett. Az elkészített MATLAB felület lehetőséget biztosít a struktúrák minden fontos paraméterének módosítására. Ide tartozik például a bátorsági tényezők értéke.

3. fejezet

Az adaptív jelfeldolgozás elméleti alapjai

A diplomatervben kidolgozásra kerülő hallgatói mérések feladata az adaptív jelfeldolgozó algoritmusok működésének ismertetése, valamint a különböző algoritmusokra épülő gyakorlati alkalmazások bemutatása. A mérések kidolgozásához előzetesen mindenképpen szükséges az adaptív jelfeldolgozás elméleti hátterének átfogó áttekintése, hogy a későbbiek során megfontolt döntéseket lehessen hozni arra vonatkozólag, hogy mely algoritmusokat célszerű bemutatni elméleti szinten, illetve mely struktúrák kerüljenek a mérés során szimulációs, és melyek jelfeldolgozó kártyán realizált struktúrák mérési feladataiba.

Ebben a fejezetben röviden áttekintem tehát a modellillesztés problémakörébe tartozó paraméterbecslést, a Wiener-szűrőket, valamint az LMS algoritmusról és annak különféle változatairól is rövid ismertetést nyújtok.

3.1. Bevezető

Manapság egyre több alkalmazásban van szükség illetve lehetőség adaptív jelfeldolgozás beépítésére. Ezek legnagyobb csoportja az akusztikával van szoros kapcsolatban. Az akusztikai tér átvitele ugyanis két fizikai hely – például hangszórók és mikrofonok, vagy természetes zajforrások és az emberi fül – között jellemzően változó. Az akusztikai átvitelt nagyon sok tényező befolyásolja, melyek közül példaként említhető a belső téri berendezés, az emberek, tárgyak elhelyezkedése, a légnyomás, hőmérséklet, páratartalom, vagy akár a nyílászárók nyitott vagy csukott állapota egyaránt.

Számos gyakorlati feladatban alkalmazott algoritmus megfontolásai szerint szükség van viszont adott akusztikai átvitel vagy átvitelek megfelelő pontosságú ismeretére. Ez az előző szempontokból kiindulva, az átvitel sztochasztikusan változó jellegű voltának ismeretében nem egyértelmű feladat. Lehetővé kell tenni tehát valamilyen módon egy ismeretlen és az időben változó átvitelű rendszer modelljének kielégítő pontosságú előállítását, valamint azt, hogy a modell a fizikai rendszer változásait valós időben megfelelő gyorsasággal kövesse.

A probléma tehát a modellillesztés elméleti feladatát tűzi ki célul, melynek része a modell parametrikus vagy strukturális meghatározása, vagy a rendszerrel hasonló viselkedésű, de eltérő felépítésű modell kialakítása.

3.2. A modellillesztés elméleti alapjai

A modellillesztés feladata – egy mondatban összefoglalva – egy ismeretlen rendszernek megfelelő modell strukturális meghatározása, illetve adott esetben az előzőleg ismertnek feltételezett struktúra paramétereinek meghatározása (*paraméterbecslés*) [1]. A gyakorlatban leginkább a paraméterbecslési feladattal találkozunk, melynek során az ismeretlen rendszerre legjobban illeszkedő paraméterekkel rendelkező, előzetesen adott struktúrájú modellt keressük.

A paraméterbecslés során a valóságos rendszer és a hozzá illesztett modell kimeneteinek eltérését adott hibakritérium alapján kívánjuk minimalizálni. A modellillesztés feladatköre ezen belül is két csoportra bontható. Az *identifikáció* célja az időben állandónak feltételezett rendszer paramétereinek meghatározása, míg az *adaptáció* feladata az időben változó rendszerparaméterek követése. Az identifikáció során tehát nagy pontosságú mérés elvégzésére van elvi lehetőség, ugyanis a rendszert a szükséges hosszú ideig megfigyelhetjük, a szükséges mennyiségű adatot gyűjthetjük be róla. Az identifikáció ebből következően jellemzően időigényes folyamat. Az adaptáció során kevésbé fontos a nagyfokú pontosság és a modellnek a valóságos rendszerhez való pontos illeszkedése. Sokkal inkább lényeges, hogy a modell paramétereinek változtatásával a fizikai rendszer változásait követni tudjuk, és a modell a rendszerrel valós időben tudjon együtt mozogni. Természetesen a fizikai rendszer struktúrája nem ismert, és paraméterei sem mérhetők, kizárólag a fizikai rendszer és a modell kimenetének eltérése adhat támpontot a modell paramétereinek korrigálásához. Ehhez az esetek döntő többségében le kell mondanunk a nagy pontosságú modellről, és helyette egy egyszerűbb, a gyakorlatban könnyebben kezelhető struktúrát kell alkalmaznunk, mely még elfogadható a valóságos rendszer modellezésére.

3.2.1. Regressziószámítás

A modellillesztés egy speciális esete a regressziószámítás feladata, amely célja bizonyos változók között fennálló determinisztikus kapcsolat meghatározása. Ezen változók a gyakorlatban legtöbbször bizonyos rendszerek be- és kimenetei. A valóságos rendszerek, melyekre modellt kívánunk illeszteni, a bemenetük és kimenetük között azonban nem tisztán determinisztikus függvénykapcsolatot valósítanak meg, mert a kimenet rendszerint zajjal terhelt, tehát a determinisztikus mellett sztochasztikus komponenst is tartalmaz. Az illesztendő modell ellenben kizárólag determinisztikus függvénykapcsolatot realizál, melynek kialakítása tipikusan bizonyos szabad paraméterek valamilyen szempontból megválasztott elv alapján történő beállítását jelenti a mérések után rendelkezésre álló bemenet-kimenet adatpár-sorozat alapján.

A paraméterek hangolásához egy a valóságos rendszer és az illesztett modell kimeneteinek különbségétől függő ún. *költségfüggvényt* definiálunk, és a paraméterek hangolásával ennek minimális értékét kíséreljük meg beállítani. Célunk tehát annak elérése, hogy a modell kimenete a lehető legjobban közelítse a valóságos rendszer kimenetét, a „lehető legjobban” kritérium egzakt leírása pedig a költségfüggvény vagy hibakritérium-függvény kifejezése [1]:

$$\varepsilon = \mathrm{E}\{(\mathbf{y} - \hat{\mathbf{y}})^T(\mathbf{y} - \hat{\mathbf{y}})\}. \tag{3.1}$$

A költségfüggvény egy előnyös megválasztása az, ha értéke a kimenetek különbségétől, vagyis a modell hibájától négyzetesen függ, ugyanis az ilyen költségfüggvény minimum-keresése matematikailag egyszerű, és emellett fizikailag a hibajel teljesítményének minimalizálását végezzük. Az olyan szűrőt, melynek együtthatóit ilyen módon definiált költségfüggvény értékének minimalizálása ér-

dekében hangoljuk, *Wiener-szűrőnek* nevezzük.

A paraméterillesztés elvégzéséhez azonban nem elegendő a mért értékpárok pillanatnyi értékeinek ismerete, szükség van azok statisztikai jellemzőinek, vagy legalábbis bizonyos fokú momentumaiknak ismeretére. A lineáris regresszió alkalmazásakor például elegendő, ha az első- és másodfokú momentumok rendelkezésre állnak.

Ha az illesztendő modell speciálisan a paraméterek lineáris függvénye, és négyzetes költségfüggvényt alkalmazunk, a hibafüggvény a paraméterek síkja fölött kifeszített felület, melynek minimumkeresése után azonnal egyértelműen kiadódik a paraméterek optimális értéke. Ehhez viszont szükséges a hibafelület teljes ismerete.

Az illesztendő modell bemenete, kimenete és paraméterei közötti összefüggés a következő:

$$\hat{\mathbf{y}} = \hat{\mathbf{g}}(\mathbf{u}) = \hat{\mathbf{g}}(\mathbf{W}, \mathbf{u}), \tag{3.2}$$

ahol $\mathbf{u}$ a bemenőjel-vektor, $\mathbf{W}$ az állítható paraméterek mátrixa, $\hat{\mathbf{y}}$ pedig a modell kimenete. A bemenet és kimenet, valamint a modell kimenete mind-mind legalább gyengén stacionárius sztochasztikus folyamatok bizonyos realizációi. A jelölés kifejezi, hogy a modell a $\mathbf{W}$ paramétermátrixon keresztül adaptálható.

A modell dinamikus és esetlegesen nemlineáris részét ($f(\mathbf{u}) = \mathbf{x}$) célszerű rögzíteni, és különválasztani a lineáris, adaptálható résztől ($\mathbf{W}$):

$$\hat{\mathbf{y}} = \hat{\mathbf{g}}(\mathbf{W}, \mathbf{u}) = \mathbf{W}^T f(\mathbf{u}) = \mathbf{W}^T \mathbf{x}, \tag{3.3}$$

ahol $\mathbf{x} = f(\mathbf{u})$ a modell rögzített része, amely az $\mathbf{u}$ bemenőjel-vektorból előállítja az $\mathbf{x}$ regressziós vektort, amely ezután az adaptálható rész bemenetét képezi. Az $\hat{\mathbf{y}}$ egyetlen kimenet esetén két vektor skaláris szorzata, több kimenet esetén viszont a képletből is látható módon mátrix-vektor szorzatként áll elő. A kifejezések alakja egy bemenet és egy kimenet esetében:

$$\hat{y} = \hat{g}(\mathbf{w}, u) = \mathbf{w}^T f(u) = \mathbf{w}^T \mathbf{x}. \tag{3.4}$$

A sztochasztikus folyamatokról az időfüggvényekre (a folyamatok realizációira) áttérve:

$$\hat{y}(n) = \mathbf{w}^T(n)\mathbf{x}(n) = \sum_{i=1}^{M-1} w_i(n) x_i(n), \tag{3.5}$$

ahol M a $\mathbf{w}$ és $\mathbf{x}$ vektorok hossza, a mátrix-vektor szorzat pedig vektorok skaláris szorzatává egyszerűsödik, és (3.5) alapján számítható.

A szétválasztás eredményeképpen az adaptálható rész paramétereiben lineáris, és a paraméterek megváltoztatása által okozott tranziens véges idő alatt kifut a rendszerből.

Mivel már felírtuk a modell kimenetének kifejezését az $\mathbf{x}$ regressziós vektor illetve a $\mathbf{w}$ paramétervektor segítségével (3.4 és 3.5), írjuk fel a modellillesztéshez használt kritériumfüggvényt (3.1) az átlagos négyzetes hiba alapján:

$$\varepsilon(n) = \mathrm{E}\{(y(n) - \hat{y}(n))^2\} = \mathrm{E}\{(y(n) - \mathbf{w}^T\mathbf{x}(n))^2\} = \tag{3.6}$$

$$= \mathrm{E}\{y^2(n)\} - 2\underbrace{\mathrm{E}\{y(n)\mathbf{x}^T(n)\}}_{\mathbf{p}^T}\mathbf{w} + \mathbf{w}^T\underbrace{\mathrm{E}\{\mathbf{x}(n)\mathbf{x}^T(n)\}}_{\mathbf{R}}\mathbf{w} = \tag{3.7}$$

$$= \mathrm{E}\{y^2(n)\} - 2\mathbf{p}^T\mathbf{w} + \mathbf{w}^T\mathbf{R}\mathbf{w}, \tag{3.8}$$

ahol $\mathbf{p}^T$ a kimenet és a regressziós vektor közötti keresztkorrelációs vektor, illetve $\mathbf{R}$ a regressziós vektor autokorrelációs mátrixa. A kritériumfüggvény (3.8) szerinti kifejezése egy $M+1$-dimenziós térben elhelyezkedő paraboloidot ír le. A kritériumfüggvény minimuma az M-dimenziós $\mathbf{w}$ vektor szerinti differenciálással a következőnek adódik:

$$\frac{\partial \varepsilon}{\partial \mathbf{w}} = -2\mathbf{p} + 2\mathbf{R}\mathbf{w} = 2(\mathbf{R}\mathbf{w} - \mathbf{p}) = \mathbf{0}, \tag{3.9}$$

melynek megoldása a Wiener-Hopf egyenlet, mely megadja az optimum helyét a $\mathbf{w}$ szűrőegyütthatók terében:

$$\mathbf{w}_{opt} = \mathbf{R}^{-1}\mathbf{p}. \tag{3.10}$$

Az optimális paraméterkészlet meghatározása ezek alapján szélsőérték-keresési feladat. Az eredmény nehézsége viszont az, hogy a $\mathbf{w}_{opt}$ együtthatókészlet számításához a modellezendő rendszer be- és kimeneti jeleinek statisztikai tulajdonságait *a priori* ismernünk kell a korrelációs vektorok és mátrixok meghatározásához.

A korrelációs vektorok ismeretének hiányában illetve azok részleges ismerete esetében a paraméterbeállítás csak iteratív módon végezhető. Ekkor a statisztikai paraméterek helyett a jelek pillanatnyi értékeinek felhasználásával törekszünk az optimális paraméterkészlet megközelítésére. Az iteráció egyes lépéseiben meg kell határozni, hogy az adott pontban mennyi a hibafelület gradiense, és a következő lépés paraméterkészletét ennek ismeretében kell kialakítani.

Egy lehetséges megközelítés a legmeredekebb lejtő módszere, amelyben a legmeredekebb irányba, a negatív gradiens mentén mozdulunk el az ún. bátorsági tényező által megszabott mértékben. Ennek alkalmazásával elkerülhetjük a bonyolult számítások elvégzését, valamint kiküszöbölhetjük a statisztikai adatok előzetes ismeretének hiányát is.

3.3. Adaptív jelfeldolgozó algoritmusok

3.3.1. Az LMS algoritmus

A statisztikai jellemzők megkövetelt ismeretét kiküszöbölhetjük, valamint a bonyolult számításokat is elkerülhetjük, ha a költségfüggvényben az átlagos hiba helyett csak a pillanatnyi hibát vesszük figyelembe, és a legmeredekebb lejtő módszerét eszerint módosítjuk. Így jár el az LMS[1] algoritmus a modellparaméterek adaptálásakor [1]. Az $\mathbf{R}$ autokorrelációs mátrix, valamint a $\mathbf{p}^T$ keresztkorrelációs vektor tehát a következőképpen módosul:

$$\mathbf{R} = \mathrm{E}\{\mathbf{x}(n)\mathbf{x}^T(n)\} \text{ helyett: } \hat{\mathbf{R}} = \mathbf{x}(n)\mathbf{x}^T(n), \tag{3.11}$$

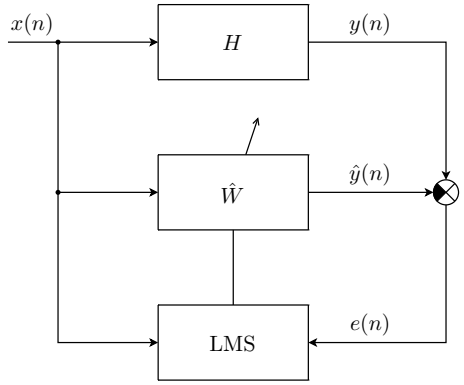
$$\mathbf{p}^T = \mathrm{E}\{y(n)\mathbf{x}^T(n)\} \text{ helyett: } \hat{\mathbf{p}}^T = y(n)\mathbf{x}^T(n). \tag{3.12}$$

A pillanatnyi hibán alapuló becslés miatt azonban a paraméterbeállítás módosítása pontatlan lesz, nem feltétlenül a negatív gradiens irányába történik. Hosszabb időintervallumban azonban az egyes módosítások pontatlansága kiátlagolódik. A módszer alkalmazásával viszont a modell a becsült rendszer parametrikus változásait jobban képes követni. Az LMS algoritmus sajátossága, hogy a hibaminimum körül a pillanatnyi hiba kicsi, a hibafelület gradiense nulla körül van. Így a para-

[1] Least Mean Squares

méterkészlet soha nem állandósul a minimális értéken, azt ugyanis nem is éri el, hanem annak egy kis környezetében lépésről-lépésre ugrál. Az LMS algoritmusban ezek alapján tehát a paramétertérben történő lépések mértékét, a *bátorsági tényezőt* célszerű kicsire megválasztani. Ezzel azonban az algoritmus beállási tulajdonságai romlanak. A μ bátorsági tényező optimális értéke az $\mathbf{R}$ autokorrelációs mátrix ismeretében meghatározható.

Az LMS algoritmust alkalmazó legegyszerűbb struktúra blokkvázlata a 3.1. ábrán látható.



3.1. ábra. Az LMS algoritmus blokkvázlata

Az LMS algoritmusban kiszámításra kerülő rekurzív egyenletek a hibára valamint az együtthatókra vonatkozóan a következőkre adódnak:

$$e(n) = y(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n), \tag{3.13}$$

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + 2\mu e(n)\mathbf{x}(n). \tag{3.14}$$

Az LMS algoritmus által használt modell, melynek paraméterei hangolásra kerülnek, egy véges impulzusválaszú (FIR) digitális szűrő, melynek impulzusválasza a $\mathbf{w}$ vektor. Az n-edik ütemben rendelkezésre álló szűrőegyüttható-készletet a $\hat{\mathbf{w}}(n)$ vektor adja meg. Az LMS algoritmus tehát az $\mathbf{R}$ autokorrelációs mátrix, valamint a $\mathbf{p}^T$ keresztkorrelációs vektor ismerete nélkül képes a $\mathbf{w}_{opt}$ együtthatókészletet megközelítő szűrőegyütthatókat előállítani.

3.3.2. Az NLMS algoritmus

Ha a stabilitási és beállási tulajdonságokat egyaránt megfelelően jobbra illetve gyorsabbra kívánjuk beállítani, nyilvánvaló, hogy a két szempont szerint ellentétes irányba kellene a bátorsági tényezőt hangolni. A problémára egy lehetséges megoldást kínál az NLMS[2] algoritmus. Az NLMS a bátorsági tényezőt a bemeneti jel alapján normálja, így jobb stabilitási tulajdonságokkal, adott feltételek mellett kb. kétszeresen gyorsabb beállással rendelkezik, valamint a lépések méretének változtatása miatt az LMS algoritmusnál tapasztalttól kisebb gradiens-zaj[3] jellemzi.

Az NLMS algoritmus rekurzív összefüggései a hibára valamint az együtthatókra vonatkozóan a következők:

$$e(n) = y(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n), \tag{3.15}$$

[2]Normalised LMS

[3]A pillanatnyi derivált a deriválthoz viszonyítva zajjal terhelt. $\frac{\partial\hat{\varepsilon}(n)}{\partial\mathbf{w}(n)} = \frac{\partial\varepsilon(n)}{\partial\mathbf{w}(n)} + \text{zaj}$. Ezt nevezzük gradiens-zajnak.

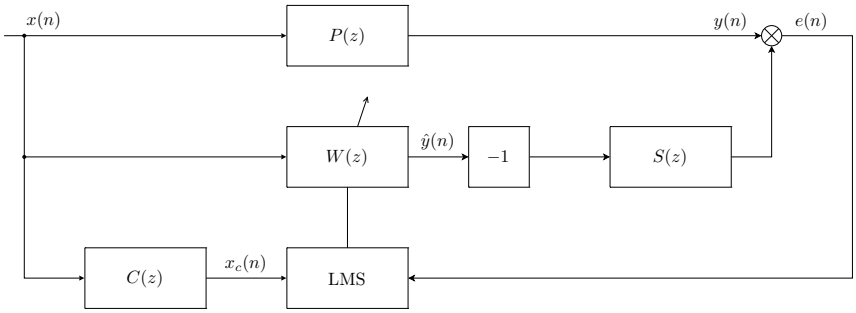
$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \frac{\tilde{\mu}}{a + \mathbf{x}^T\mathbf{x}} e(n)\mathbf{x}(n). \tag{3.16}$$

Az a konstans funkciója az, hogy kis bemenetek esetén se válhasson a tört nevezője nullához közelivé, és így a lépésköz túl naggyá.

3.3.3. Az FxLMS algoritmus

Egyes gyakorlati alkalmazásokban az LMS algoritmus segítségével adaptált paraméterkészletű szűrő kimenetére – a rendszer fizikai felépítéséből adódóan – egy másik, nem egységnyi átvitelű tag is kapcsolódik, és a különbségi jel képzésekor ennek kimenete kerül felhasználásra. Ez a szűrő a frekvenciatartománybeli szűrésen kívül fázistolást, késleltetést is visz a rendszerbe, amely könnyen instabilitást okozhat, ha a modellezendő fizikai rendszer késleltetése ettől kisebb vagy összességében nem elegendően nagy értékű. Az ily módon a beavatkozó jel útjába iktatott szűrőt a – főként az aktív zajcsökkentéssel foglalkozó – szakirodalomban „másodlagos útnak" nevezik (az „elsődleges út" a modellezendő rendszert jelenti [2]).

A probléma kiküszöbölésére több elvi megoldás is kínálkozik. Elméleti jelentőségű az az elgondolás, hogy a másodlagos átviteli úttal sorosan az inverzét elhelyezve annak hatása eliminálható. Ez a gyakorlatban nem feltétlenül valósítható meg, mert az átviteli függvény inverze nem biztos, hogy létezik. Egy másik megközelítés szerint a másodlagos átvitel becslőjét kell elhelyezni az LMS algoritmus bemenetén oly módon, hogy az a referenciajelet szűrje, de az adaptálandó szűrő bemenetére ne legyen hatással. Ez a felépítés tekinthető meg a 3.2. ábrán. A megoldás problémája viszont az, hogy ezt a másodlagos átvitelt valamilyen módon identifikálni kell, hogy a becslője a rendszerbe beépíthető legyen. A gyakorlatban alkalmazható megoldást az FxLMS[4] algoritmus által megvalósított struktúra kínálja.



3.2. ábra. Az FxLMS algoritmus blokkvázlata

Az FxLMS algoritmust alkalmazó struktúrában a hibajel a következőnek megfelelően áll elő:

$$e(n) = y(n) - \hat{y}_s(n), \tag{3.17}$$

ahol $\hat{y}_s(n)$ az $\hat{y}(n)$ jel $S(z)$ másodlagos átvitellel szűrt változata. Legyen $C(z)$ átvitelű blokk az S átvitel becslője, valamint $\mathbf{c}$ annak véges impulzusválasza (a vektor a FIR szűrőegyütthatókat tartalmazza)! Ekkor $e(n)$ felírható a következő módon:

$$e(n) = y(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}_c(n), \text{ ahol} \tag{3.18}$$

[4]Filtered-X LMS

$$\mathbf{x}_c(n) = \begin{bmatrix} \sum_{i=0}^{I-1} c_i x(n-i) \\ \sum_{i=0}^{I-1} c_i x(n-i-1) \\ \dots \\ \sum_{i=0}^{I-1} c_i x(n-i-(M-1)) \end{bmatrix}. \tag{3.19}$$

A $\mathbf{c} = [c_1, c_2, \dots, c_I]$ vektor a másodlagos átvitel becslőjének (véges) impulzusválasza, I pedig $\mathbf{c}$ hossza.

A szűrőegyütthatók adaptálására alkalmazható rekurzív egyenlet:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + 2\mu e(n)\mathbf{x}_c(n). \tag{3.20}$$

4. fejezet

Jelfeldolgozó processzorok

4.1. Áttekintés

A DSP-k[1], azaz a digitális jelfeldolgozó processzorok széles körben elterjedtek a beágyazott rendszerek területén. Megtalálhatók mobiltelefonokban, hang- és video, DVD lejátszókban, digitális fényképezőgépekben és kamerákban, orvosi műszerekben, valamint számos olyan ipari alkalmazásban, ahol jelekkel vagy adatfolyamokkal kell dolgozni.

A digitális jelfeldolgozó processzorok az analóg vagy digitális jelfolyamba iktatásuk révén valamilyen tényleges jelfeldolgozást valósítanak meg az egyszerű szűréstől a bonyolult jelfeldolgozó struktúrákig. A DSP-ket szabályozási rendszerekben is alkalmazzák, ahol a szabályzást végző algoritmus is a jelfeldolgozó processzoron kerül megvalósításra [16].

A jelfeldolgozó processzorok rendszerint az általános célú mikroprocesszorok minden funkcióját tartalmazzák, viszont számos tekintetben fejlettebbek azoknál. Az általános célú processzorokkal szemben a DSP-k valós idejű jelfeldolgozási alkalmazásokra kerülnek optimalizálásra mind architekturális szinten, mind az utasításkészletük kialakítása terén. Úgy alakítják ki őket, hogy a hatékony jelfeldolgozást, a leggyakrabban előforduló műveletek elvégzését a lehető legkevesebb utasítás végrehajtása mellett, azaz a leggyorsabban lehessen használatukkal elvégezni [10]. A DSP-k fejlett aritmetikai egységgel rendelkeznek, amelyből rendszerint több darabot is implementálnak egy processzoron belül.

4.2. Alapvető ismeretek a jelfeldolgozó processzorokról

A kereskedelemben kapható DSP-k néhány jellemző szolgáltatását illetve szokványos architekturális kialakításának tulajdonságait a következőkben foglaltam össze [15].

4.2.1. Architektúra

A DSP-k az általános célú processzorokkal ellentétben Harvard architektúra szerint készülnek, azaz a program- és adatmemória fizikailag is különválik bennük [11]. Nagy mértékben párhuzamosított architektúrájuk lehetővé teszi a párhuzamos műveletvégzést és memóriahozzáférést, ugyanis ezekre a célokra külön-külön buszrendszereket készítettek. Az elkülönített buszrendszer révén fizikailag

[1] Digital Signal Processor

egyidejűleg lehet hozzáférni az utasításkódhoz és az operandusokhoz egyaránt. A DSP-k rendszerint több különálló aritmetikai egységet is tartalmaznak. A jelfeldolgozó processzorokkal tehát – bizonyos megszorítások betartásával – lehetőség van egy órajelciklusban belül két memória- és két aritmetikai művelet elvégzésére. A következőkben ismertetésre kerülő MAC műveletvégzéssel kombinálva ennek alapján például egy FIR szűréshez szükséges órajelciklusok száma – kis többlettel – megegyezik a szűrő impulzusválaszának hosszával.

A jelfeldolgozó processzorok hardveresen támogatják a MAC[2] műveletvégzést, melyre mátrixműveleteknél van nagy szükség. Használatával egy szorzás oly módon végezhető el, hogy eredménye automatikusan hozzáadódik egy ily módon szorzásonként kumulálódó összeghez. Pontosabban fogalmazva, egy ciklus alatt a processzor nagy pontossággal összeszoroz két számot, hozzáadja a kumulálódó összeghez, illetve beolvassa a memóriából a következő szorzás elvégzéséhez szükséges két tényezőt. Ily módon egy ciklusban végrehajtható egy nagy pontosságú szorzás és a szükséges összeadás. A MAC műveletvégzés felhasználására a legegyszerűbb gyakorlati példa a konvolúciószámítás digitális szűréshez.

A DSP-k különálló belső programmemóriával, és nem ritkán több különálló adatmemóriával is rendelkeznek a párhuzamos architektúra által biztosított lehetőségek kihasználhatósága biztosításának érdekében. A belső memória mérete blokkonként megabites nagyságrendű, amely kielégítő a legtöbb alkalmazás megvalósításához.

4.2.2. Aritmetika

A jelfeldolgozó processzorok fejlett hardveres műveletvégző egységekkel rendelkeznek, így számos művelet végrehajtásához egyetlen utasításciklusra van szükség. Ilyen például az összeadás, (kivonás), szorzás, logikai műveletek, vagy a bitléptetés. Az aritmetika legfontosabb jellemzője a szóhossz, azaz az a mérőszám, hogy az aritmetika milyen hosszú operandusokkal dolgozik, és az eredmény milyen hosszúságban áll elő. Ezen kívül a számábrázolás tekintetében megkülönböztetünk fixpontos processzorokat, de néhány processzorcsalád a lebegőpontos számábrázolás és műveletvégzés lehetőségét is felkínálja. Ekkor az exponens hosszának alkalmas változtatásával kis- és nagy számok esetében egyaránt azonos pontosságú ábrázolás biztosítható, nagy dinamikatartomány érhető el. A lebegőpontos processzorokban lehetőség van fixpontos számábrázolás használatára is. A legtöbb jelfeldolgozó processzor azért mégis fixpontos számábrázolást implementál, mert a valóságban nincs szükség a lebegőpontos számábrázolás által biztosított dinamikatartományra, és az egyszerűbb hardverfelépítés a sebesség javára és az alacsonyabb ár lehetővé tételére fordítható.

4.2.3. Címzési módok

Az általános célú processzoroktól semmiben sem elmaradva, a jelfeldolgozó processzorokban is lehetőség van direkt és indirekt címzés használatára egyaránt. A DSP-k fejlett címaritmetikája olyan indirekt címzési módokat kínál, melyek szükségesek a párhuzamos aritmetika teljesítőképességének maximális kihasználásához. Az indirekt címzési mód néhány, jellemzően 8-16 db címregiszter használatával érhető el, melyek értékét a címaritmetika automatikusan módosítja.

A módosítás lehet egyszerű növelés vagy csökkentés, vagy a címregiszterekhez rendelt módosító- vagy más elnevezés szerint ofszetregiszterek tartalmával való módosítás. A jelfeldolgozó processzorok jellemzője ezeken kívül a modulo-címzés hardveres támogatása, amely lehetővé teszi ún. cirkuláris bufferek kezelését anélkül, hogy a címet szoftveresen kellene kiszámítani. Ehhez a címregisz-

[2] Multiply and ACcumulate

terekhez rendelt különálló regiszterben kell letárolni a cirkuláris buffer kívánt hosszát. A modulocímaritmetika lehetővé teszi a lineárisan érkező adatfolyamból egy csúszóablakkal kiragadott rész kezelését, így hatékonyan támogatja többek között a gyakori konvolúció és korrelációszámítást. Ekkor ugyanis maga a címaritmetika gondoskodik a cirkularitás biztosításáról szoftveres kiegészítés nélkül is.

A jelfeldolgozó processzorok fejlett címaritmetikája támogatja az ún. bitfordított (bitreverse) címzési módot, melyre az FFT algoritmusok implementálásakor van szükség.

4.2.4. Programszervezés

A jelfeldolgozó processzorok hardveres ciklusszervezésre is lehetőséget biztosítanak, amely funkció használatával nincs szükség a ciklusváltozó szoftveres vizsgálatára és értékének karbantartására a ciklus futása során. A hardveres ciklusszervezés használatához csak a ciklus elejét és végét kell jelölni, valamint meg kell adni a lépések számát [17]. A ciklus iterációiban ekkor valóban csak a ciklusmag utasításainak végrehajtására van szükség.

A jelfeldolgozó programok legtöbbször egy úgynevezett főprogram futtatását végzik, amely nem végez tényleges műveleteket, csak egy üres ciklust hajt végre folyamatosan. Az egyes mintavételi ciklusokban, amikor az A/D átalakító kimenetén érvényes adat érhető el, megszakításkérés érkezik. Ekkor a főprogram felfüggesztésre kerül és a végrehajtás a kérés kiszolgálásával folytatódik. A kiszolgálást be kell fejezni a következő kérés megérkezéséig a helyes működés biztosításának érdekében.

4.2.5. Be- és kimenetek

A jelfeldolgozó processzorok gyors soros portokkal rendelkeznek, melyek – az általános célú processzorokban megszokottal ellentétben – akár a processzor órajelének sebességével is képesek működni. Ez a megoldás igen gyors adatáramlást tesz lehetővé a perifériák és a processzor között.

5. fejezet

Gyakorlati alkalmazások

A diplomaterv részét képezi az a feladat, hogy a mérés során ismertetésre kerülő adaptív jelfeldolgozó algoritmusok egy részét a gyakorlatban is ki lehessen próbálni a mérés folyamán. Ennek teljesítésére feladatom részét képezi a vonatkozó algoritmusok gyakorlati alkalmazásainak áttekintése, majd annak mérlegelése, hogy melyeket célszerű a mérés során a hallgatóknak bemutatni, valamint melyeket érdemes a hallgatók által üzemeltetni.

Egyrészt súlyt kell fektetni a gyakorlatban fontos alkalmazások bemutatására, másrészt időt kell fordítani a gyakorlatban kevésbé fontos, de az elméleti ismeretek elmélyítését nagy mértékben elősegítő struktúrák ismertetésére egyaránt.

Általánosságban elmondható, hogy az adaptív jelfeldolgozás fontosabb gyakorlati alkalmazásai az akusztikai területhez köthetők. Az egyik legelterjedtebb alkalmazás az aktív zajcsökkentés példája, amelyben a külvilágból érkező zajt a beavatkozó hangszórókon kiadott ellenfázisú zaj segítségével a hibamikrofonok bizonyos környezetében igyekszünk kioltani [4]. Az aktív zajcsökkentő rendszerekben többféle struktúrát meg lehet valósítani. A jó stabilitással és jelentősebb zajelnyomással jellemezhető struktúrák mindegyikében szükség van azonban a beavatkozó hangszórók és a mikrofonok közötti akusztikai átvitel identifikációjára, amelyre egy lehetséges módszer az LMS algoritmus alkalmazása (rendszeridentifikáció).

Hasonló gyakorlati alkalmazás a mérőhidak kiegyenlítésének problémaköre. Ekkor a kalibrálatlanság következtében nyugalmi állapotban is kiegyenlítetlen a híd, kimeneti feszültsége nem zérus, hanem a gerjesztő feszültséggel arányosan változik. Ha a kimenetből ezt az ofszetet alkalmas módon kivonjuk, a mérőerősítőnek kisebb dinamikatartományt kell átfognia, így pontosabb mérést tesz lehetővé.

Egy másik gyakorlati alkalmazás a visszhangcsökkentés problémaköre, melyben egy hosszú átviteli vonalon a helytelen lezárás következtében reflektálódó jel kioltását tűzzük ki célul az adó oldalán. A visszhang felléphet a vevőoldali reflexión kívül annak következtében is, hogy a vevőoldali hangszóró és a mikrofon között áthallás tapasztalható.

Strukturálisan egyszerű, de az LMS algoritmus működésének megértését nagyban megkönnyítő struktúra az adaptív vonaljavító, amely egy villamos jelből képes elnyomni a szélessávú zajkomponenst a keskenysávú hasznos jel javára.

A következő pontokban ezeket a struktúrákat tekintjük át részletesebben.

5.1. Rendszeridentifikáció – eljárások az átviteli függvény meghatározására

Ahogy az 5. fejezet elején említettük, számos gyakorlati alkalmazásban szükség van valamely fizikai átviteli út identifikációjának elvégzésére. A rendszeridentifikáció során egy ismeretlen rendszer rendelkezésre álló bemeneti és kimeneti jelei alapján meghatározásra kerül az általa realizált átviteli függvény (természetesen ehhez a rendszer linearitását feltételezzük). A témakör elméleti alapjait a 3.2. fejezet ismerteti részletesen.

Az átviteli függvény mérésére számos paraméterbecslési modell és megközelítés is kínálkozik, melyek mindegyike természetes módon bizonyos előnyökkel és hátrányokkal rendelkezik. Fontos mérlegelendő szempont például az, hogy a meghatározott paraméterkészlet milyen jól illeszkedik ahhoz a rendszerhez, amelyben később felhasználásra fog kerülni. A módszerek egy csoportja adaptív jelfeldolgozó algoritmust alkalmaz a feladat megoldásának érdekében. A következőkben áttekintést adunk a rendszeridentifikációra alkalmazható módszerekről a teljesség igénye nélkül.

Az átvitel meghatározásának céljából lehetőség van a vizsgálni kívánt rendszer bemenetét a frekvenciatartományban léptetett szinuszos jellel gerjeszteni, és a kimenet amplitúdóját minden egyes frekvencián állandósult állapotban megmérni. Ebben az esetben, mivel a gerjesztés keskeny sávú, a mérés varianciája kicsi, az eredmény nagy pontosságú lehet, de a teljes mérés-sorozat hosszú időt vesz igénybe. Ezt az elvet alkalmazó egy lehetséges gyakorlati alkalmazás lehet a rezonátorokból álló rendszeridentifikációs struktúra [21].

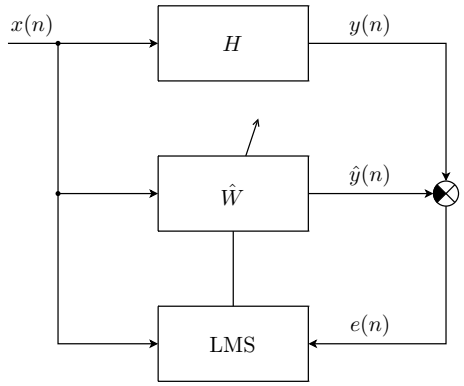
Egy másik lehetséges megoldás szerint a rendszer bemenetét egy arra alkalmasan kiválasztott és előállított jellel gerjesztve a kimenet FFT segítségével kiértékelhető. Ekkor a gerjesztés spektrumát előzetesen ismertnek kell beállítani, erre alkalmas például a multiszinusz vagy fehér zaj használata. Ebben az esetben azonban a gerjesztőjel teljesítménye a frekvenciatartományban jelentősen eloszlik, egy frekvenciára kis teljesítmény esik. A mérés varianciája tehát nagy, és az is előfordulhat, hogy azokon a frekvenciákon, ahol az átvitel nagy csillapítású, nem gerjeszti eléggé a rendszer bemenetét. Ekkora a külső zajok nagy mértékben befolyásolják a mérési eredményt. Problémát okozhat még az átvitel egyes, analóg áramkörökkel megvalósított részeinek intermodulációs torzítása is. Az FFT alkalmazásával történő kiértékelés azonban meglehetősen gyors mérést tesz lehetővé. A multiszinuszos jel további hátránya, hogy előállítása nehéz, ugyanis az azonos fázishelyzetbe kerülő komponensek miatt nagy amplitúdók alakulhatnak ki, melyek telítésbe vihetik a rendszer D/A és A/D átalakító átviteli elemeit [22].

Egy harmadik megközelítés szerint a rendszert fehér zajjal gerjesztve és a be- és kimeneti jeleket megfigyelve egy adaptív szűrő együtthatóit hangolva állítjuk elő az identifikált átvitel becslőjét. Ebben az esetben a megfigyelt, végtelen impulzusválasszal jellemezhető átvitel impulzusválaszát kíséreljük megmérni, melyet csonkolt hosszban tárolunk. A mérés gyors és pontos is lehet megfelelő beállítások alkalmazása esetén. A módszer jellegzetessége, hogy a modell, az adaptív szűrő számos tekintetben különbözik az identifikálandó rendszertől, azonban ezek a különbözőségek a gyakorlati alkalmazásokban legtöbbször nem okoznak problémát. A különbségek a következő okokból fakadnak [2]:

- IIR rendszer FIR rendszerrel történő modellezése.
- A gerjesztés nem minden frekvencián vezérli ki megfelelően a bemenetet.
- Az optimális együtthatókészlet nem érhető el, csak közelíthető.

A gerjesztőjel fehérségének biztosítása azért fontos, mert az adaptív szűrő csak azon frekvenciákon hangolódik, ahol a rendszer gerjesztve van. A gyakorlatban ezen feltétel helyett azt kell teljesíteni, hogy a rendszer sávszélességében egyenletes legyen a gerjesztés spektruma [22].

A 3.3.1. pontban ismertetett LMS algoritmuson alapuló rendszeridentifikációt megvalósító struktúra az 5.1. ábrán látható. H az ismeretlen rendszer átvitele, $x(n)$ és $y(n)$ a be- és kimenet mintavett értékeinek sorozata. $\hat{W}$ a digitálisan realizált adaptív szűrő, mely az $\hat{y}(n)$ becsült kimenetét oly módon állítja elő, hogy az $e(n)$ hibajel minimális legyen. Ennek érdekében együtthatóit az LMS algoritmus hangolja $x(n)$ és $e(n)$ ismeretében.



5.1. ábra. A rendszeridentifikációt megvalósító struktúra

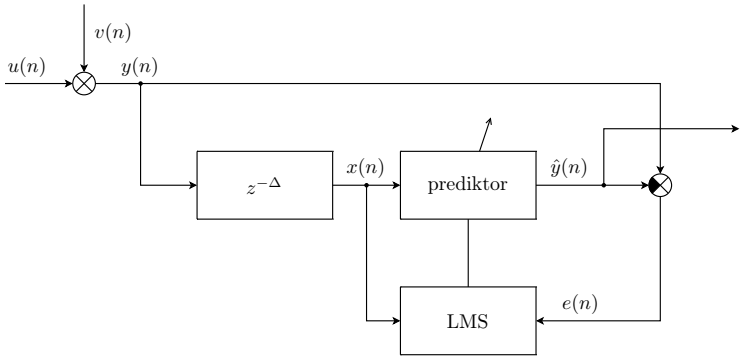
5.2. Adaptív vonaljavító

Az adaptív digitális jelfeldolgozás egyik fő területe a különféle zajok elnyomását tűzi ki célul. Egyik lehetséges felmerülő gyakori probléma például az információt hordozó jel kiszűrése az azt terhelő additív zajból. Az adaptív vonaljavító, azaz ALE[1] egy ilyen feladatot megvalósító struktúra [13], [14]. Alkalmazására olyan esetekben lehet szükség, amikor a keskeny sávú hasznos jel sávszélessége és egyéb paraméterei időben változnak, míg kevés előzetes, *a priori* ismeret áll rendelkezésünkre róluk. Ilyen alkalmazási terület lehet például a szonár, bizonyos orvosi alkalmazások valamint különféle beszéddel kapcsolatos feldolgozások.

Az adaptív vonaljavító egy olyan speciális zajelnyomó struktúra, amely lehetővé teszi a bemenetét terhelő szélessávú zaj megfelelő mértékű csillapítását, elnyomását, míg a keskeny sávú, hasznos jelet hordozó bemeneti komponenst ideális esetben egységnyi erősítéssel, de legalábbis kis csillapítással továbbengedi. A számunkra hasznos jellel kapcsolatban az ALE működéséhez szükséges egyetlen teljesítendő feltétel, hogy a jel megfelelően keskeny sávszélességű legyen. Az adaptív jelfeldolgozás miatt nem szükséges, hogy a jel stacionárius tulajdonságú legyen.

Az ALE struktúráját tekintve egy késleltető valamint egy lineáris prediktor összekapcsolásából épül fel, amint az az 5.2. ábrán látható. Az ALE $y(n)$ bemeneti jele a keskenysávú, hasznos $u(n)$ jelből, valamint az ahhoz adódó $v(n)$ szélessávú zajból áll. A prediktor $\hat{y}(n)$ kimeneti jelét a $y(n)$ bemeneti jelből kivonva megkapjuk az $e(n)$ becslési hibát, melyet a prediktor adaptív hangolására használunk fel. A lineáris prediktort esetünkben egy FIR szűrő valósítja meg, melynek együtthatóit az LMS algoritmus hangolja.

[1] Adaptive Line Enhancer



5.2. ábra. Az ALE struktúra blokkvázlata

A prediktor $x(n)$ bemeneti jele a ALE bemenetének Δ számú mintával késleltetett változata:

$$x(n) = y(n - \Delta). \tag{5.1}$$

A struktúra Δ késleltetésének mértékét oly módon kell az implementáció során megválasztani, hogy az $y(n)$ bemeneti jel $v(n)$ zajkomponense, valamint annak $v(n-\Delta) \ldots v(n-\Delta-M)$ késleltetett változata korrelálatlan legyen (M a FIR szűrő hosszát jelöli). Ily módon a becslési hiba képzésekor a bemenet zajkomponense, valamint az adaptív szűrő kimenetének a bejövő zajból származó komponense korrelálatlan lesz. Ennek hatására a szűrő hangolásakor nem fordul elő az, hogy az adaptív algoritmus a szűrőegyütthatókat olyan irányba állítja, hogy a szűrő a zajt is átengedje annak érdekében, hogy a különbségképzéskor a becslési hibát még jobban minimalizálhassa. Mivel a struktúra kimeneti jele az $\hat{y}(n)$ jel, az ALE működése akkor jó, ha ez minél kevesebb zajt tartalmaz. Ha tehát a Δ ún. dekorreláló késleltetés értéke megfelelő, lehetővé válik a zaj kioltása, ugyanis azt a szűrő nem engedi át, miközben az időben változó, hasznos jelösszetevőt nem próbálja meg elnyomni.

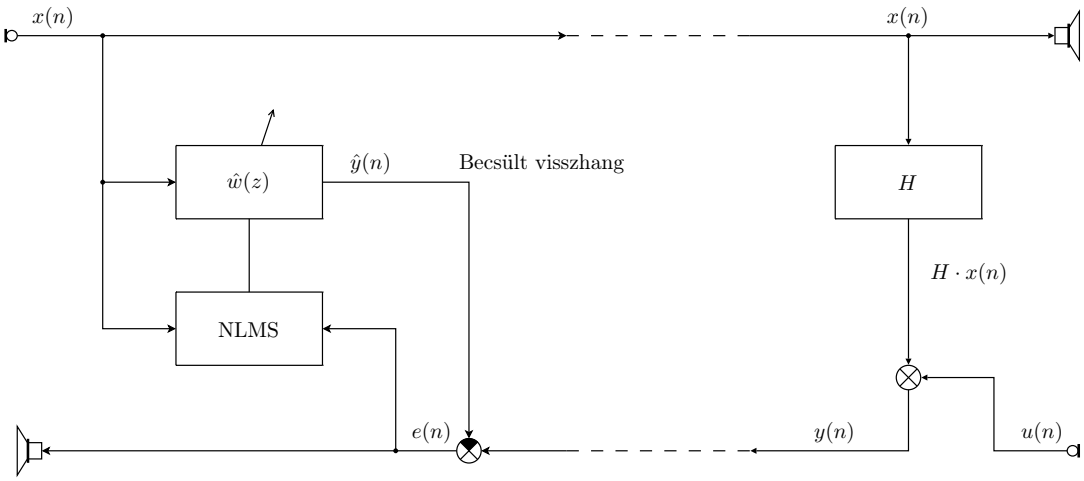
5.3. Adaptív visszhangcsökkentés

A visszhang problémája legszélesebb körben elsőként a PSTN[2] telefonhálózatokban jelentkezett [26]. A kapcsolóközpontokban elhelyezett ún. hibridek közötti nagy távolság és a tökéletlen illesztés miatt reflexió lép fel, amely a nagy késleltetés miatt hallható visszhangot képes okozni, amely bizonyos határ fölött zavarja a beszélőket, vagy akár a beszédérthetőséget is jelentősen ronthatja [23], [24].

Az adaptív visszhangcsökkentő struktúra modellje az 5.3. ábrán látható. Az ábra bal oldalán látható hangszóró-mikrofon pár az egyik, míg a jobb oldalon látható pár a másik személyhez tartozó végpont. A szaggatott vonalak a hosszú átviteli utat szemléltetik, de a visszhang szempontjából az ezek által okozott késleltetést beleértjük a H átvitelbe, amely a hallgató oldalán elhelyezkedő hibrid átvitelét jelöli.

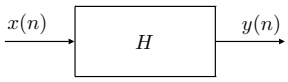
A beszélő akkor hallja meg saját visszhangját, ha $H \neq 0$. Tekintsük azt az esetet, amikor a hallgató csendben van (ekkor $u(n) = 0$). Ebben az esetben a hallgató oldaláról $y(n) = H \cdot x(n)$ visszhang indul vissza a beszélő felé. Az 5.4. ábrán az 5.3. ábra jobb oldalát láthatjuk erre az esetre vonatkozóan újrarajzolva. Ebből már látható, hogy az adaptív visszhangcsökkentés nem más, mint

[2]Public Switched Telephone Network



5.3. ábra. Az adaptív visszhangcsökkentő struktúra blokkvázlata

a hallgató oldalán elhelyezkedő hibrid átvitelének rendszeridentifikációs feladata.



5.4. ábra. Ha $u(n) = 0$, a hallgatóoldali hibridet identifikáljuk

Ha a visszhangot alkalmas módon becsülni tudjuk, az $\hat{y}(n)$ becsült visszhangjelet kivonhatjuk a beszélő oldalára visszaérkező $y(n)$ jelből. A két jel különbsége a becslési hibajel: $e(n)$. A becslés az $e(n)$ hibajelet és a beszélő $x(n)$ jelét felhasználó, a 3.3.2. pontban ismertetett NLMS algoritmussal történik.

Az adaptációt nehezíti, ha a hallgató a beszélővel egyidejűleg beszél a telefonba, azaz ha $u(n) \neq 0$. Az ilyen esetekben jelentkező problémák kiküszöbölésére az adaptációt az „összebeszélés" idejére fel kell függeszteni.

5.4. Aktív zajcsökkentés

Az aktív zajcsökkentés feladata a környezetből, vagy valamilyen zajforrásból érkező zajnak ellenfázisú zajjal történő kioltása a tér adott kiválasztott környezetében. A passzív hanggátló struktúrákkal szemben a kitűzött cél elérésének érdekében arra alkalmas helyeken ún. beavatkozó hangszórók elhelyezésére van szükség. A beavatkozó hangszórók segítségével kiadott ellenfázisú zajjal a hibamikrofonok bizonyos környezetében csendes zónák alakíthatók ki. Az akusztikai rendszerek nagy dinamikatartományban jó közelítéssel lineárisnak tekinthetők, így a szuperpozíció elve eredményesen alkalmazható [4].

Az aktív zajcsökkentő rendszereknek akkor van létjogosultsága, ha a kioltandó zaj alacsony frekvenciájú, és a passzív hanggátlás már kis hatékonyságú volna, vagy abban az esetben, ha a mechanikai struktúra nem teszi lehetővé passzív hanggátlás költséghatékony telepítését [5]. Ez utóbbi eset vonatkozik például a csövekben terjedő zavarokra, amely probléma a mérés során konkrétan is bemutatásra kerül.

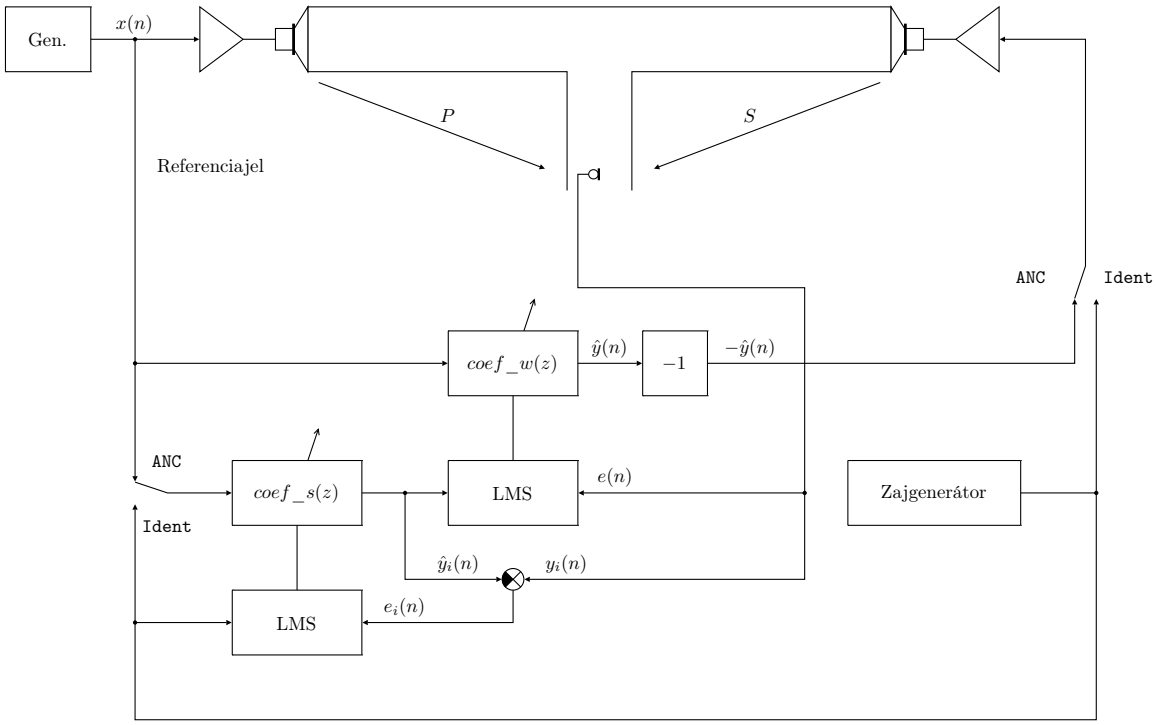
Az aktív zajcsökkentés az akusztikai alkalmazásokon kívül mechanikai rezgések csillapítására is eredményesen alkalmazható, a megfelelő szenzorok és beavatkozók ebben az esetben is rendelkezésre

állnak (például gyorsulásérzékelők és rázóasztalok) [6].

Az aktív zajcsökkentő rendszerekben többféle struktúrát meg lehet valósítani. A jó stabilitási tulajdonságokkal és jelentősebb elnyomással jellemezhető struktúrák mindegyikében szükség van azonban a beavatkozó hangszórók és a mikrofonok közötti akusztikai átvitel identifikációjára, amelyre egy lehetséges módszer az LMS algoritmus alkalmazása (rendszeridentifikáció, ld. 5.1. pont).

Az adaptív jelfeldolgozó algoritmust alkalmazó aktív zajcsökkentő struktúrák közül az FxLMS algoritmuson alapulót mutatjuk be. Az FxLMS algoritmus a 3.3.3. pontban került ismertetésre.

Az 5.5. ábrán egy komplex FxLMS algoritmust alkalmazó aktív zajcsökkentő struktúra blokkvázlata látható, amely csövekben terjedő zaj kioltását végzi [25]. A cső egyik végén található a zajforrás, melyet egy hangszóró modellez, míg a másik oldalán a beavatkozó hangszóró került elhelyezésre. A cső leágazásában található mérőmikrofon jele a cső szája által lesugárzott zajjal arányos. A struktúra a gyakorlatban jól alkalmazható ipari klímaberendezések zajának csökkentésére.



5.5. ábra. Az FxLMS algoritmus megvalósítása aktív zajcsökkentő rendszerben

A zajforrástól a hibamikrofonig terjedő P átvitel az elsődleges út (*primary path*), míg a beavatkozó hangszórótól a hibamikrofonig terjedő S átvitel a másodlagos út (*secondary path*), a szakirodalomban alkalmazott terminológiát követve. Aktív zajcsökkentés üzemmódban a kapcsolókat `ANC`, míg a másodlagos út identifikációja során `Ident` állásban kell az ábrán tekinteni.

Példánkban az elnyomandó zajt egy generátor állítja elő, melyet azonnal felhasználunk az $x(n)$ referenciajel formájában. Az $x(n)$ jel a $coef\_w(z)$ adaptív szűrőre kerül, melynek $\hat{y}(n)$ kimenete – fázisfordítás után – a beavatkozó hangszóróra kerül. A fázisfordításra azért van szükség, mert az akusztikus térben összegzés valósul meg. A félreértések elkerülése érdekében már a 3.2. ábrát is ennek megfelelően alakítottuk ki.

A $coef\_w(z)$ szűrőt olyan adaptív algoritmus hangolja, mely hozzáfér a hibamikrofon $e(n)$ jeléhez, valamint az $x(n)$ referenciajel $coef\_s(z)$ átvitellel szűrt változatához. A $coef\_s(z)$ adaptív

szűrőben a zajcsökkentéskor az előzetesen identifikált S átvitel becslőjének megfelelő együtthatókészlet található.

A másodlagos átviteli út identifikálása során a $coef\_s(z)$ szűrő együtthatóit kell az S átvitel fehér zajjal történő gerjesztése és a mikrofon jelének mérése alapján hangolni. A kapcsolók `Ident` állásában a zajgenerátor jele a beavatkozó hangszóróra, valamint a $coef\_s(z)$ szűrő bemenetére, és az őt hangoló LMS algoritmus referenciajel-bemenetére kerül. Az $e_i(n)$ hibajel a mikrofon $y_i(n)$ jelének, valamint a $coef\_s(z)$ szűrő $\hat{y}_i(n)$ kimeneti jelének különbségeként áll elő:

$$e_i(n) = y_i(n) - \hat{y}_i(n). \tag{5.2}$$

A zajcsökkentési és identifikációs módokban egyes jelekre – az ábrán is látható módon – kétféle jelölést is alkalmaztam, amely az algoritmusok működésének megértését segíti elő.

5.5. Mérőhidak adaptív nullkompenzálása

Az aktív zajcsökkentéshez nagyon hasonló, szintén az FxLMS algoritmust alkalmazó struktúra használható mérőhidak kiegyenlítésére is.

A gyakorlatban nagyon sok olyan szenzort kínálnak a gyártók, melyek úgynevezett félhidas felépítésűek. Mechanikai kiképzésükből adódóan ugyanis két, ellentétes irányban változó impedanciát tartalmaznak. A félhíd két, a szenzoron kívül elhelyezett rezisztív elemmel teljes híddá egészíthető ki. A hidat az előírt váltakozó feszültséggel gerjesztve a kimeneti feszültség az impedanciák megváltozásával arányosan változik.

Ilyen szenzor például a pozíciómérésre alkalmas, két tekercsből és egy mozgó vasmagból álló LVDT[3]. A vasmag középhelyzetében az LVDT alkalmazásával kialakított híd kimeneti feszültsége nulla. A valóságban azonban a mérendő mechanikai szerkezet alapállapotában sem középen helyezkedik el a vasmag, így a híd kiegyenlítetlen. Erre a problémára nyújt megoldást az 5.6. ábrán bemutatott struktúra.

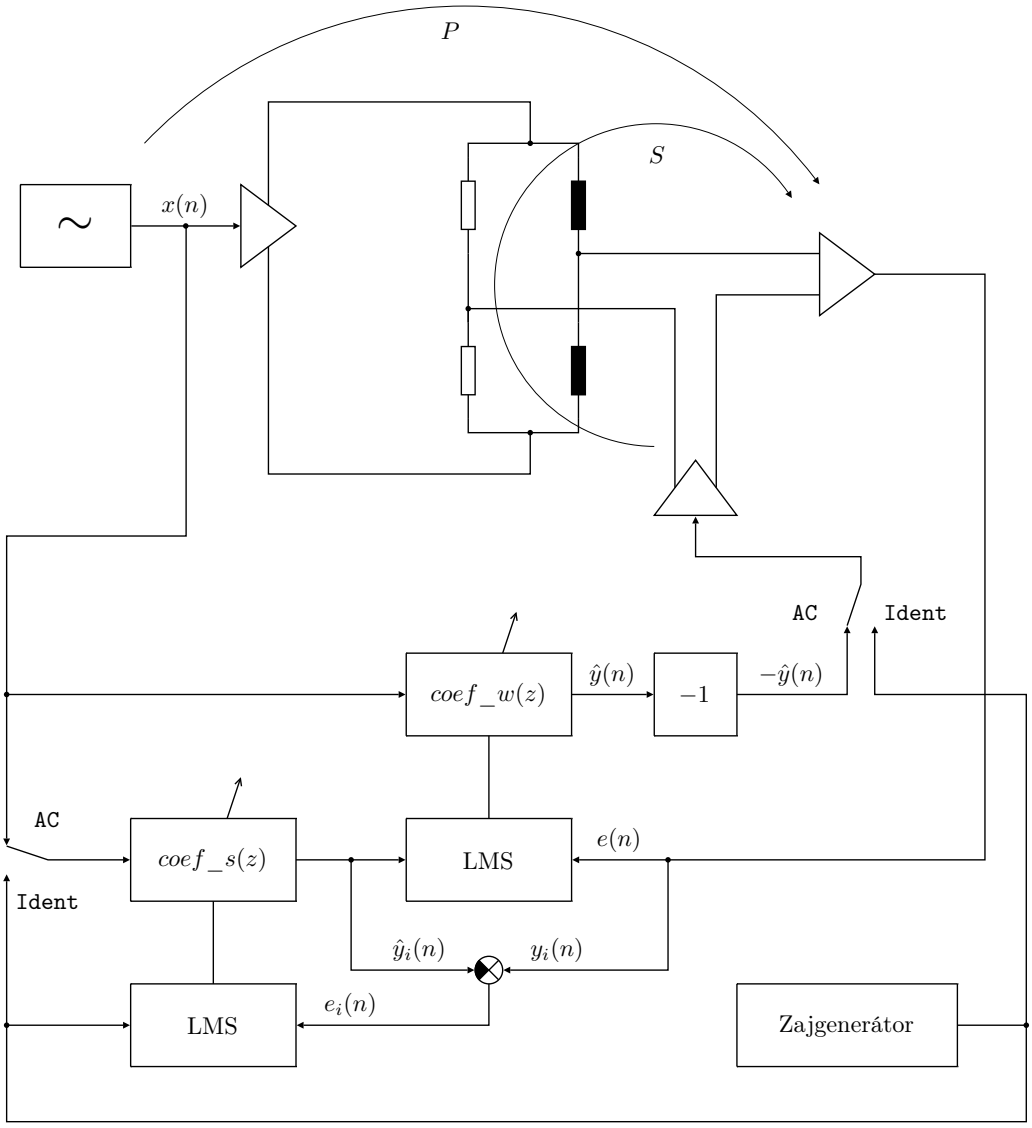
A mérőhidat egy szinuszgenerátor gerjeszti, melynek $x(n)$ jele, mint referenciajel a többi blokk számára közvetlenül hozzáférhető. A híd kimeneti feszültségéhez hozzáadódik a $-\hat{y}(n)$ beavatkozó jel értéke. A gerjesztéstől a mérőerősítőig terjedő P átvitel az elsődleges út (*primary path*), míg a beavatkozó erősítőtől a mérőerősítőig terjedő S átvitel a másodlagos út (*secondary path*) az aktív zajcsökkentő rendszerek elnevezéseinek mintájára. Aktív kompenzálás üzemmódban a kapcsolókat `AC`, míg a másodlagos út identifikációja során `Ident` állásban kell tekinteni.

A blokkvázlat további elemei megegyeznek az 5.5. ábrán az aktív zajcsökkentő rendszerben látottakkal.

Az $x(n)$ referenciajel a $coef\_w(z)$ adaptív szűrőre kerül, melynek $\hat{y}(n)$ kimenete – fázisfordítás után – a beavatkozó erősítőre kerül. A $coef\_w(z)$ adaptív szűrőt olyan adaptív algoritmus hangolja, mely hozzáfér a mérőerősítő $e(n)$ jeléhez, valamint az $x(n)$ referenciajel $coef\_s(z)$ átvitellel szűrt változatához. A $coef\_s(z)$ adaptív szűrőben az aktív kiegyenlítéskor az előzetesen identifikált S átvitel becslőjének megfelelő együtthatókészlet található.

A másodlagos átviteli út identifikálása során a $coef\_s(z)$ szűrő együtthatóit kell az S átvitel fehér zajjal történő gerjesztése és a mérőerősítő jelének mérése alapján hangolni. A kapcsolók `Ident` állásában a zajgenerátor jele a beavatkozó erősítőre, valamint a $coef\_s(z)$ szűrő bemenetére, és

[3]Linear Variable Differential Transformer



5.6. ábra. Az FxLMS algoritmus megvalósítása hídkapcsolás nullkiegyenlítésére

az őt hangoló LMS algoritmus referenciajel-bemenetére kerül. Az $e_i(n)$ hibajel a $coef\_s(z)$ szűrő $\hat{y}_i(n)$ kimeneti jelének, valamint a mérőerősítő $y_i(n)$ jelének különbségeként áll elő.

A híd ilyen módon történő kiegyenlítésére azért van szükség még a mérőerősítő előtt, mert ily módon a mérőerősítőre jutó jel sokkal kisebb dinamikatartományt fog át, mintha a kiegyenlítetlenségből származó ofszettel el lenne tolva, így a mérés során sokkal nagyobb pontosság érhető el.

6. fejezet

A megvalósításban alkalmazott hardvereszközök

6.1. Az ADSP-21364 EZ-KIT Lite fejlesztőkártya

Az ADSP-21364 EZ-KIT Lite fejlesztőkártya [8] az ADSP-21364 jelfeldolgozó processzorral [9] történő fejlesztés hatékony és gyors eszköze. A fejlesztőkártyán kialakításra került a jelfeldolgozó processzor minden olyan szükséges perifériája és környezete, amelyre a leggyakoribb feladatok elvégzésekor szükség lehet. Az analóg és digitális hang ki- és bemenetek, LED-ek, kapcsolók, nyomógombok azonnal hozzáférhetőek, így a szoftverfejlesztés mindenféle hardvertervezési lépés nélkül azonnal elkezdhető. A fejlesztőkártya tervezésekor igyekeztek a lehető legtöbb felhasználási lehetőséget kipróbálhatóvá, a processzor által nyújtott funkcionalitást pedig a lehető legmélyebben megismerhetővé tenni. A kártya a következő hardverelemeket bocsátja a fejlesztő rendelkezésére:

- Analog Devices ADSP-21364 processzor [9]
- 512 kbit × 8 bit SRAM[1]
- 1 Mbit × 8 bit flash memória
- 2 Mbit SPI[2] által kezelhető flash memória
- Analóg hanginterfész (AD1835A kodek, 4 db sztereó bemenet, 1 db sztereó kimenet)
- Digitális hanginterfész (1 db bemenet, 1 db kimenet)
- 11 db LED (ebből 1 db „power”, 1 db „board reset”, 1 db „USB monitor”, 8 db általános célú)
- 5 db nyomógomb (1 db „reset”, 2 db DAI lábra, 2 db FLAG lábra kötve)
- Bővítőinterfész (párhuzamos port, FLAG-ek, DAI, SPI)
- JTAG emulátor port
- USB port a PC csatlakoztatásához.

[1] Synchronous Random Access Memory
[2] Serial Peripheral Interface

6.2. Az ADSP-21364 lebegőpontos jelfeldolgozó processzor

6.2.1. Architekturális felépítés

A 6.1. pontban bemutatott fejlesztőkártya központi eleme az ADSP-21364 lebegőpontos jelfeldolgozó processzor [9], amely a SHARC[3] család tagja [10]. Ennek megfelelően felépítése a beágyazott rendszerekben elterjedt processzorok körében gyakori Hardvard architektúra továbbfejlesztett változata szerint került kialakításra. Ennek legfőbb jellegzetessége a személyi számítógépek körében elterjedt Neumann-architektúrával szemben az adat- és programmemória különválasztása. Mivel e két memória külön buszra került illesztésre, konkurens módon is elérhetők. A SHARC architektúra pontosan négy különálló busszal rendelkezik, két adatbusz, egy utasításbusz és egy I/O busz került benne kialakításra [11]. A processzor kettős belső számítási egységgel készült, így az architektúra lehetővé teszi egyetlen órajelcikluson belül egy szorzás vagy ALU művelet, kettős memóriaírás vagy -olvasás végrehajtását valamint egy utasításbeolvasást. Ezek segítségével a programok futási ideje jelentősen lecsökkenthető.

6.2.2. Teljesítményadatok

- 333 MHz processzor órajel-frekvencia, 2 GFLOPs / 66 MMACs

6.2.3. Adatok, jellemzők

- 3 Mbit SRAM (block 0,1: 1 Mbit, block 2,3: 0,5 Mbit)
- Kettős címaritmetika cirkuláris és bitfordított címzéssel
- 4 Mbit egyportos, maszk-programozható ROM (block 0: 2 Mbit, block 1: 2 MBit)
- Overhead nélküli ciklusszervezés
- SIMD[4] architektúra
  - Kettős számításokat végző egység és többszörös buszrendszer
  - Konkurens utasítás-végrehajtás
  - Assembly szintű kódkompatibilitás a SHARC család többi tagjával
- 25 csatornás DMA vezérlő
- Aszinkron párhuzamos port
- Digitális hanginterfész
- 2 db SPI interfész
- S/PDIF kompatibilis digitális audio adó-vevő
- 16 db PWM kimenet
- Kettős tápfeszültség: mag: 1,2 V vagy 1,0 V, I/O: 3,3 V

[3] Super Harvard ARchitecture Computer
[4] Single-Instruction Multiple-Data

7. fejezet

A megvalósításban alkalmazott szoftvereszközök

7.1. A VisualDSP++ fejlesztőkörnyezet

7.1.1. Ismertetés

A PC és a hozzá USB porton csatlakoztatott ADSP-21364 EZ-KIT Lite jelfeldolgozó kártya közötti kapcsolatot magas szinten a VisualDSP++ fejlesztői környezet (IDE[1]) koordinálja, amely szintén az Analog Devices terméke. Segítségével lehetőség van a forrásállományok szerkesztésére, a projektek fordítására, a program a kártyára való letöltésére, a processzor futásának indítására és megállítására. Mindezek mellett a memóriatartalom fejlesztői környezetben történő módosítása és kiolvasása révén hatékony hibakeresési felületet biztosít a fejlesztő számára.

A VisualDSP++ projektszerkesztője és debuggere tehát különálló funkciókat lát el. A projektszerkesztő a forrásállományok szerkesztésére és rendszerezésére szolgál. A forrásállományokat egységekbe foglaló projektekből a fordítás után hardverre tölthető kimeneti fájl készül, mely kiterjesztése `.exe`, hibakeresést is lehetővé tevő esetben pedig `.dxe`. A forrásfájlok lehetnek gépi, azaz assembly nyelven megírtak, vagy C-ben készítettek (`.asm`, `.c` és `.h`). A fordítás közben előálló objektfájlok összekapcsolását az `.ldf` linker állományok definiálják.

7.1.2. Automatizálási lehetőségek

A hallgatói mérés során szükség van a VisualDSP++ által biztosított funkcionalitás használatára, ugyanis a jelfeldolgozó kártyára eltérő struktúrákat megvalósító állományokat kell letölteni. Ilyen funkció lehet a projektek kezelése, fordítása, a programok kártyára való letöltése, a programok futásának indítása és megállítása, valamint egyes memóriaterületek kiolvasása és azok tartalmának módosítása. Ezek működtetéséhez alapesetben a fejlesztői környezet átfogó ismeretére és gyakorlott használatára van szükség. A mérések során azonban nincs idő, sem lehetőség egy ilyen komplex környezet bemutatására, és nem lehet elvárni a hallgatóktól annak azonnali sikeres használatát sem.

Szerencsére a VisualDSP++ lehetőséget biztosít funkcióinak magas szinten történő programból

[1] Integrated Development Environment

való elérésére egy API[2] [19] révén, így lehetőség van a felület kezelését a hallgatók elől elrejteni. Az automatizálásra úgynevezett COM interfész-objektumokon keresztül van lehetőség, melyek révén más programok is hozzáférhetnek a VisualDSP++ funkcionalitásához. A COM objektumokról bővebben a 7.3. pontban lesz szó.

A VisualDSP++ fejlesztőkörnyezet API-ját használó automatizácós rendszer szoftverkomponenseinek kapcsolatát a 7.1. ábra szemlélteti [20]. Az ábrából látható módon a *külső automatizációs kliens*, amely lehet MATLAB, Microsoft Excel, egy Visual Studio program, csak az *API* réteggel van közvetlen kapcsolatban. Az API nem más, mint egy olyan interfész, amely a VisualDSP++ funkcióit külső kliens számára elérhetővé teszi. A kliens és API közötti kapcsolatot a Windows beépített COM kezelő rendszere biztosítja. Az API funkciói ezen kívül a *szkriptmotor*, az ún. Microsoft ActiveX keretrendszer segítségével is elérhetők egy ActiveScript kompatibilis szkriptnyelvből, amely lehet például VBScript vagy JScript.

Lehetőség van a VisualDSP++ fejlesztői rendszerhez olyan *kiegészítő komponenseket* is készíteni, melyek csak az IDE-n belül használhatók, azt saját hozzáadott funkcionalitásukkal kiegészítik. Ezek szintén az automatizációt biztosító API révén kommunikálnak a VisualDSP++ környezettel.

A *szimbólumkezelő* egy olyan szoftverkomponens, amely a lefordított programoknak az ún. *célra* (targetre) történő letöltéséért felelős. Nyilvántartja és igény szerint rendelkezésre bocsátja ezen kívül az egyes változók és hasonló szimbólumok fizikai címeit is hibakeresési és kommunikációs célokból.

A *cél* vagy *target* egy absztrakt komponens, amely a célprocesszorral való mindennemű kommunikáció lehetővé tételéért felelős. Ez implementálja a memória- és regiszterhozzáférést, a futás indítását és megállítását, a szakaszos programfuttatást és a többi ebbe a sorba tartozó funkciót. A cél lehet akár fizikai processzor, vagy esetleg emulátor vagy szimulátor is.

A *processzor-könyvtár* a processzorspecifikus műveletek végrehajtásához biztosít háttéradatokat. Információt nyújt az adott konkrét processzor számábrázolási lehetőségeiről, memóriájáról és regisztereiről is.

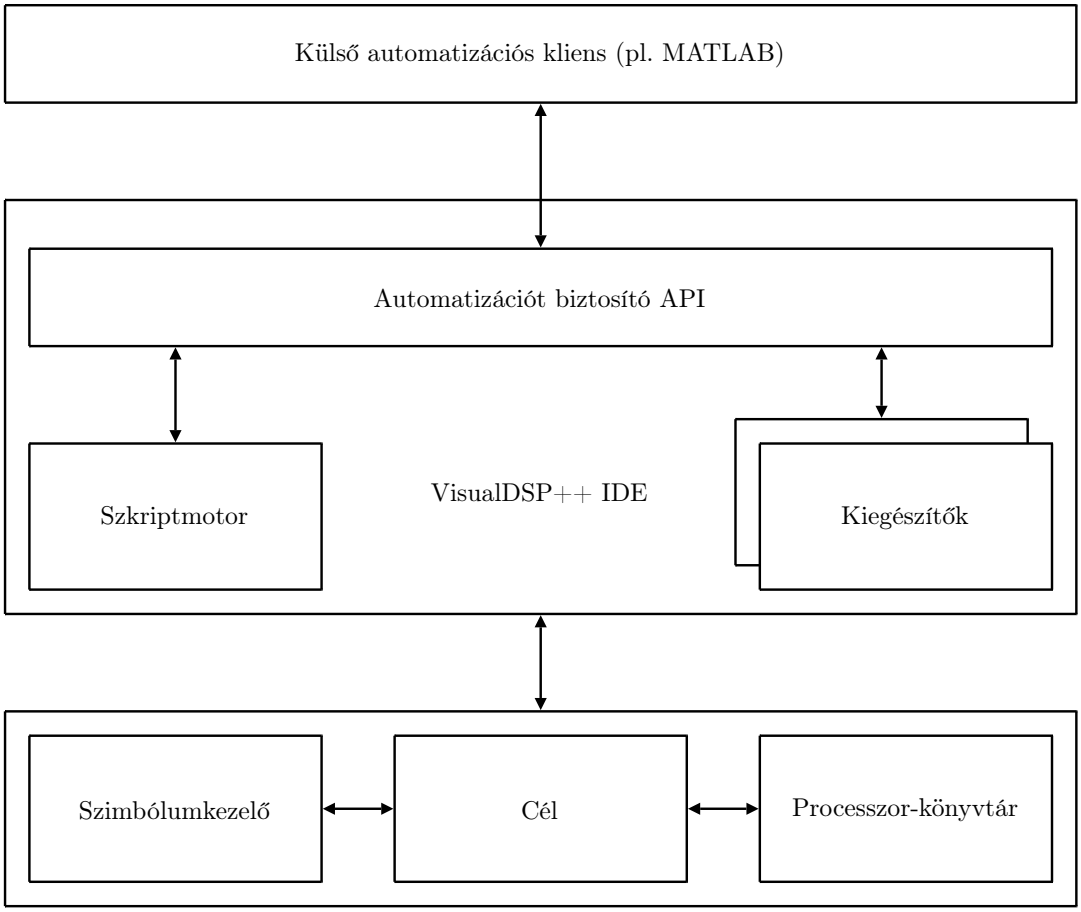
7.2. MATLAB

A Mathworks MATLAB a villamosmérnöki képzés során számos helyen alkalmazott és oktatott programcsomag, amely a hallgatók körében ismert. Kézenfekvőnek kínálkozik a mérést a MATLAB köré felépíteni, ugyanis interpreter jellegű, valamint a numerikus számításokra a hozzáférhető legrugalmasabb és leghatékonyabb eszköz a mérés során.

7.3. A Component Object Model

A *Component Object Model* (COM), amely más néven *ActiveX*-ként is ismert, a *Microsoft* által kifejlesztett technológia a komponens alapú fejlesztés támogatására, amely a különálló szoftverek közötti kommunikációt és adatcserét teszi lehetővé. Bár a COM technológiát több platformon is megvalósították, elsősorban Microsoft Windows operációs rendszereken használatos. A Component Object Model elődje az *Object Linking and Embedding* (OLE) technológia volt, ma a Microsoft *.NET* keretrendszer veszi át szerepét.

[2] Application Programming Interface



7.1. ábra. A VisualDSP++ Automation API környezete

A Component Object Model különböző komponens-típusai osztályazonosítókkal (CLSID[3]) azonosíthatók, amelyek globálisan egyéni azonosítók, *GUID*-ok[4]. A COM osztályokat regisztrálni kell a központilag nyilvántartott „regisztrációs adatbázisban". A globálisan egyedi azonosító egy, a szoftveralkalmazások által használt álvéletlen szám, melynek egyedisége „matematikailag garantált". Ez azt jelenti, hogy az egyedi kulcsok száma annyira nagy (2^{128}, azaz 10^{38} nagyságrendű), hogy két azonos azonosítószám generálásának a valószínűsége gyakorlatilag zérus. A GUID lényegében a Microsoft által megvalósított implementációja az Open Software Foundation (OSF) által specifikált *Univerzálisan egyedi azonosító*, vagy *UUID*[5] szabványnak [12].

Minden COM komponens egy vagy több interfészen keresztül teszi elérhetővé funkcióit. A különböző, az adott komponens által támogatott interfészek az ún. interfész-azonosítókkal különböztethetőek meg egymástól (IID[6]), amelyek szintén *GUID* azonosítók.

A COM interfészeket több programozási nyelvből el lehet érni, például *C*, *C++*, *Visual Basic* használatával, valamint több Windows platformon implementált szkriptnyelvből. A komponensekhez mindennemű hozzáférés az interfészek metódusain keresztül történik. A COM objektumokra folyamaton belül, folyamatok között és hálózaton keresztül lehet hivatkozni.

A komponensek COM típuskönyvtárakkal (*COM Type Library*) tudják önmagukat leírni. A

[3]Class Identifier
[4]Globally Unique Identifier
[5]Universally Unique Identifier
[6]Interface Identifier

típuskönyvtár olyan információkat tartalmaz, mint a komponens CLSID-je, a komponens által megvalósított interfészek IID azonosítója, valamint ezen interfészek metódusainak leírása. A típuskönyvtárakat jellemzően RAD[7] környezetek használják, mint a *Visual Basic* vagy a *Visual Studio*, hogy segítsék a fejlesztőt a komponensek létrehozása során.

[7]Rapid Application Development

8. fejezet

A megvalósított struktúrák

A mérési feladatok nagy részét előre elkészített struktúrák vizsgálata képezi. Ennek lehetővé tétele érdekében a diplomatervezés során ezeket előre el kellett készíteni. A jelfeldolgozó kártyán megvalósított struktúrákon kívül meg kellett valósítani a kártya rejtett, hallgatók által nem látható kezelését, amely kiküszöböli egy fejlesztőrendszer ismeretének igényét. Mindezek mellett ki kellett dolgozni a szimulációs feladatok megoldásait is. A következő pontokban a megvalósított struktúrákat részletesen ismertetem.

8.1. A jelfeldolgozó kártyán megvalósított struktúrák

8.1.1. Adaptív vonaljavító

A jelfeldolgozó kártyán megvalósított egyik struktúra az 5.2. pontban ismertetett adaptív vonaljavító (ALE). A kártya kimenetein a rendszerben előforduló minden jel hozzáférhető, hangszóró csatlakoztatásával meghallgatható, oszcilloszkóp és spektrumanalizátor segítségével vizsgálható, illetve a számítógép hangkártyájához csatlakoztatva igény esetén bármely jelből regisztrátum rögzíthető.

Az 5.2. ábrán látható jelöléseket alkalmazva, az egyes jelek a következő kimeneteken férhetők hozzá:

- y: 1. kimenet bal és jobb csatornája
- $\hat{y}$: 2. kimenet bal és jobb csatornája
- e: 3. kimenet bal és jobb csatornája
- x: 4. kimenet bal és jobb csatornája.

A bemenet tehát az első kimeneten módosítás nélkül hozzáférhető, míg a negyedik kimeneten a késleltetett változata kerül előállításra. Az ALE struktúra kimeneti jele, azaz a becsült jel a második kimeneten, míg a hibajel a harmadik kimeneten mérhető.

A struktúrában a bemeneti jelet egy olyan cirkuláris buffer tárolja, amelynek mérete megegyezik az adaptív szűrő hosszának és a késleltetés ciklusszámának összegével. A késleltetés oly módon valósul meg, hogy a mintavételi ciklusonként beérkező új adat eltolt pozícióba kerül a bufferbe.

8.1.2. Aktív zajcsökkentő FxLMS algoritmussal

A jelfeldolgozó kártyán az 5.5. ábrán látható, FxLMS algoritmust alkalmazó aktív zajcsökkentő struktúra is elkészítésre került. A program a másodlagos átviteli út identifikációját és a zajcsökkentést egyaránt implementálja. A funkciók közül a `FLAG1 / SW1` nyomógomb megnyomásával lehet váltani. A `FLAG1` nyomva tartásának idejéig a struktúra identifikációs módban üzemel, melyet a `LED1` felvillanása jelez.

A csatlakozók bekötését a következőknek megfelelően kell elvégezni:

- Bemenetek:
  - 1. bemenet (fehér): referenciajel
    `ANC` módban: x
    `Ident` módban nincs használatban
  - 2. bemenet (piros): hibamikrofon jele
    `ANC` módban: e
    `Ident` módban: y_i.
- Kimenetek:
  - 1. kimenet mindkét csatornája: beavatkozó hangszóró jele
    `ANC` módban: $-\hat{y}$
    `Ident` módban: fehér zaj
  - 2. kimenet mindkét csatornája:
    `ANC` módban: hibajel (e)
    `Ident` módban nincs használatban.

Identifikációs módban a beavatkozó hangszóróból a jelfeldolgozó kártyán generált álvéletlen zaj hallható. A zajt szoftveres LFSR[1], azaz az előírásoknak megfelelően visszacsatolt léptetőregiszter állítja elő [7]. Mivel az implementáció C programozási nyelvben történt, a választás az ún. *Galois LFSR* struktúrára esett annak előnyös megvalósítási tulajdonságai és kis számításigénye miatt. Az így előállított jelet spektrumanalizátor segítségével megvizsgálva azt tapasztaltuk, hogy a decimált mintavételi frekvencia feléig a zaj spektruma fehérnek látszik.

Identifikáció során a fent körülírt módon előállított gerjesztőjel egyrészt a beavatkozó hangszóró segítségével kisugárzásra kerül, másrészt a program az adaptív szűrő együtthatóival konvolválja, vagyis az $coef\_s(z)$ adaptív FIR szűrővel szűri. Az így előálló $\hat{y}_i$ jelet a mikrofon y_i beolvasott jeléből kivonva megkapható az e hibajel mintasorozata, amely aztán a referenciajellel együtt az LMS algoritmus által a szűrőegyütthatók hangolásában kerül felhasználásra. A FIR szűrés működtetése érdekében a gerjesztőjel annyi darab mintáját kell cirkuláris buffer segítségével tárolni, amennyi együtthatóval a szűrő rendelkezik.

Aktív zajcsökkentő üzemmódban a $coef\_s(z)$ átvitel már nem módosul. Az x referenciajel egy mintájának beolvasása után egy olyan hosszú cirkuláris bufferbe kerül, mely hossza elegendő a FIR szűrések végrehajtásához. A $coef\_w(z)$-vel történő szűrés után az $\hat{y}$, a $coef\_s(z)$-vel való szűrés után pedig az x_c jel áll elő, mely szintén cirkuláris bufferben tárolódik annak érdekében, hogy a $coef\_w(z)$ szűrő együtthatói segítségével, és a szintén beolvasott e hibajel mintáinak felhasználásával adaptálni lehessen.

[1] Linear Feedback Shift Register

A `FLAG2 / SW2` nyomógomb megnyomásával a $coef\_w(z)$ szűrő, míg a `DAI_P19 / SW3` nyomógombbal a $coef\_s(z)$ szűrő együtthatói nullázhatók. Az előbbi használatával a rendszer beállási tulajdonságai vizsgálhatók, míg a másodikkal a referenciajel vehető le az LMS algoritmus bemenetéről azáltal, hogy csupa nulla együtthatójú szűrővel kerül megszűrésre. Referenciajel hiányában a $coef\_w(z)$ együtthatói nem módosulnak.

A `DAI_P20 / SW4` nyomógomb segítségével a kimenet tiltható. A szűrőegyütthatók stabilitása megőrzésének érdekében letiltott kimenet esetén a $coef\_w(z)$ együtthatók adaptálása is felfüggesztésre kerül.

8.2. MATLAB szimulációk

Az előzetesen elkészített MATLAB szimulációk tulajdonképpen a megfelelő mérési feladatok megoldásai, melyekben a hallgatók feladata MATLAB kód készítése.

A megoldás az algoritmusok implementációján kívül kiterjed a működtetési paraméterek megfelelő értékeinek kikísérletezésére. A mérésvezető számára tehát kvantitatíve rendelkezésre állnak azok a paraméterek, amelyek beállítása esetén az egyes struktúrák működése jól demonstrálható.

8.2.1. LMS algoritmus, rendszeridentifikáció

Az első programozási feladat az LMS algoritmus megvalósítása olyan MATLAB függvényben, amely rendszeridentifikációs feladat végrehajtására alkalmas. A függvény a számára átadott, az identifikálandó rendszer bemenetének és kimenetének megfelelő regisztrátumvektorok felhasználásával annyi iterációt végez a megszabott méretű szűrő adaptálására, amennyit a regisztrátumok hossza lehetővé tesz. Az adaptációhoz szüksége van továbbá a bátorsági tényező értékére is.

A függvény visszatér egyrészt az iterációk végeztével előálló szűrőegyüttható-készlettel, valamint az egyes lépésekben számított különbségi hibából készített vektorral, amely annyi értéket tartalmaz, ahány iteráció történt.

A megvalósított függvény az N hosszú `x` és `y` paraméter-vektorokban várja az identifikálandó rendszer be- és kimeneti jeleinek értéksorozatát. Az `M` skaláris paraméterben a szűrő M hosszát, míg a `mu` skaláris paraméterben a μ bátorsági tényező értékét kell átadni. A `w` szűrőegyütthatókészlet az M hosszú `w` vektorban, az e hibajel pedig az N hosszú `e` vektorban áll rendelkezésre a függvény visszatérése után. A hibajel-vektor első M db értéke zérus.

A következő program az előzőleg megvalósított algoritmus verifikációjára használható. Ennek alapgondolata, hogy ha az identifikálandó rendszer is FIR szűrő, és hossza is megegyezik az LMS algoritmus által hangolt adaptív szűrőével, akkor megfelelő számú iteráció és jó konvergenciatulajdonságok esetén a két paraméterkészlet jó közelítéssel megegyezik.

A verifikációhoz tehát elő kell állítani egy ismert paraméterkészlettel rendelkező FIR szűrőt reprezentáló vektort, illetve egy álvéletlen szekvenciából álló olyan vektort, amely a bemeneti jelet fogja reprezentálni. A bemeneti jelet a szűrő impulzusválaszával konvolválva előállítható a szűrő kimenete. Mindezeken kívül már csak a bátorsági tényezőt szükséges alkalmas módon megválasztani, és következhet az LMS algoritmus függvényének hívása. A függvény visszatérése után az általa előállított együtthatókészlet összevethető az ismert szűrő impulzusválaszával. A hibajelet ábrázolva és megtekintve az algoritmus beállási és stabilitási tulajdonságairól szerezhetünk információkat.

8.2.2. NLMS algoritmus, visszhangcsökkentés

Az NLMS algoritmus kódja az LMS program kiegészítésével állítható elő. A program bemeneti és kimeneti paraméterei is megegyeznek az ott használtakkal, azzal a kiegészítéssel, hogy az NLMS esetében szükség van továbbá az a korrekciós tényező értékére is, amelyet az `a` skaláris paraméterben kell átadni.

Az algoritmus kipróbálására előre elkészített regisztrátumok állnak rendelkezésre, melyek a hallgatóoldali hibrid bemeneti és kimeneti jelei. Mivel az adaptív visszhangcsökkentés feladata a hibrid identifikációja, az NLMS algoritmust realizáló függvénynek ezeket a jelvektorokat kell átadni. A hibajel-vektorban a maradó visszhang áll rendelkezésre a függvény lefuttatása után. Az NLMS algoritmus előnyös tulajdonságai akkor vizsgálhatók, ha a maradó visszhang az LMS algoritmus esetében és az NLMS algoritmus alkalmazásával összehasonlításra kerül.

8.2.3. FxLMS algoritmus

Az FxLMS algoritmust realizáló kód szintén az LMS programkódjából származtatható. A be- és kimeneti függvényparaméterek sorához csak a másodlagos út átvitelét kell hozzátenni. Ezt az I hosszú `c` vektorban várja a függvény. A program működése az LMS algoritmus mintájára vizsgálható. Ha a bemeneti jelet az elnyomandó zajnak tételezzük fel, a struktúra az aktív zajcsökkentő rendszerek működését modellezi.

9. fejezet

Értékelés, továbbfejlesztési lehetőségek

A diplomaterv keretében kidolgozott hallgatói mérésben minden olyan témakör bemutatásra került, amely a diplomaterv-kiírásban elő volt írva. A feladatok ennek alapján felölelik az LMS algoritmus megvalósítását és a segítségével adaptált véges impulzusválaszú szűrők tulajdonságainak analízisét. Az LMS algoritmus bemutatásra került mind rendszeridentifikációs feladatban, mind gyakorlati jelentőssséggel bíró struktúrák részeként. A mérésben helyt kapott az adaptív vonaljavító (ALE) és az adaptív visszhangcsökkentő (AEC) rendszer egyaránt. Ez utóbbi struktúra ismertetésénél sor került az NLMS algoritmus megvalósítására és vizsgálatára is. Bemutatásra került egy aktív zajcsökkentő (ANC) struktúra is, melynek vizsgálata előtt az FxLMS algoritmus implementálása is a feladatok része lett.

Az aktív zajcsökkentő struktúra nagy hátránya, hogy akusztikai csatornát alkalmaz, így a mérés zajjal jár, a többi mérőcsoport munkavégzését zavarhatja. Mivel a fizikailag megvalósított rendszerből, mely a mérési útmutatóban részletesen ismertetésre került, jelenleg egyetlen példány áll rendelkezésre, a következő lehetőségek jöhetnek számításba:

- Az aktív zajcsökkentő struktúra vizsgálatát a mérőcsoportok együttesen, oktatói segítséggel végzik, ugyanis a megfigyelhető jelenségek legtöbbje magyarázatot igényel a pontos megértéshez.
- Mivel hangszóróból és mikrofonból több példány is rendelkezésre áll, a struktúra sikeresen üzemeltethető akár a laboratórium akusztikai terében is, az erre vonatkozó kísérletek is azt mutatták, hogy a rendszer ilyen körülmények között is működőképes.
- Lehetőség van az ismertetett rendszerből több azonos példány elkészítésére is.
- A laboratórium felszerelései között nagy számban rendelkezésre áll több típusú induktív távolságmérő szenzor, amelyek segítségével megvalósítható a mérőhidak FxLMS algoritmus segítségével történő nullkompenzálása. Erről az elrendezésről a gyakorlati alkalmazásokkal foglalkozó részben részletes ismertetést adtam. A struktúra mellett szól, hogy nem jár a működés közben hallható zajjal, és a másodlagos átviteli út is kedvezőbb tulajdonságokkal rendelkezik, mint az aktív zajcsökkentő rendszer. Hátránya viszont, hogy működtetéséhez új hardver fejlesztésére volna szükség, amely az elektronikai komponensein túlmenően mechanikai problémákat is felvet, valamint működtetése sem jár az ANC rendszerrel összemérhető

„látványos” eredménnyel. Az ANC üzemeltetése közben hallható jelek nagy mértékben hozzájárulnak ahhoz, hogy a hallgatók testközelből kerüljenek kapcsolatba az adaptív jelfeldolgozó rendszerekkel és azok tulajdonságait is a lehető legjobban megértsék. Mindezek alapján a mérési feladatok közé végül az aktív zajcsökkentést megvalósító struktúra került be.

A munka során megvalósított szimulációk és fizikai struktúrák teljes mértékben működőképesek, azok további fejlesztésére nincs szükség. Továbbfejlesztési lehetőséget a fizikai rendszerek grafikus felhasználói felületének bővítésében látok. Az ily módon hozzáadott funkcionalitással lehetővé válna a rendszerek további paramétereinek módosítása is. A tervezés során azonban úgy ítéltem meg, hogy csak azon paraméterek állítását teszem a hallgatók számára a grafikus felület révén lehetővé, amelyek feltétlenül szükségesek a rendszerek tulajdonságainak vizsgálatához. A struktúrák ugyanis viszonylag érzékenyek például a szűrők hosszának túl kicsire vagy esetleg túlzottan nagyra történő választására, így az optimális és kipróbált értékek a program forráskódjában, konstansban definiált értékek maradtak. Ezek átírása a mérésvezető számára természetesen semmilyen problémát nem jelent.

Értékelésként elmondható, hogy az elkészített mérési útmutató szükséges és elegendő mennyiségű elméleti ismeretet tartalmaz a mérés elvégzéséhez, ha a hallgatók minden olyan előzetes ismeret birtokában vannak, amelyet feltételeztem. Az elméleti ismertetésen túl a struktúrák működéséről is megfelelő összefoglalást adtam. A mérési feladatok megfelelően egymásra épülnek, a rendelkezésre bocsátott tudnivalók birtokában elvégezhetők. A feladatok során minden olyan – az adaptív szűrőket tartalmazó rendszereket jellemző – tulajdonság bemutatásra került, amely a célkitűzések között szerepelt. A feladatok keretében bemutatott struktúrákat oly módon választottam ki, hogy segítségükkel a lehető legjobban meg lehessen érteni az adaptív jelfeldolgozó algoritmusok lényeges kérdéseit. Fizikailag, jelfeldolgozó kártyán olyan rendszerek kerültek megvalósításra, melyek működtetése látványos, valamint működtetésük hozzájárul egy olyan szemléletmód kialakításához, amelyet a feldolgozott témakör hivatott felépíteni.

10. fejezet

Összefoglalás

Diplomamunkám keretében sikeresen elkészült az *Adaptív szűrők vizsgálata* című laboratóriumi mérés tematikája, valamint megszületett a részletes mérési és mérésvezetői útmutató dokumentuma is. A munka során nem csak a feladatok egyszerű kijelölése történt meg, hanem megvalósultak az azokhoz szükséges struktúrák, részben jelfeldolgozó kártyán realizált program formájában, részben számítógépes szimulációkként MATLAB-ban.

A munka során felfrissítettem a modellillesztéssel és az adaptív szűrők elméleti alapjaival kapcsolatos ismereteimet, valamint az irodalomkutatás során megismertem számos olyan struktúrát, melyben adaptív jelfeldolgozást alkalmaznak. A feladatok kijelölése során olyan irányú jártasságot szereztem, amely segítségével didaktikai szempontok alapján tudtam mérlegelni, milyen szintű elméleti áttekintést szükséges a hallgatók rendelkezésére bocsátani a mérést megelőző otthoni felkészülés segítésére, valamint mérlegelni tudtam azt is, milyen lépések szerint célszerű egy adott struktúrát bemutatni, annak vizsgálatának folyamatát irányítani. Teljesíteni tudtam azt a követelményt is, hogy az MSc képzés jellegéből adódóan inkább elméleti, tudományos szemléletmódot kell a mérés során követni és a feladatokat ennek tudatában kell kijelölni.

A feladatokban alkalmazott rendszerek implementációja során megismerkedtem az adaptív jelfeldolgozó algoritmusok gyakorlati alkalmazásának módszereivel. A jelfeldolgozó processzorokkal kapcsolatos ismereteim elmélyítése mellett a C és assembly nyelvű DSP programozást is adott szinten el kellett sajátítanom. A mérési feladatokban kijelölt valóságos fizikai rendszerek az Analog Devices által gyártott ADSP-21364 lebegőpontos jelfeldolgozó processzoron kerültek implementálásra az ADSP-21364 EZ-KIT Lite fejlesztőkártya segítségével. A szimulációk MATLAB-ban kerültek megvalósításra.

A mérési feladatokat elvégeztem, és a megoldásokat, mérési eredményeket, valamint az azokból levonható következtetéseket a mérésvezetői útmutatóban rendelkezésre bocsátottam.

A diplomaterv során részletezett témák közül újdonságnak számít a jelfeldolgozó kártya VisualDSP++ fejlesztőkörnyezetének vezérlése MATLAB alól, mely részben az önálló laboratóriumi munka során került kifejlesztésre. A probléma megoldására számos szoftvertechnológiai nehézséggel kellett megküzdeni, ugyanis a rendelkezésre álló dokumentáció meglehetősen szűkszavú volt, az interneten rendelkezésre álló irodalomban pedig csak elvétve lehetett találkozni a VisualDSP++ vezérlési lehetőségeit taglaló dokumentumokkal.

Irodalomjegyzék

[1] BME MIT Tanszéki Munkaközösség, *Digitális jelfeldolgozás*, Segédlet, Budapesti Műszaki és Gazdaságtudományi Egyetem, 2006.

[2] Widrow, Wallach, *Adaptive Inverse Control*, Prentice Hall PTR, Upper Saddle River, New Jersey, 1996.

[3] Elek Kálmán, *Adaptív jelfeldolgozás*, Műegyetemi Kiadó, 2005.

[4] Kuo, S.M., Morgan, D.R., „Active Noise Control: A Tutorial Review", *Proceedings of the IEEE*, vol. 87, No. 6, pp 943-973, 1999.

[5] Fürjes Andor Tamás, *Akusztika és Hangtechnika*, Jegyzetvázlat, Budapesti Műszaki és Gazdaságtudományi Egyetem, 2008.

[6] Elliott., S. J., „Distributed Control of Sound and Vibration", 2004 Int. Symp. On Active Noise Control of Sound and Vibration, ACTIVE '04, pp 1-25, 2004

[7] Wikipedia: Linear feedback shift register, megtekintve: 2008. 05. 02.
`http://en.wikipedia.org/wiki/Linear_feedback_shift_register`

[8] ADSP-21364 EZ-KIT Lite Evaluation System Manual, Revision 3.0, August 2006, megtekintve: 2008. 03. 11.
`http://www.analog.com/UploadedFiles/Associated_Docs/33168315105663ADSP_21364_EZ_KIT_Lite_Manual_Rev._3.0.pdf`

[9] Analog Devices, SHARC Processors: ADSP-21362/ADSP-21363/ADSP-21364/ADSP-21365/ADSP-21366, Rev. D, 2008, megtekintve: 2008. 02. 21.
`http://www.analog.com/UploadedFiles/Data_Sheets/ADSP_21362_21363_21364_21365_21366.pdf`

[10] SHARC Processor Architectural Overview, megtekintve: 2008. 04. 14.
`http://www.analog.com/processors/sharc/overview/archOverview.html`

[11] Getting Started With SHARC Processors, megtekintve: 2008. 04. 14.
`http://www.analog.com/UploadedFiles/Associated_Docs/352228244SHARC_getstart_online.pdf`

[12] Wikipedia: Universally Unique Identifier, megtekintve: 2008. 05. 03.
`http://en.wikipedia.org/wiki/UUID`

[13] Knill, K.M., Constantinides, A.G., „Least-Mean Square Adaptation of the Orthogonal Block Adaptive Line Enhancer", Signal Processing Section, Dept. of Electrical and Electronic Engineering, Imperial College, London, UK, megtekintve: 2008. 04. 02.
`http://citeseer.ist.psu.edu/370480.html`

[14] Department of Electroscience, Faculty of Engineering, Lund Institute of Technology, „Adaptive Signal Processing", megtekintve: 2008. 04. 03.
`http://www.es.lth.se/ugradcourses/asb/asb.html`

[15] Wikipedia: Digital signal processor, megtekintve: 2008. 05. 03.
`http://en.wikipedia.org/wiki/Digital_signal_processor`

[16] Molnár Károly, Sujbert László, „Jelfeldolgozó processzor alkalmazása", Mérési útmutató, 2006, megtekintve: 2008. 05. 02.
`http://portal.mit.bme.hu/oktatas/targyak/vimm5158/jegyzet/DSP-meres.pdf`

[17] ADSP-2136x SHARC Processor Programming Reference, Revision 1.1
megtekintve: 2008. 04. 10.
`http://www.analog.com/UploadedFiles/Associated_Docs/`
`979493310ADSP_2136x_PGR_rev1_1.pdf`

[18] VisualDSP++ 4.5 C/C++ Compiler and Library Manual for SHARC Processors, Revision 6.0, April 2006, megtekintve: 2008. 04. 10.
`http://www.analog.com/UploadedFiles/Associated_Docs/`
`62998592645399486989674 45_SHARC_comp_man.pdf`

[19] Application Programming Interface, megtekintve: 2008. 05. 03.
`http://en.wikipedia.org/wiki/API`

[20] VisualDSP++ 4.5 Súgó: „Automation" fejezet

[21] Sujbert, L., G. Péceli, Gy. Simon, „Resonator based non-parametric identification of linear systems", *IEEE Trans. on Instrumentation and Measurement*, vol. 54. no. 1., pp. 386-390, 2005.

[22] Orosz György, *Adaptív jelfeldolgozó eljárások megvalósítása szenzorhálózatban, Diplomaterv*, Budapesti Műszaki és Gazdaságtudományi Egyetem, 2006.

[23] Chair of Multimedia Communications and Signal Processing, University of Erlangen-Nuremberg, „Echo and feedback cancellation", megtekintve: 2008. 04. 24.
`http://www.lnt.de/lms/research/projects/echo/index.php?lang=eng`

[24] Michael Hutson, *Acoustic echo cancellation using digital signal processing*, The University of Queensland, 2003.

[25] Horváth András, *Aktív zajcsökkentést demonstráló rendszer tervezése, Diplomaterv*, Budapesti Műszaki és Gazdaságtudományi Egyetem, 2007.

[26] The International Engineering Consortium (IEC), Web ProForum Tutorials, „Echo Cancellation", 2002.

Függelék

11. fejezet

Adaptív szűrők vizsgálata – mérési útmutató

11.1. Bevezetés

A mérés célja a korábbi szakirányú tanulmányok során megszerzett, a digitális jelfeldolgozás témakörébe tartozó adaptív szűrőkkel kapcsolatos ismeretek elmélyítése, azok gyakorlati vonatkozásainak megismerése. A mérés tematikája szerint a már meglévő elméleti ismeretek szükséges szintű elmélyítését azok gyakorlati alkalmazásainak vizsgálata követi.

11.2. A mérés célja

A manapság jellemző tendenciák szerint egyre jobban terjed az adaptív jelfeldolgozás gyakorlati alkalmazása, így ez a témakör a villamosmérnök eszköztárát már nem csak elméleti síkon, hanem gyakorlati szinten is bővíti.

Napjainkban a gyártók termékpalettáján kínált jelfeldolgozó processzorok számítási teljesítménye egyre növekszik, ezzel párhuzamosan áruk egyre alacsonyabb, így elterjedtségük is növekvő tendenciát mutat, valamint olyan területeken is teret hódítanak, ahol eddig magas áruk vagy nem elégséges komplexitásuk miatt nem alkalmazták őket. A mai, korszerű jelfeldolgozó processzorok tehát olyan gyakorlati problémák megoldására kínálnak lehetőséget, amelyek korábban elérhetetlenek és elképzelhetetlenek voltak – többek között – a bonyolult valósidejű jelfeldolgozás igénye miatt.

Mára tehát egyre több alkalmazásban van szükség illetve lehetőség adaptív jelfeldolgozás beépítésére. Ezek legnagyobb csoportja az akusztikával van szoros kapcsolatban. Az akusztikai tér átvitele ugyanis két fizikai hely – például hangszórók és mikrofonok, vagy természetes zajforrások és az emberi fül – között jellemzően változó. Az akusztikai átvitelt nagyon sok tényező befolyásolja, melyek közül példaként említhető a belső téri berendezés, az emberek, tárgyak elhelyezkedése, a légnyomás, hőmérséklet, páratartalom, vagy akár a nyílászárók nyitott vagy csukott állapota egyaránt.

Számos gyakorlati feladatban alkalmazott algoritmus megfontolásai szerint szükség van viszont adott akusztikai átvitel vagy átvitelek megfelelő pontosságú ismeretére. Ez az előző szempontokból kiindulva, az átvitel sztochasztikusan változó jellegű voltának ismeretében nem egyértelmű feladat.

Lehetővé kell tenni tehát valamilyen módon egy ismeretlen és az időben változó átvitelű rendszer modelljének kielégítő pontosságú előállítását, valamint azt, hogy a modell a fizikai rendszer változásait valós időben megfelelő gyorsasággal kövesse.

A probléma tehát a modellillesztés elméleti feladatát tűzi ki célul, melynek része a modell parametrikus vagy strukturális meghatározása, vagy a rendszerrel hasonló viselkedésű, de eltérő felépítésű modell kialakítása.

11.3. A mérés elméleti alapjai

A mérés elvégzéséhez mindenképpen szükséges az adaptív jelfeldolgozás elméleti hátterének átfogó ismerete, ugyanis a mérési feladatok egy részében jelfeldolgozó algoritmusok MATLAB-ban történő implementációját kell elvégezni, a többi feladatban pedig fizikailag realizált struktúrákon kell méréseket és paraméterhangolási feladatokat végezni.

11.3.1. A modellillesztés elméleti alapjai

A modellillesztés feladata – egy mondatban összefoglalva – egy ismeretlen rendszernek megfelelő modell strukturális meghatározása, illetve adott esetben az előzőleg ismertnek feltételezett struktúra paramétereinek meghatározása (*paraméterbecslés*). A gyakorlatban leginkább a paraméterbecslési feladattal találkozunk, melynek során az ismeretlen rendszerre legjobban illeszkedő paraméterekkel rendelkező, előzetesen adott struktúrájú modellt keressük.

A paraméterbecslés során a valóságos rendszer és a hozzá illesztett modell kimeneteinek eltérését adott hibakritérium alapján kívánjuk minimalizálni. A modellillesztés feladatköre ezen belül is két csoportra bontható. Az *identifikáció* célja az időben állandónak feltételezett rendszer paramétereinek meghatározása, míg az *adaptáció* feladata az időben változó rendszerparaméterek követése. Az identifikáció során tehát nagy pontosságú mérés elvégzésére van elvi lehetőség, ugyanis a rendszert a szükséges hosszú ideig megfigyelhetjük, a szükséges mennyiségű adatot gyűjthetjük be róla. Az identifikáció ebből következően jellemzően időigényes folyamat. Az adaptáció során kevésbé fontos a nagyfokú pontosság és a modellnek a valóságos rendszerhez való pontos illeszkedése. Sokkal inkább lényeges, hogy a modell paramétereinek változtatásával a fizikai rendszer változásait követni tudjuk, és a modell a rendszerrel valós időben tudjon együtt mozogni. Természetesen a fizikai rendszer struktúrája nem ismert, és paraméterei sem mérhetők, kizárólag a fizikai rendszer és a modell kimenetének eltérése adhat támpontot a modell paramétereinek korrigálásához. Ehhez az esetek döntő többségében le kell mondanunk a nagy pontosságú modellről, és helyette egy egyszerűbb, a gyakorlatban könnyebben kezelhető struktúrát kell alkalmaznunk, mely még elfogadható a valóságos rendszer modellezésére.

A modellillesztés egy speciális esete a regressziószámítás feladata, amely célja bizonyos változók között fennálló determinisztikus kapcsolat meghatározása. Ezen változók a gyakorlatban legtöbbször bizonyos rendszerek be- és kimenetei. A valóságos rendszerek, melyekre modellt kívánunk illeszteni, a bemenetük és kimenetük között azonban nem tisztán determinisztikus függvénykapcsolatot valósítanak meg, mert a kimenet rendszerint zajjal terhelt, tehát a determinisztikus mellett sztochasztikus komponenst is tartalmaz. Az illesztendő modell ellenben kizárólag determinisztikus függvénykapcsolatot realizál, melynek kialakítása tipikusan bizonyos szabad paraméterek valamilyen szempontból megválasztott elv alapján történő beállítását jelenti a mérések után rendelkezésre álló bemenet-kimenet adatpár-sorozat alapján.

A paraméterek hangolásához egy a valóságos rendszer és az illesztett modell kimeneteinek különbségétől függő ún. *költségfüggvényt* definiálunk, és a paraméterek hangolásával ennek minimális értékét kíséreljük meg beállítani. Célunk tehát annak elérése, hogy a modell kimenete a lehető legjobban közelítse a valóságos rendszer kimenetét, a „lehető legjobban" kritérium egzakt leírása pedig a költségfüggvény vagy hibakritérium-függvény kifejezése:

$$\varepsilon = \mathrm{E}\{(\mathbf{y} - \hat{\mathbf{y}})^T(\mathbf{y} - \hat{\mathbf{y}})\}. \tag{11.1}$$

A költségfüggvény egy előnyös megválasztása az, ha értéke a kimenetek különbségétől, vagyis a modell hibájától négyzetesen függ, ugyanis az ilyen költségfüggvény minimum-keresése matematikailag egyszerű, és emellett fizikailag a hibajel teljesítményének minimalizálását végezzük. Az olyan szűrőt, melynek együtthatóit ilyen módon definiált költségfüggvény értékének minimalizálása érdekében hangoljuk, *Wiener-szűrőnek* nevezzük [3].

A paraméterillesztés elvégzéséhez azonban nem elegendő a mért értékpárok pillanatnyi értékeinek ismerete, szükség van azok statisztikai jellemzőinek, vagy legalábbis bizonyos fokú momentumaiknak ismeretére. A lineáris regresszió alkalmazásakor például elegendő, ha az első- és másodfokú momentumok rendelkezésre állnak.

Ha az illesztendő modell speciálisan a paraméterek lineáris függvénye, és négyzetes költségfüggvényt alkalmazunk, a hibafüggvény a paraméterek síkja fölött kifeszített felület, melynek minimumkeresése után azonnal egyértelműen kiadódik a paraméterek optimális értéke. Ehhez viszont szükséges a hibafelület teljes ismerete.

Az illesztendő modell bemenete, kimenete és paraméterei közötti összefüggés a következő:

$$\hat{\mathbf{y}} = \hat{\mathbf{g}}(\mathbf{u}) = \hat{\mathbf{g}}(\mathbf{W}, \mathbf{u}), \tag{11.2}$$

ahol $\mathbf{u}$ a bemenőjel-vektor, $\mathbf{W}$ az állítható paraméterek mátrixa, $\hat{\mathbf{y}}$ pedig a modell kimenete. A bemenet és kimenet, valamint a modell kimenete mind-mind legalább gyengén stacionárius sztochasztikus folyamatok bizonyos realizációi. A jelölés kifejezi, hogy a modell a $\mathbf{W}$ paramétermátrixon keresztül adaptálható.

A modell dinamikus és esetlegesen nemlineáris részét ($f(\mathbf{u}) = \mathbf{x}$) célszerű rögzíteni, és különválasztani a lineáris, adaptálható résztől ($\mathbf{W}$):

$$\hat{\mathbf{y}} = \hat{\mathbf{g}}(\mathbf{W}, \mathbf{u}) = \mathbf{W}^T f(\mathbf{u}) = \mathbf{W}^T \mathbf{x}, \tag{11.3}$$

ahol $\mathbf{x} = f(\mathbf{u})$ a modell rögzített része, amely az $\mathbf{u}$ bemenőjel-vektorból előállítja az $\mathbf{x}$ regressziós vektort, amely ezután az adaptálható rész bemenetét képezi. Az $\hat{\mathbf{y}}$ egyetlen kimenet esetén két vektor skaláris szorzata, több kimenet esetén viszont a képletből is látható módon mátrix-vektor szorzatként áll elő. A kifejezések alakja egy bemenet és egy kimenet esetében:

$$\hat{y} = \hat{g}(\mathbf{w}, u) = \mathbf{w}^T f(u) = \mathbf{w}^T \mathbf{x}. \tag{11.4}$$

A sztochasztikus folyamatokról az időfüggvényekre (a folyamatok realizációira) áttérve:

$$\hat{y}(n) = \mathbf{w}^T(n)\mathbf{x}(n) = \sum_{i=1}^{M-1} w_i(n) x_i(n), \tag{11.5}$$

ahol M a $\mathbf{w}$ és $\mathbf{x}$ vektorok hossza, a mátrix-vektor szorzat pedig vektorok skaláris szorzatává

egyszerűsödik, és (11.5) alapján számítható.

A szétválasztás eredményeképpen az adaptálható rész paramétereiben lineáris, és a paraméterek megváltoztatása által okozott tranziens véges idő alatt kifut a rendszerből.

Mivel már felírtuk a modell kimenetének kifejezését az $\mathbf{x}$ regressziós vektor illetve a $\mathbf{w}$ paramétervektor segítségével (11.4 és 11.5), írjuk fel a modellillesztéshez használt kritériumfüggvényt (11.1) az átlagos négyzetes hiba alapján:

$$\varepsilon(n) = \mathrm{E}\{(y(n) - \hat{y}(n))^2\} = \mathrm{E}\{(y(n) - \mathbf{w}^T\mathbf{x}(n))^2\} = \tag{11.6}$$

$$= \mathrm{E}\{y^2(n)\} - 2\underbrace{\mathrm{E}\{y(n)\mathbf{x}^T(n)\}}_{\mathbf{p}^T}\mathbf{w} + \mathbf{w}^T\underbrace{\mathrm{E}\{\mathbf{x}(n)\mathbf{x}^T(n)\}}_{\mathbf{R}}\mathbf{w} = \tag{11.7}$$

$$= \mathrm{E}\{y^2(n)\} - 2\mathbf{p}^T\mathbf{w} + \mathbf{w}^T\mathbf{R}\mathbf{w}, \tag{11.8}$$

ahol $\mathbf{p}^T$ a kimenet és a regressziós vektor közötti keresztkorrelációs vektor, illetve $\mathbf{R}$ a regressziós vektor autokorrelációs mátrixa. A kritériumfüggvény (11.8) szerinti kifejezése egy $M+1$-dimenziós térben elhelyezkedő paraboloidot ír le. A kritériumfüggvény minimuma az M-dimenziós $\mathbf{w}$ vektor szerinti differenciálással a következőnek adódik:

$$\frac{\partial\varepsilon}{\partial\mathbf{w}} = -2\mathbf{p} + 2\mathbf{R}\mathbf{w} = 2(\mathbf{R}\mathbf{w} - \mathbf{p}) = \mathbf{0}, \tag{11.9}$$

melynek megoldása a Wiener-Hopf egyenlet, mely megadja az optimum helyét a $\mathbf{w}$ szűrőegyütthatók terében:

$$\mathbf{w}_{opt} = \mathbf{R}^{-1}\mathbf{p}. \tag{11.10}$$

Az optimális paraméterkészlet meghatározása ezek alapján szélsőérték-keresési feladat. Az eredmény nehézsége viszont az, hogy a $\mathbf{w}_{opt}$ együtthatókészlet számításához a modellezendő rendszer be- és kimeneti jeleinek statisztikai tulajdonságait *a priori* ismernünk kell a korrelációs vektorok és mátrixok meghatározásához.

A korrelációs vektorok ismeretének hiányában illetve azok részleges ismerete esetében a paraméterbeállítás csak iteratív módon végezhető. Ekkor a statisztikai paraméterek helyett a jelek pillanatnyi értékeinek felhasználásával törekszünk az optimális paraméterkészlet megközelítésére. Az iteráció egyes lépéseiben meg kell határozni, hogy az adott pontban mennyi a hibafelület gradiense, és a következő lépés paraméterkészletét ennek ismeretében kell kialakítani.

Egy lehetséges megközelítés a legmeredekebb lejtő módszere, amelyben a legmeredekebb irányba, a negatív gradiens mentén mozdulunk el az ún. bátorsági tényező által megszabott mértékben. Ennek alkalmazásával elkerülhetjük a bonyolult számítások elvégzését, valamint kiküszöbölhetjük a statisztikai adatok előzetes ismeretének hiányát is.

11.3.2. Adaptív jelfeldolgozó algoritmusok

Az LMS algoritmus

A statisztikai jellemzők megkövetelt ismeretét kiküszöbölhetjük, valamint a bonyolult számításokat is elkerülhetjük, ha a költségfüggvényben az átlagos hiba helyett csak a pillanatnyi hibát vesszük figyelembe, és a legmeredekebb lejtő módszerét eszerint módosítjuk. Így jár el az LMS[1] algoritmus a modellparaméterek adaptálásakor. Az $\mathbf{R}$ autokorrelációs mátrix, valamint a $\mathbf{p}^T$ keresztkorrelációs

[1] Least Mean Squares

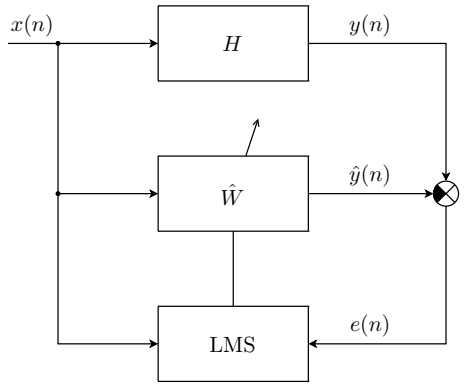
vektor tehát a következőképpen módosul:

$$\mathbf{R} = \mathrm{E}\{\mathbf{x}(n)\mathbf{x}^T(n)\} \text{ helyett: } \hat{\mathbf{R}} = \mathbf{x}(n)\mathbf{x}^T(n), \tag{11.11}$$

$$\mathbf{p}^T = \mathrm{E}\{y(n)\mathbf{x}^T(n)\} \text{ helyett: } \hat{\mathbf{p}}^T = y(n)\mathbf{x}^T(n). \tag{11.12}$$

A pillanatnyi hibán alapuló becslés miatt azonban a paraméterbeállítás módosítása pontatlan lesz, nem feltétlenül a negatív gradiens irányába történik. Hosszabb időintervallumban azonban az egyes módosítások pontatlansága kiátlagolódik. A módszer alkalmazásával viszont a modell a becsült rendszer parametrikus változásait jobban képes követni. Az LMS algoritmus sajátossága, hogy a hibaminimum körül a pillanatnyi hiba kicsi, a hibafelület gradiense nulla körül van. Így a paraméterkészlet soha nem állandósul a minimális értéken, azt ugyanis nem is éri el, hanem annak egy kis környezetében lépésről-lépésre ugrál. Az LMS algoritmusban ezek alapján tehát a paramétertérben történő lépések mértékét, a *bátorsági tényezőt* célszerű kicsire megválasztani. Ezzel azonban az algoritmus beállási tulajdonságai romlanak. A μ bátorsági tényező optimális értéke az $\mathbf{R}$ autokorrelációs mátrix ismeretében meghatározható.

Az LMS algoritmust alkalmazó legegyszerűbb struktúra blokkvázlata a 11.1. ábrán látható.



11.1. ábra. Az LMS algoritmus blokkvázlata

Az LMS algoritmusban kiszámításra kerülő rekurzív egyenletek a hibára valamint az együtthatókra vonatkozóan a következőkre adódnak:

$$e(n) = y(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n), \tag{11.13}$$

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + 2\mu e(n)\mathbf{x}(n). \tag{11.14}$$

Az LMS algoritmus által használt modell, melynek paraméterei hangolásra kerülnek, egy véges impulzusválaszú (FIR) digitális szűrő, melynek impulzusválasza a $\mathbf{w}$ vektor. Az n-edik ütemben rendelkezésre álló szűrőegyüttható-készletet a $\hat{\mathbf{w}}(n)$ vektor adja meg. Az LMS algoritmus tehát az $\mathbf{R}$ autokorrelációs mátrix, valamint a $\mathbf{p}^T$ keresztkorrelációs vektor ismerete nélkül képes a $\mathbf{w}_{opt}$ együtthatókészletet megközelítő szűrőegyütthatókat előállítani.

Az NLMS algoritmus

Ha az LMS algoritmus stabilitási és beállási tulajdonságait egyaránt megfelelően jobbra illetve gyorsabbra kívánjuk beállítani, nyilvánvaló, hogy a két szempont szerint ellentétes irányba kellene

a bátorsági tényezőt hangolni. A problémára egy lehetséges megoldást kínál az NLMS[2] algoritmus. Az NLMS a bátorsági tényezőt a bemeneti jel alapján normálja, így jobb stabilitási tulajdonságokkal, adott feltételek mellett kb. kétszeresen gyorsabb beállással rendelkezik, valamint a lépések méretének változtatása miatt az LMS algoritmusnál tapasztalttól kisebb gradiens-zaj[3] jellemzi.

Az NLMS algoritmus rekurzív összefüggései a hibára valamint az együtthatókra vonatkozóan a következők:

$$e(n) = y(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n), \tag{11.15}$$

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \frac{\tilde{\mu}}{a + \mathbf{x}^T\mathbf{x}} e(n)\mathbf{x}(n). \tag{11.16}$$

Az a konstans funkciója az, hogy kis bemenetek esetén se válhasson a tört nevezője nullához közelivé, és így a lépésköz túl naggyá.

Az FxLMS algoritmus

Egyes gyakorlati alkalmazásokban az LMS algoritmus segítségével adaptált paraméterkészletű szűrő kimenetére – a rendszer fizikai felépítéséből adódóan – egy másik, nem egységnyi átvitelű tag is kapcsolódik, és a különbségi jel képzésekor ennek kimenete kerül felhasználásra. Ezekben az esetekben tehát nem lehet már feltételezni, hogy az adaptív szűrő kimenete az identifikált rendszer kimenetéből azonnal kivonható, és az így előállított különbségi jel azonnal felhasználható az adaptív algoritmusban az adaptív szűrő hangolására.

Másképp megfogalmazva, a blokkvázlatot tekintve, az adaptív szűrő kimenete és a különbségképző blokk közé egy átvitel iktatódik, melynek legnagyobb problémája az, hogy a frekvenciatartománybeli szűrésen kívül fázistolást, késleltetést realizál az adaptív szűrő kimenete és a különbségképző között, így könnyen instabilitást okozhat, ha a modellezendő fizikai rendszer késleltetése ettől kisebb vagy összességében nem elegendően nagy értékű. Az ily módon a beavatkozó jel útjába iktatott szűrőt a – főként az aktív zajcsökkentéssel foglalkozó – szakirodalomban „másodlagos útnak" nevezik (az „elsődleges út" a modellezendő rendszert jelenti).

Ha tehát egy adott bemeneti jelre az identifikálandó rendszer valamilyen késleltetés mellett reagál, az LMS algoritmus a bemenet és a kimenet megfigyelése alapján igyekszik a szűrőt a rendszerrel azonosra hangolni, nem számol viszont azzal, hogy egy beiktatott átvitel módosítja a szűrő kimenetét. Az ún. másodlagos átvitel hatása akkor eliminálódik, ha az adaptív szűrő az identifikált rendszer mellett a másodlagos átvitel inverzét is modellezi. Szélsőséges esetben, ha a beiktatott átvitel késleltetése nagyobb az eredeti rendszer késleltetésétől, ez a feladat nem valósítható meg. Ha a másodlagos átvitel bizonyos frekvenciákon jelentős csillapítást realizál, az a szabályozási hurok instabilitásához vezethet.

A probléma kiküszöbölésére több elvi megoldás is kínálkozik. Elméleti jelentőségű az az elgondolás, hogy a másodlagos átviteli úttal sorosan az inverzét elhelyezve annak hatása eliminálható. Ez a gyakorlatban nem feltétlenül valósítható meg, mert az átviteli függvény inverze nem biztos, hogy létezik [4]. A gyakorlatban alkalmazható megoldást az FxLMS algoritmus által megvalósított struktúra kínálja. E megközelítés szerint a másodlagos átvitel becslőjét kell elhelyezni az LMS algoritmus bemenetén oly módon, hogy az a referenciajelet szűrje, de az adaptálandó szűrő bemenetére ne legyen hatással. Ez a felépítés tekinthető meg a 11.2. ábrán. A rendszer lineáris volta alapján a másodlagos út átvitele az adaptív szűrő elé is képzelhető, így nyilvánvalóvá válik, miért szükséges

[2] Normalised LMS

[3] A pillanatnyi derivált a deriválthoz viszonyítva zajjal terhelt. $\frac{\partial \hat{\varepsilon}(n)}{\partial \mathbf{w}(n)} = \frac{\partial \varepsilon(n)}{\partial \mathbf{w}(n)} + \text{zaj}$. Ezt nevezzük gradiens-zajnak.

a referenciajelet is ezzel az átvitellel megszűrni. Ennek megvalósításához azonban szükséges a másodlagos átvitel megfelelő pontosságú ismerete. Az FxLMS algoritmus problémája tehát az, hogy a másodlagos átvitelt valamilyen módon identifikálni kell, hogy a becslője a rendszerbe beépíthető legyen.

Az FxLMS algoritmusban a hibajel a következőknek megfelelően áll elő:

$$e(n) = y(n) - \hat{y}_s(n), \tag{11.17}$$

ahol $\hat{y}_s(n)$ az $\hat{y}(n)$ jel S másodlagos átvitellel szűrt változata. Legyen C átvitelű blokk az S átvitel becslője, valamint $\mathbf{c}$ annak véges impulzusválasza (a vektor a FIR szűrőegyütthatókat tartalmazza)! Ekkor $e(n)$ felírható a következő módon:

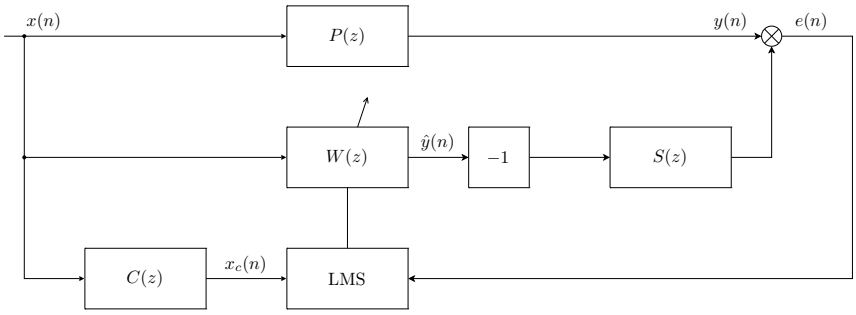
$$e(n) = y(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}_c(n), \text{ ahol} \tag{11.18}$$

$$\mathbf{x}_c(n) = \begin{bmatrix} \sum_{i=0}^{I-1} c_i x(n-i) \\ \sum_{i=0}^{I-1} c_i x(n-i-1) \\ \dots \\ \sum_{i=0}^{I-1} c_i x(n-i-(M-1)) \end{bmatrix}. \tag{11.19}$$

A $\mathbf{c} = [c_1, c_2, \dots, c_I]$ vektor a másodlagos átvitel becslőjének (véges) impulzusválasza, I pedig $\mathbf{c}$ hossza.

A szűrőegyütthatók adaptálására alkalmazható rekurzív egyenlet:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + 2\mu e(n)\mathbf{x}_c(n). \tag{11.20}$$



11.2. ábra. Az FxLMS algoritmus blokkvázlata

11.4. Az adaptív jelfeldolgozás módszerét alkalmazó struktúrák

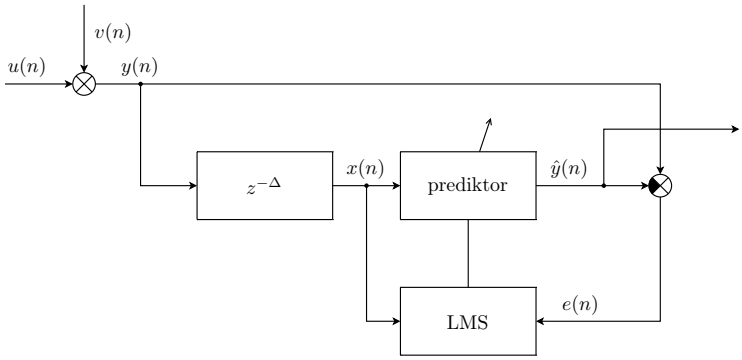
11.4.1. Adaptív vonaljavító

Az adaptív digitális jelfeldolgozás egyik fő területe a különféle zajok elnyomását tűzi ki célul. Egyik lehetséges felmerülő gyakori probléma például az információt hordozó jel kiszűrése az azt

terhelő additív zajból. Az adaptív vonaljavító, azaz ALE[4] egy ilyen feladatot megvalósító struktúra. Alkalmazására olyan esetekben lehet szükség, amikor a keskeny sávú hasznos jel sávszélessége és egyéb paraméterei időben változnak, míg kevés előzetes, *a priori* ismeret áll rendelkezésünkre róluk. Ilyen alkalmazási terület lehet például a szonár, bizonyos orvosi alkalmazások valamint különféle beszéddel kapcsolatos feldolgozások.

Az adaptív vonaljavító egy olyan speciális zajelnyomó struktúra, amely lehetővé teszi a bemenetét terhelő szélessávú zaj megfelelő mértékű csillapítását, elnyomását, míg a keskeny sávú, hasznos jelet hordozó bemeneti komponenst ideális esetben egységnyi erősítéssel, de legalábbis kis csillapítással továbbengedi. A számunkra hasznos jellel kapcsolatban az ALE működéséhez szükséges egyetlen teljesítendő feltétel, hogy a jel megfelelően keskeny sávszélességű legyen. Az adaptív jelfeldolgozás miatt nem szükséges, hogy a jel stacionárius tulajdonságú legyen.

Az ALE struktúráját tekintve egy késleltető valamint egy lineáris prediktor összekapcsolásából épül fel, amint az a 11.3. ábrán látható. Az ALE $y(n)$ bemeneti jele a keskenysávú, hasznos $u(n)$ jelből, valamint az ahhoz adódó $v(n)$ szélessávú zajból áll. A prediktor $\hat{y}(n)$ kimeneti jelét a $y(n)$ bemeneti jelből kivonva megkapjuk az $e(n)$ becslési hibát, melyet a prediktor adaptív hangolására használunk fel. A lineáris prediktort esetünkben egy FIR szűrő valósítja meg, melynek együtthatóit az LMS algoritmus hangolja.



11.3. ábra. Az ALE struktúra blokkvázlata

A prediktor $x(n)$ bemeneti jele a ALE bemenetének Δ számú mintával késleltetett változata:

$$x(n) = y(n - \Delta). \tag{11.21}$$

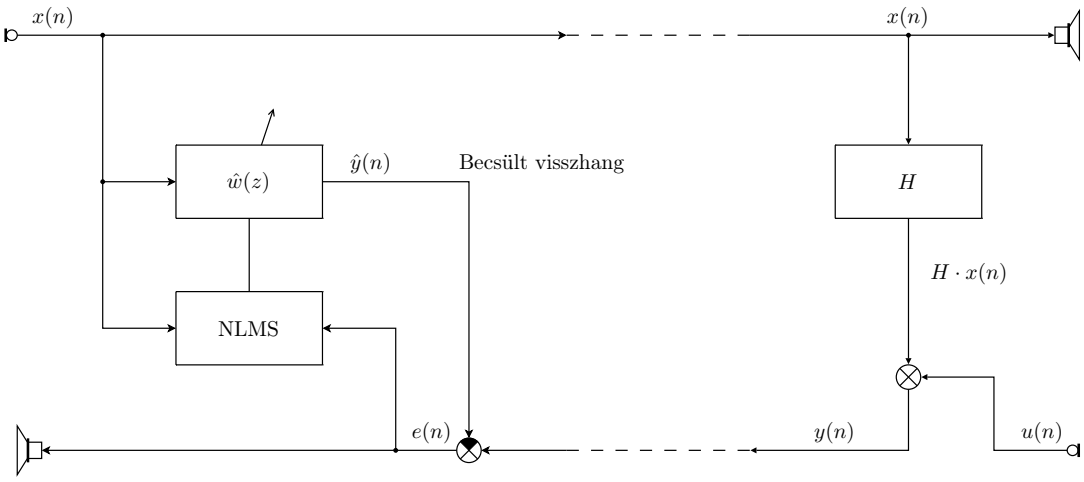
A struktúra Δ késleltetésének mértékét oly módon kell az implementáció során megválasztani, hogy az $y(n)$ bemeneti jel $v(n)$ zajkomponense, valamint annak $v(n-\Delta) \ldots v(n-\Delta-M)$ késleltetett változata korrelálatlan legyen (M a FIR szűrő hosszát jelöli). Ily módon a becslési hiba képzésekor a bemenet zajkomponense, valamint az adaptív szűrő kimenetének a bejövő zajból származó komponense korrelálatlan lesz. Ennek hatására a szűrő hangolásakor nem fordul elő az, hogy az adaptív algoritmus a szűrőegyütthatókat olyan irányba állítja, hogy a szűrő a zajt is átengedje annak érdekében, hogy a különbségképzéskor a becslési hibát még jobban minimalizálhassa. Mivel a struktúra kimeneti jele az $\hat{y}(n)$ jel, az ALE működése akkor jó, ha ez minél kevesebb zajt tartalmaz. Ha tehát a Δ ún. dekorreláló késleltetés értéke megfelelő, lehetővé válik a zaj kioltása, ugyanis azt a szűrő nem engedi át, miközben az időben változó, hasznos jelösszetevőt nem próbálja meg elnyomni.

[4] Adaptive Line Enhancer

11.4.2. Adaptív visszhangcsökkentés

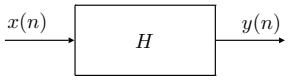
A visszhang problémája legszélesebb körben elsőként a PSTN[5] telefonhálózatokban jelentkezett. A kapcsolóközpontokban elhelyezett ún. hibridek közötti nagy távolság és a tökéletlen illesztés miatt reflexió lép fel, amely a nagy késleltetés miatt hallható visszhangot képes okozni, amely bizonyos határ fölött zavarja a beszélőket, vagy akár a beszédérthetőséget is jelentősen ronthatja.

Az adaptív visszhangcsökkentő struktúra modellje a 11.4. ábrán látható. Az ábra bal oldalán látható hangszóró-mikrofon pár az egyik, míg a jobb oldalon látható pár a másik személyhez tartozó végpont. A szaggatott vonalak a hosszú átviteli utat szemléltetik, de a visszhang szempontjából az ezek által okozott késleltetést beleértjük a H átvitelbe, amely a hallgató oldalán elhelyezkedő hibrid átvitelét jelöli.



11.4. ábra. Az adaptív visszhangcsökkentő struktúra blokkvázlata

A beszélő akkor hallja meg saját visszhangját, ha $H \neq 0$. Tekintsük azt az esetet, amikor a hallgató csendben van (ekkor $u(n) = 0$). Ebben az esetben a hallgató oldaláról $y(n) = H \cdot x(n)$ visszhang indul vissza a beszélő felé. A 11.5. ábrán a 11.4. ábra jobb oldalát láthatjuk erre az esetre vonatkozóan újrarajzolva. Ebből már látható, hogy az adaptív visszhangcsökkentés nem más, mint a hallgató oldalán elhelyezkedő hibrid átvitelének rendszeridentifikációs feladata.



11.5. ábra. Ha $u(n) = 0$, a hallgatóoldali hibridet identifikáljuk

Ha a visszhangot alkalmas módon becsülni tudjuk, az $\hat{y}(n)$ becsült visszhangjelet kivonhatjuk a beszélő oldalára visszaérkező $y(n)$ jelből. A két jel különbsége a becslési hibajel: $e(n)$. A becslés az $e(n)$ hibajelet és a beszélő $x(n)$ jelét felhasználó, a 11.3.2. pontban ismertetett NLMS algoritmussal történik.

Az adaptációt nehezíti, ha a hallgató a beszélővel egyidejűleg beszél a telefonba, azaz ha $u(n) \neq 0$. Az ilyen esetekben jelentkező problémák kiküszöbölésére az adaptációt az „összebeszélés” idejére fel kell függeszteni.

[5]Public Switched Telephone Network

11.4.3. Aktív zajcsökkentés

Az aktív zajcsökkentés feladata a környezetből, vagy valamilyen zajforrásból érkező zajnak ellenfázisú zajjal történő kioltása a tér adott kiválasztott környezetében. A passzív hanggátló struktúrákkal szemben a kitűzött cél elérésének érdekében arra alkalmas helyeken ún. beavatkozó hangszórók elhelyezésére van szükség. A beavatkozó hangszórók segítségével kiadott ellenfázisú zajjal a hibamikrofonok bizonyos környezetében csendes zónák alakíthatók ki. Az akusztikai rendszerek nagy dinamikatartományban jó közelítéssel lineárisnak tekinthetők, így a szuperpozíció elve eredményesen alkalmazható.

Az aktív zajcsökkentő rendszereknek akkor van létjogosultsága, ha a kioltandó zaj alacsony frekvenciájú, és a passzív hanggátlás már kis hatékonyságú volna, vagy abban az esetben, ha a mechanikai struktúra nem teszi lehetővé passzív hanggátlás költséghatékony telepítését. Ez utóbbi eset vonatkozik például a csövekben terjedő zavarokra, amely probléma a mérés során konkrétan is bemutatásra kerül.

Az aktív zajcsökkentés az akusztikai alkalmazásokon kívül mechanikai rezgések csillapítására is eredményesen alkalmazható, a megfelelő szenzorok és beavatkozók ebben az esetben is rendelkezésre állnak (például gyorsulásérzékelők és rázóasztalok).

Az aktív zajcsökkentő rendszerekben többféle struktúrát meg lehet valósítani. A jó stabilitási tulajdonságokkal és jelentősebb elnyomással jellemezhető struktúrák mindegyikében szükség van azonban a beavatkozó hangszórók és a mikrofonok közötti akusztikai átvitel identifikációjára, amelyre egy lehetséges módszer az LMS algoritmus alkalmazása.

Az adaptív jelfeldolgozó algoritmust alkalmazó aktív zajcsökkentő struktúrák közül az FxLMS algoritmuson alapulót mutatjuk be. Az FxLMS algoritmus a 11.3.2. pontban került ismertetésre.

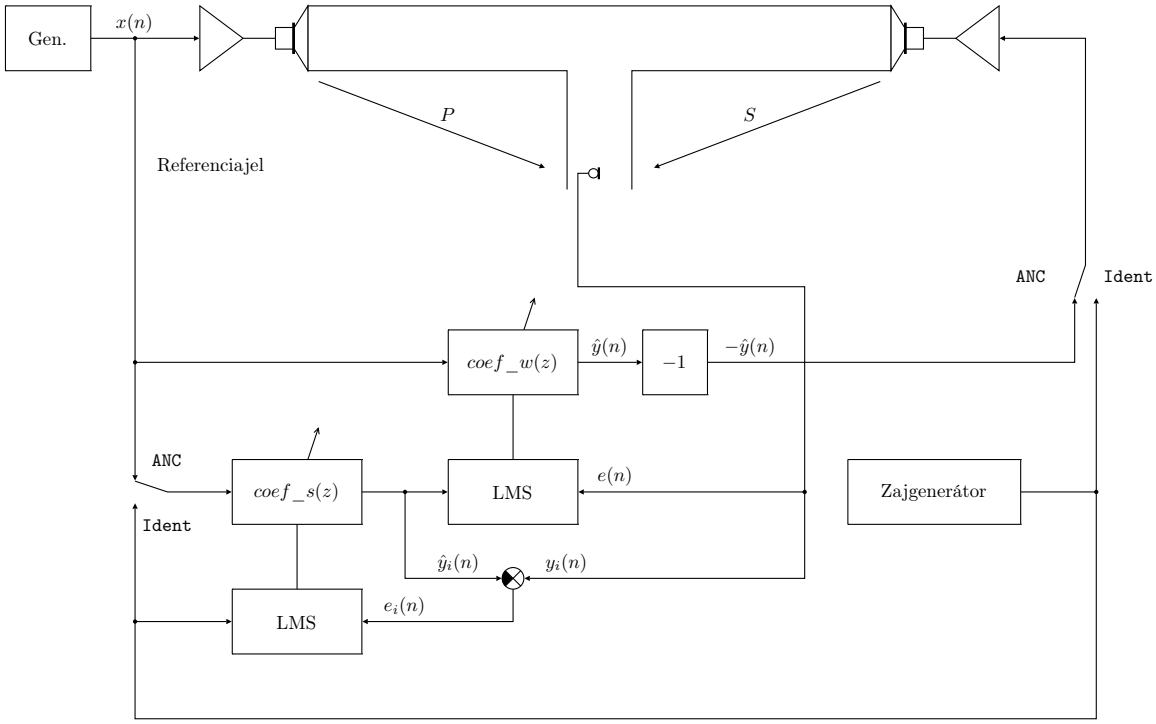
A 11.6. ábrán egy komplex FxLMS algoritmust alkalmazó aktív zajcsökkentő struktúra blokkvázlata látható, amely csövekben terjedő zaj kioltását végzi. A cső egyik végén található a zajforrás, melyet egy hangszóró modellez, míg a másik oldalán a beavatkozó hangszóró került elhelyezésre. A cső leágazásában található mérőmikrofon jele a cső szája által lesugárzott zajjal arányos. A struktúra a gyakorlatban jól alkalmazható ipari klímaberendezések zajának csökkentésére.

A zajforrástól a hibamikrofonig terjedő P átvitel az elsődleges út (*primary path*), míg a beavatkozó hangszórótól a hibamikrofonig terjedő S átvitel a másodlagos út (*secondary path*), a szakirodalomban alkalmazott terminológiát követve. Aktív zajcsökkentés üzemmódban a kapcsolókat `ANC`, míg a másodlagos út identifikációja során `Ident` állásban kell az ábrán tekinteni.

Példánkban az elnyomandó zajt egy generátor állítja elő, melyet azonnal felhasználunk az $x(n)$ referenciajel formájában. Az $x(n)$ jel a $coef\_w(z)$ adaptív szűrőre kerül, melynek $\hat{y}(n)$ kimenete – fázisfordítás után – a beavatkozó hangszóróra kerül. A fázisfordításra azért van szükség, mert az akusztikus térben összegzés valósul meg. A félreértések elkerülése érdekében már a 3.2. ábrát is ennek megfelelően alakítottuk ki.

A $coef\_w(z)$ szűrőt olyan adaptív algoritmus hangolja, mely hozzáfér a hibamikrofon $e(n)$ jeléhez, valamint az $x(n)$ referenciajel $coef\_s(z)$ átvitellel szűrt változatához. A $coef\_s(z)$ adaptív szűrőben a zajcsökkentéskor az előzetesen identifikált S átvitel becslőjének megfelelő együtthatókészlet található.

A másodlagos átviteli út identifikálása során a $coef\_s(z)$ szűrő együtthatóit kell az S átvitel fehér zajjal történő gerjesztése és a mikrofon jelének mérése alapján hangolni. A kapcsolók `Ident` állásában a zajgenerátor jele a beavatkozó hangszóróra, valamint a $coef\_s(z)$ szűrő bemenetére, és az őt hangoló LMS algoritmus referenciajel-bemenetére kerül. Az $e_i(n)$ hibajel a mikrofon $y_i(n)$



11.6. ábra. Az FxLMS algoritmus megvalósítása aktív zajcsökkentő rendszerben

jelének, valamint a $coef\_s(z)$ szűrő $\hat{y}_i(n)$ kimeneti jelének különbségeként áll elő:

$$e_i(n) = y_i(n) - \hat{y}_i(n). \tag{11.22}$$

A zajcsökkentési és identifikációs módokban egyes jelekre – az ábrán is látható módon – kétféle jelölést is alkalmaztam, amely az algoritmusok működésének megértését segíti elő.

11.5. A jelfeldolgozó kártya

11.5.1. A jelfeldolgozó processzorok

A DSP-k[6], azaz a digitális jelfeldolgozó processzorok széles körben elterjedtek a beágyazott rendszerek területén. Megtalálhatók mobiltelefonokban, hang- és video, DVD lejátszókban, digitális fényképezőgépekben és kamerákban, valamint számos olyan ipari alkalmazásban, ahol jelekkel vagy adatfolyamokkal kell dolgozni. Az általános célú processzorokkal szemben a DSP-k valós idejű jelfeldolgozási alkalmazásokra kerülnek optimalizálásra mind architekturális szinten, mina az utasításkészletük kialakítása terén. Úgy alakítják ki őket, hogy a hatékony jelfeldolgozást a lehető legkevesebb utasítás végrehajtása mellett lehessen használatukkal elvégezni.

11.5.2. Az ADSP-21364 EZ-KIT Lite fejlesztőkártya

Az ADSP-21364 EZ-KIT Lite fejlesztőkártya az ADSP-21364 jelfeldolgozó processzorral történő fejlesztés hatékony és gyors eszköze. A fejlesztőkártyán kialakításra került a jelfeldolgozó processzor minden olyan szükséges perifériája és környezete, amelyre a leggyakoribb feladatok elvégzésekor

[6]Digital Signal Processor

szükség lehet. Az analóg és digitális hang ki- és bemenetek, LED-ek, kapcsolók, nyomógombok azonnal hozzáférhetőek, így a szoftverfejlesztés mindenféle hardvertervezési lépés nélkül azonnal elkezdhető. A fejlesztőkártya tervezésekor igyekeztek a lehető legtöbb felhasználási lehetőséget kipróbálhatóvá, a processzor által nyújtott funkcionalitást pedig a lehető legmélyebben megismerhetővé tenni. A kártya a következő hardverelemeket bocsátja a fejlesztő rendelkezésére:

- Analog Devices ADSP-21364 processzor
- 512 kbit × 8 bit SRAM[7]
- 1 Mbit × 8 bit flash memória
- 2 Mbit SPI[8] által kezelhető flash memória
- Analóg hanginterfész (AD1835A kodek, 4 db sztereó bemenet, 1 db sztereó kimenet)
- Digitális hanginterfész (1 db bemenet, 1 db kimenet)
- 11 db LED (ebből 1 db „power”, 1 db „board reset”, 1 db „USB monitor”, 8 db általános célú)
- 5 db nyomógomb (1 db „reset”, 2 db DAI lábra, 2 db FLAG lábra kötve)
- Bővítőinterfész (párhuzamos port, FLAG-ek, DAI, SPI)
- JTAG emulátor port
- USB port a PC csatlakoztatásához.

A kártya képe a 11.7. ábrán, a csatlakozók oldalnézeti képe pedig a 11.8. ábrán látható.

11.6. A mérési feladatok

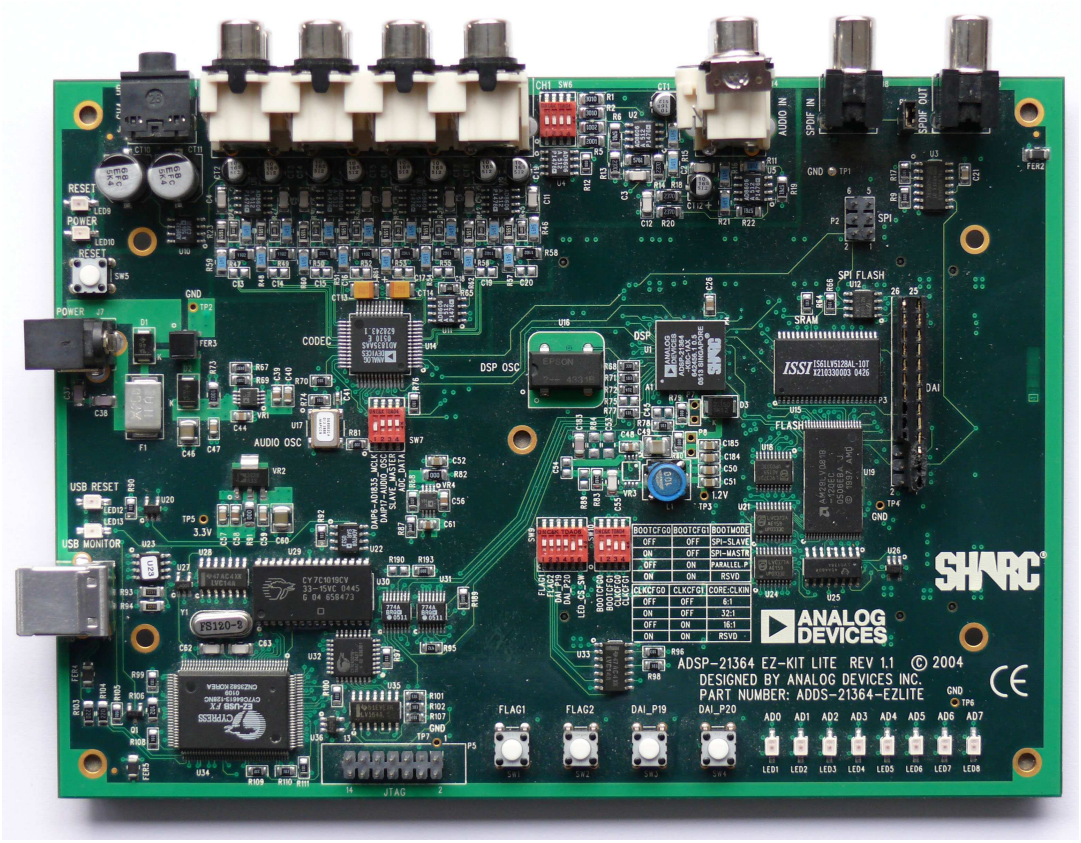
11.6.1. Az LMS algoritmus

Az LMS algoritmus megvalósítása

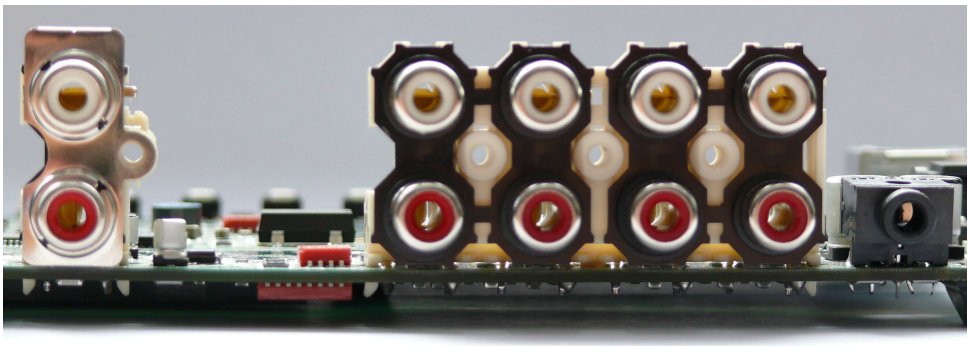
Az elméleti áttekintésben szereplő (11.13 és 11.14) egyenleteknek megfelelően írjon MATLAB függvényt az LMS algoritmus 11.9. ábra szerinti rendszeridentifikációs feladatot ellátó megvalósítására a következőknek megfelelően!

```
1  function [e,w]=lms(mu,M,x,y);
2
3  %   Bemeneti paraméterek:
4  %
5  %   mu = lépésköz-paraméter (bátorsági tényező) [1x1]
6  %   M = a szűrő hossza [1x1]
7  %   x = a rendszer bemeneti jele [Nx1]
8  %   y = a rendszer kimeneti jele [Nx1]
9
10 %   Kimeneti változók:
11 %
12 %   e = becslési hiba [Nx1]
```

[7] Synchronous Random Access Memory
[8] Serial Peripheral Interface



11.7. ábra. Az ADSP-21364 jelfeldolgozó kártya képe



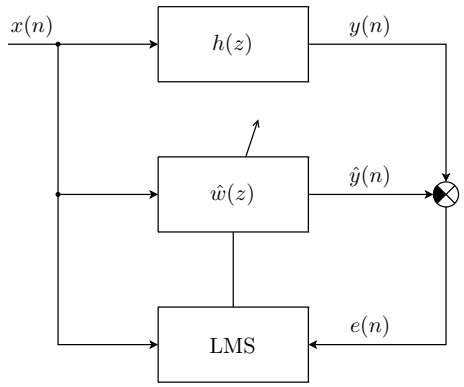
11.8. ábra. Az ADSP-21364 jelfeldolgozó kártya csatlakozóinak képe

```
% w = végső szűrőegyüttható-készlet [Mx1]
```

LMS algoritmust realizáló MATLAB függvény váza

A megvalósítandó rendszeridentifikációs alkalmazás a lehető legegyszerűbb struktúra az LMS algoritmus üzemeltetésére. Célja, hogy egy ismeretlen **h** átvitelű rendszert gerjesztve, a bemeneti és kimeneti jelének megfigyelése révén olyan adott hosszúságú FIR szűrő együtthatókészlete álljon elő, amelynek átvitele a **h**-t legjobban közelíti.

A MATLAB függvény az identifikálandó, ismeretlen rendszer rendelkezésre álló N hosszú $x(n)$ bemeneti és $y(n)$ kimeneti jeléből számítsa ki az $e(n)$ pillanatnyi hiba értékét minden ütemben,



11.9. ábra. Az LMS algoritmus segítségével adaptált szűrő blokkvázlata

valamint a megfelelő számú iteráció során állítsa elő a becsült M darab szűrőegyütthatót (11.13) és (11.14) alapján!

Segítség: a bemeneti vektorból történő $n \ldots n-(M-1)$ indexű M darab mintavett érték kiválasztására a MATLAB-ban használja az `xx = x(n:-1:n-M+1)` parancsot, amely a következőnek felel meg:

$$\mathbf{xx}(n) = [x(n), x(n-1), \ldots, x(n-(M-1))]. \tag{11.23}$$

A megvalósított algoritmus ellenőrzése

Ellenőrizze le az elkészített algoritmus működését egy ismert átvitelű rendszernek az LMS algoritmus felhasználásával történő identifikálásával és a kapott **w** szűrőparaméterek **h**-val történő összevetésével!

1. Állítson elő egy FIR szűrőt reprezentáló, néhány elemű impulzusválasz-vektort a MATLAB-ban! Ez lesz az identifikálandó, de ismert átvitelű rendszer.
2. Gerjesztőjelnek készítsen egy néhány ezer elemű Gauss eloszlású értékeket tartalmazó zajvektort!
3. Állítsa elő a szűrő kimeneti jelét! (Konvolválja a gerjesztőjelet az előbb előállított szűrő impulzusválaszával!)
4. Futtassa le az előző feladatban elkészített LMS algoritmust ezekkel a bemenő adatokkal! Ügyeljen rá, hogy M értéke egyezzen meg az ismert rendszert megvalósító szűrő hosszával az eredmények egyszerű összevetése érdekében! A μ értéke legyen kezdetben 0.1!
5. Ábrázolja az $e(n)$ becslési hibát a diszkrét idő függvényében, vizsgálja meg annak beállási tulajdonságait! Ha az algoritmus nem stabil, válasszon kisebb μ-t!
6. Próbáljon ki különféle μ értékeket, és vizsgálja a rendszer beállási és stabilitási tulajdonságait!

11.6.2. Adaptív vonaljavító (ALE)

A zavarok hatásainak kiküszöbölése

Az 5.2. ábrán látható ALE struktúra $y(n)$ bemenő jele a gyakorlati alkalmazásokban vezetéken, analóg átvitel révén kerül a jelfeldolgozó eszközbe történő bevezetésre. Ennek következményeképpen

az $y(n)$ jel additív zajjal és zavarokkal terhelt. A megvalósított rendszerben az $y(n)$ jelet a PC állítja elő és a hangkártya közvetítésével kerül a DSP kártya bemenetére. A bemeneti jelben jelentős zavarkomponens származhat az 50 Hz-es hálózatból.

Gondolja át, miért szükséges az esetleges 50 Hz-es zavarkomponenst még az ALE struktúrát megelőzően kiszűrni! Milyen működési jelenségeket okozna ennek elmulasztása?

Az $y(n)$ jelet terhelő kisfrekvenciás zaj valamint a hálózatból származó zavarok kiszűrésére tehát egy felüláteresztő szűrőt kell iktatni az ALE struktúra elé, a DSP-ben történő megvalósítással. A szűrőt a MATLAB segítségével kell megtervezni.

A MATLAB Filter Design & Analysis Tool eszköze segítségével tervezzen digitális szűrőt a fenti hatás kiküszöbölésére!

1. Adja ki MATLAB-ban az `fdatool` parancsot!
2. A megnyíló grafikus felhasználói felület segítségével tervezzen 50 tap-es véges impulzusválaszú (FIR) egyenletes ingadozású felüláteresztő szűrőt, amely 50 Hz alatt jelentős elnyomással rendelkezik, de 150-200 Hz fölött már kis csillapítás jellemzi! A mintavételi frekvencia legyen 8 kHz.
3. Az amplitúdó- és fáziskarakterisztikán kívül vizsgálja meg a szűrő pólus-zérus elrendezését is! Miből látható a szűrő típusa és jellege?

Az ALE működésének vizsgálata

A 11.3. ábrán látható jelöléseket alkalmazva, a mérés során az ALE struktúra $y(n)$ bemeneti jelét a jelfeldolgozó kártyára kell kötni. A kártya kimenetein az $y(n)$, $\hat{y}(n)$, valamint az $e(n)$ jelek hallgathatók meg, illetve dolgozhatók fel:

- y: 1. kimenet bal és jobb csatornája
- $\hat{y}$: 2. kimenet bal és jobb csatornája
- e: 3. kimenet bal és jobb csatornája
- x: 4. kimenet bal és jobb csatornája.

A bemenet tehát az első kimeneten módosítás nélkül hozzáférhető, míg a negyedik kimeneten a késleltetett változata kerül előállításra. Az ALE struktúra kimeneti jele, azaz a becsült jel a második kimeneten, míg a hibajel a harmadik kimeneten mérhető.

1. Adja ki a `ale` parancsot MATLAB-ban a jelfeldolgozó kártya működtetésére szolgáló grafikus kezelői felület megnyitásához!
2. Kattintson az *Inicializálás és program-letöltés* gombra! Ezzel az ALE struktúrát megvalósító kód letöltődik a jelfeldolgozó kártyára.
3. Indítsa el a program futását a *Futtatás* gombbal!
4. Állítson elő fehér gaussi zajjal terhelt szinuszos jelet a függvénygenerátor és a zajgenerátor segítségével, és kösse a jelfeldolgozó kártya y bemenetére! Próbáljon ki több különböző frekvenciát!

5. Hallgassa meg az ALE különféle jeleit, majd vizsgálja meg azok spektrumát is! A szűrő a megfelelő frekvenciát engedi át? Mely jelek tartalmazzák a hasznos, és melyek a zajkomponenst?

6. Ismételje meg a feladatot néhány különböző frekvenciájú szinuszos jel alkalmazásával, és foglalja össze a tapasztalatokat! Változtassa a μ értékét is! Szükség esetén nullázza a szűrőegyütthatók értékét!

7. Figyelje a struktúra beállási tulajdonságait, mikor a frekvenciát ugrásszerűen megváltoztatja! Hallgassa meg és analizálja az ALE különféle jeleit, különös tekintettel a frekvenciaugrás utáni részre!

8. Ismételje meg a feladatot eltérő frekvenciákon és μ értékeken! Vizsgálja a beállási és stabilitási tulajdonságokat!

9. Állítson elő lineárisan változó frekvenciájú szinuszos jelet[9] MATLAB segítségével, majd adjon hozzá az előzőekhez hasonlóan zajt! A jel legyen fél perc hosszú, frekvenciája induljon 500 Hzről és 10 s alatt érje el az 1 kHz-et. Adja ki a jelet a hangkártyára és kösse a DSP kártyára!

10. Ismételten hallgassa meg az ALE különféle jeleit, majd vizsgálja meg azok spektrumát is! Változtassa μ értékét és vizsgálja a beállási tulajdonságokat! Keresse meg a legkisebb μ-t, ahol az algoritmus még követni képes a változó frekvenciájú bemeneti jelet!

11. Keverjen zenéhez szinuszos jelet zavarásként, és adja a DSP bemenetére! Ebben az esetben mely kimeneteken hallható a hasznos és zavaró jel?

11.6.3. Adaptív visszhangcsökkentés (AEC) – az NLMS algoritmus

A 11.4. ábrán látható adaptív algoritmust alkalmazó visszhangelnyomó struktúra célja konkrétabban a H hibrid átvitelének identifikációja. Ha a hallgató oldalán a mikrofonba nem beszélnek, tehát $u(n) = 0$, az $x(n)$ és $y(n$ jelek közötti átvitel H: $y(n) = H(z)x(n)$. A beszélő oldalán a w adaptív szűrő az $x(n)$ bemeneti jelből előállítja a $\hat{y}(n)$ becsült visszhangot. Az $e(n) = y(n) - \hat{y}(n)$ maradó visszhang képviseli a hibajelet az algoritmus számára. Az $e(n)$ tehát a visszhangnak az a része, melyet a rendszer képtelen elnyomni.

A későbbiekben össze fogjuk hasonlítani az LMS valamint az NLMS algoritmust ebben az alkalmazásban. Ehhez definiálni szükséges egy dB-ben értelmezett arányt reprezentáló mennyiséget. Az ERLE (Echo Return Loss Enhancement) a visszhangcsökkentő algoritmus jóságára jellemző, a diszkrét időtől függő mennyiség:

$$ERLE(n) = 10 \lg \frac{P_y(n)}{P_e(n)}. \tag{11.24}$$

Az $ERLE(n)$ tehát a hibrid kimeneti jelének és a maradó visszhang n-edik ütembeli pillanatnyi teljesítményének aránya. Minél nagyobb érték, annál jobb a visszhangcsökkentés. Az $ERLE(n)$ vizsgálatával az algoritmus beállási tulajdonságait is vizsgálhatjuk.

Modellezze a hallgató oldalán lévő hibrid viselkedését H átvitel a reflexióra nézve! A beszélő akkor hallja meg saját visszhangját, ha $H \neq 0$. Ebben az esetben a hallgató oldaláról y visszhang indul a beszélő felé, melyet ha alkalmas módon becsülni tudunk, a becsült visszhangjelet ($\hat{y}$) kivonhatjuk belőle. A becslés a hibajelen és a beszélő jelén alapuló adaptív algoritmussal történik.

[9] Az ilyen jelet az akusztikai tartományban kiváltott érzete alapján *chirp*-nek nevezték el.

Az $ERLE(n)$ tényező kiszámítása

Írjon MATLAB függvényt, amely az $y(n)$ és $e(n)$ jelsorozatokból előállítja az $ERLE(n)$ vektort! Mivel a visszhangelnyomási tényező ütemenként jelentős mértékben eltérhet, alkalmazzon a bemeneti jeleken a négyzetre emelés után megfelelő aluláteresztő szűrést! Elegendő a következő átvitelű szűrő alkalmazása:

$$W(z) = \frac{1}{z - 0,995}. \tag{11.25}$$

Az NLMS algoritmus megvalósítása

Az adaptív visszhangcsökkentő alkalmazásokban a gyakorlatban az LMS algoritmus egy módosított változatát, az NLMS-t (Normalizált LMS) alkalmazzák. Az NLMS a bátorsági tényezőt a bemeneti jel alapján normálja, így jobb stabilitási tulajdonságokkal, adott feltételek mellett kb. kétszeresen gyorsabb beállással rendelkezik, valamint a lépések méretének változtatása miatt kisebb gradienszaj jellemzi.

Az NLMS algoritmus rekurzív összefüggései a hibára valamint az együtthatókra vonatkozóan a következők:

$$e(n) = y(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n), \tag{11.26}$$

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \frac{\tilde{\mu}}{a + \mathbf{x}^T\mathbf{x}} e(n)\mathbf{x}(n). \tag{11.27}$$

Az a konstans funkciója az, hogy kis bemenetek esetén se válhasson a tört nevezője nullához közelivé, és így a lépésköz túl naggyá.

A fenti egyenleteknek ((11.26 és 11.27)) megfelelően írjon MATLAB függvényt az NLMS algoritmus megvalósítására a következőknek megfelelően!

```
1  function [e,w]=lms(mu,M,x,y,a);
2
3  %   Bemeneti paraméterek:
4  %
5  %   mu = lépésköz-paraméter (bátorsági tényező) [1x1]
6  %   M = a szűrő hossza [1x1]
7  %   x = a rendszer bemeneti jele [Nx1]
8  %   y = a rendszer kimeneti jele [Nx1]
9  %   s = korrekció [1x1]
10
11 %   Kimeneti változók:
12 %
13 %   e = becslési hiba [Nx1]
14 %   w = végső szűrőegyüttható-készlet [Mx1]
```

NLMS algoritmust realizáló MATLAB függvény váza

A visszhangcsökkentő struktúra vizsgálata

1. Töltse be a hibrid bemeneti és kimeneti jeleit reprezentáló, a munkakönyvtárban található regisztrátum-vektorokat! Ehhez adja ki a `load hibrid` parancsot, mely után az `x1` és `y1` vektorban rendelkezésre áll mindkét jel mintasorozata.

2. Futtassa le az LMS algoritmust $M = 128$ szűrőhossz mellett és keressen jó μ értéket! A rendszert jellemezze gyors beállás és stabilitás! Vizsgálja az $e(n)$ hibajel időfüggvényét! Jegyezze meg azt a bátorsági tényezőt, melynél a legjobb tulajdonságokat tapasztalta! Ökölszabályként elmondható, hogy az a μ érték az optimális, mely kb. fele akkora, mint a még éppen stabil algoritmus bátorsági tényezője.

3. Futtassa le az NLMS algoritmust és keresse meg a leggyorsabb beállást eredményező $\tilde{\mu}$-t, épp úgy, ahogy az előző feladatban! Az a korrekció értékét válassza $a = 0,01$ körülinek! Vizsgálja az $e(n)$ hibajelet!

4. Hasonlítsa össze a két esetben kapott $e(n)$ időfüggvényeket (a maradó visszhangot), és hallgassa is meg azokat!

5. Vizsgálja meg az $ERLE(n)$ időfüggvényt a két esetben, ábrázolja közös diagramon! Vonja le a következtetéseket az LMS és NLMS algoritmus tulajdonságaira vonatkozóan!

6. Készítse el az `x2` és `y2` vektorokba az eredeti bemeneti jel (`x1`) és kimeneti jel (`y1`) tizedét! Ezzel azt az esetet szimuláljuk, ha ugyanazt a rendszert tizedakkora feszültségű jellel gerjesztjük. Futtassa le az LMS algoritmust az eredeti μ bátorsági tényezővel a csökkentett amplitúdójú regisztrátum-vektorokra, és foglalja össze tapasztalatait az $e(n)$ hibajelek és $ERLE(n)$ függvények összehasonlításával!

7. Változtassa meg a μ értékét úgy, hogy az LMS algoritmus tulajdonságai egyezzenek meg az eredetivel a csökkentett amplitúdójú bemenet esetében is! Segítség: az LMS algoritmus a pillanatnyi hiba négyzetén alapuló hibakritérium szerint működik. Az amplitúdók csökkentése után a teljesítmény mekkora lett? Milyen korrekció szükséges a μ értékén ennek alapján?

8. Futtassa le az NLMS algoritmust az eredeti $\tilde{\mu}$ és a paraméterrel a csökkentett amplitúdójú regisztrátumokon! Hasonlítsa össze a beállási tulajdonságokat a korrigált bátorsági tényezőjű LMS-nél tapasztaltakkal! Foglalja össze tapasztalatait!

9. Vizsgálja meg az eredeti regisztrátumokon mindkét algoritmus esetében, hogy a szűrő hosszának megváltoztatása mellett milyen irányban és mértékben kell a bátorsági tényezőt változtatni a stabilitás és jó beállás megőrzésének érdekében!

11.6.4. Az FxLMS algoritmus

Az FxLMS algoritmus megvalósítása

1. Írjon MATLAB függvényt az FxLMS algoritmus megvalósítására! A feladat megoldásához egészítse ki az LMS algoritmus kódját a fenti egyenleteknek megfelelően! A másodlagos átvitelt reprezentáló véges impulzusválasz-vektort paraméterként várja a függvény! Az implementáció során állítsa elő az $x_c(n)$ vektort minden iterációban!

A megvalósított algoritmus ellenőrzése

Ellenőrizze le az elkészített algoritmus működését egy ismert átvitelű rendszernek az FxLMS algoritmus felhasználásával történő identifikálásával és a kapott szűrőparaméterek vizsgálatával!

1. Állítson elő egy FIR szűrőt reprezentáló impulzusválasz-vektort a MATLAB-ban! Ez lesz az identifikálandó, de ismert átvitelű rendszer. Fontos, hogy az identifikálandó rendszer, amely az elsődleges átviteli útnak felel meg, megfelelő mértékű, 50 ütem körüli késleltetést realizáljon.

2. Gerjesztőjelnek készítsen egy néhány ezer elemű Gauss eloszlású értékeket tartalmazó zajvektort! Konvolválja a gerjesztőjelet az előbb előállított szűrő impulzusválaszával, annak érdekében, hogy megkapja a szűrő kimeneti jelét!

3. Állítson elő egy másodlagos átvitelt reprezentáló FIR szűrőt! A másodlagos átvitel realizáljon kb. fele akkora késleltetést mint az elsődleges átviteli út!

4. Futtassa le az előző feladatban elkészített FxLMS algoritmust ezekkel a bemenő adatokkal! A μ értéke legyen 0.01! Válasszon megfelelően nagy szűrőhosszt, de ügyeljen arra, hogy a futás stabil legyen!

5. Ábrázolja az $e(n)$ becslési hibát a diszkrét idő függvényében, vizsgálja meg annak beállási tulajdonságait!

6. Az FxLMS algoritmus által adaptált szűrő a másodlagos átvitellel együtt modellezi az elsődleges utat. Ábrázolja az identifikálandó átvitel amplitúdókarakterisztikáját közös diagramon e két szűrő együttes átvitelével!

7. Próbáljon ki különféle μ értékeket és vizsgálja a rendszer beállási és stabilitási tulajdonságait!

11.6.5. Az aktív zajcsökkentő struktúra

A mérés során egy olyan aktív zajcsökkentő rendszer vizsgálatára nyílik lehetőség, amely csövekben terjedő zajok kioltását végzi FxLMS algoritmus segítségével. A gyakorlatban megvalósított rendszer teljes mértékben megfelel a 11.4.3. pontban ismertetett struktúrával.

A rendszer összeállítása és működtetése

Állítsa össze a rendszert! Kapcsoljon a teljesítményerősítőre szimmetrikus, ±22 V-os tápfeszültséget, és kösse kimeneteit a cső két végén található hangszórókhoz! Csatlakoztassa a mikrofont a mikrofonerősítő egyik csatornájához, és kapcsolja be! A mikrofonerősítő belső telepes táplálású.

Végezze el az összekötéseket a következőknek megfelelően! A referenciajelet külső generátorral állítsa elő, és adja a jelfeldolgozó kártya megfelelő bemenetére és az egyik hangszóróra egyaránt!

- Bemenetek:
  - 1. bemenet (fehér): referenciajel
    `ANC` módban: x
    `Ident` módban nincs használatban
  - 2. bemenet (piros): hibamikrofon jele
    `ANC` módban: e
    `Ident` módban: y_i.
- Kimenetek:

- 1. kimenet mindkét csatornája: beavatkozó hangszóró jele
  `ANC` módban: $-\hat{y}$
  `Ident` módban: fehér zaj
- 2. kimenet mindkét csatornája:
  `ANC` módban: hibajel (e)
  `Ident` módban nincs használatban.

Az identifikációs mód a `FLAG1 / SW1` gomb nyomvatartásával működtethető. Identifikációs módban a beavatkozó hangszóróból a jelfeldolgozó kártyán generált álvéletlen zaj hallható. Aktív zajcsökkentés üzemmódban a `FLAG2 / SW2` nyomógomb megnyomásával a $coef\_w(z)$ szűrő, míg a `DAI_P19 / SW3` nyomógombbal a $coef\_s(z)$ szűrő együtthatói nullázhatók. Az előbbi használatával a rendszer beállási tulajdonságai vizsgálhatók, míg a másodikkal a referenciajel vehető le az LMS algoritmus bemenetéről azáltal, hogy csupa nulla együtthatójú szűrővel kerül megszűrésre. Referenciajel hiányában a $coef\_w(z)$ együtthatói nem módosulnak.

A `DAI_P20 / SW4` nyomógomb segítségével a kimenet tiltható. A szűrőegyütthatók stabilitása megőrzésének érdekében letiltott kimenet esetén a $coef\_w(z)$ együtthatók adaptálása is felfüggesztésre kerül.

Az aktív zajcsökkentő struktúra vizsgálata

1. Adja ki az `anc` parancsot MATLAB-ban a jelfeldolgozó kártya működtetésére szolgáló grafikus kezelői felület megnyitásához!
2. Kattintson az *Inicializálás és program-letöltés* gombra! Ezzel az ANC struktúrát megvalósító kód letöltődik a jelfeldolgozó kártyára.
3. Indítsa el a program futását a *Futtatás* gombbal!
4. Végezzen identifikációt, majd figyelje a struktúra működését!
5. Állítsa be a mikrofon- és teljesítményerősítőt oly módon, hogy a jelszintek megfelelők legyenek! Ügyeljen arra, hogy beavatkozó hangszórónak elegendő teljesítményt le kell tudnia adni a zaj elnyomására!
6. Vizsgálja meg a struktúra beállási tulajdonságait a megfelelő szűrőegyütthatók nullázásával!
7. A grafikus felhasználói felület segítségével állítsa át a bátorsági tényezők értékét. Identifikáció esetén ügyeljen arra, hogy mérés közben ne legyen zaj a csőben! A beavatkozó hangszóró jelszintje legyen elegendően magas, és a bátorsági tényező ne legyen szélsőséges értékű!
8. Aktív zajcsökkentés üzemmódban vizsgálja meg, hogy módosulnak a struktúra stabilitási, beállási és követési tulajdonságai μ függvényében!
9. Próbáljon ki zajként különféle bemeneti jeleket!

12. fejezet

Mérésvezetői útmutató

12.1. Az LMS algoritmus vizsgálata

12.1.1. Az LMS algoritmus megvalósítása

A feladat megoldása a következő:

```
1  function [e,w]=lms(mu,M,x,y);
2
3  %   Bemeneti paraméterek:
4  %
5  %   mu = lépésköz-paraméter (bátorsági tényező) [1x1]
6  %   M = a szűrő hossza [1x1]
7  %   x = a rendszer bemeneti jele [Nx1]
8  %   y = a rendszer kimeneti jele [Nx1]
9
10 %   Kimeneti változók:
11 %
12 %   e = becslési hiba [N-Mx1]
13 %   w = végső szűrőegyüttható-készlet [Mx1]
14
15 %   zérus kezdeti értékek
16     w = zeros(M,1);
17
18 %   a bemeneti jel mintáinak darabszáma
19     N = length(x);
20
21 %   x és y oszlopvektorok
22     x = x(:);
23     y = y(:);
24
25 %   LMS algoritmus
26     for n = M:N
27         xx = x(n:-1:n-M+1);
28         e(n) = y(n) - w' * xx;
29         w = w + 2 * mu * xx * e(n);
30     end
31
32 %   e oszlopvektor
33     e = e(:);
```

LMS algoritmust realizáló MATLAB függvény

12.1.2. A megvalósított algoritmus ellenőrzése

A feladat megoldása a következő:

```
% FIR szűrő együtthatói
h = 0.1.^[0:4];
% bemeneti jel
x = randn(1000,1);
% kimeneti jel
y = conv(h,x);
% LMS algoritmus
[e,w] = lms(0.1,5,x,y);
% w vizsgálata
w
% hibajel ábrázolása
plot(e);
```

Az LMS algoritmus ellenőrzése

Az algoritmus által kiszámított együtthatókészlet helyes működés esetén pontosan megegyezik a **h** értékeivel, ugyanis **w** és **h** hossza a példában megegyezik, valamint az egyes jelek megfigyelése nem zajos.

A hibajel alakulása $\mu = 0,1$ esetén a 12.1. ábrán, $\mu = 0,16$ esetén pedig a 12.2. ábrán látható. Megfigyelhető, hogy az algoritmus megfelelő paraméterek melletti, megfelelően hosszú ideig történő futtatása után a hibajel egy viszonylag kis intervallumon belül marad. Ha az iterációk során mégis nőni kezdene, az algoritmus rövid időn belül újra a helyes irányba adaptálja a szűrőegyütthatókat, és a hibajel ismételten lecsökken. A bátorsági tényező növelésével az algoritmus beállási tulajdonságai eleinte javulnak, de túlzottan nagy μ választása esetén instabillá válik.

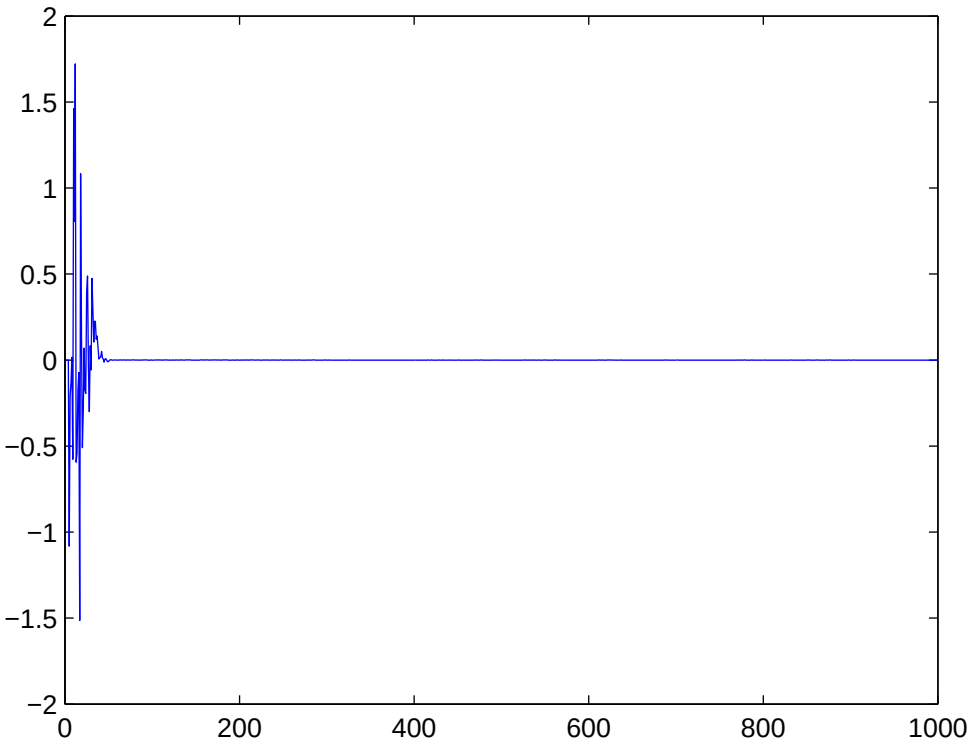
12.2. Adaptív vonaljavító (ALE)

12.2.1. A zavarok hatásainak kiküszöbölése

A felvetett probléma megoldása az, hogy a hálózatból származó 50 Hz-es zavarjel kis frekvenciájú és – mivel szinuszos – stacionárius tulajdonságú is. Az ALE struktúra adaptív algoritmussal hangolt szűrője nem szűrné ki, a kimeneti jelben változatlanul benne volna.

Az előírtaknak megfelelően megtervezett szűrő amplitúdó- és fázismenete a 12.3. ábrán látható. A szűrő egyenletes ingadozású és fázismenete lineáris. Ingadozása az áteresztő tartományban ± 1 dB, így kijelenthető, hogy itt jó közelítéssel alakhű átvitelt realizál.

A szűrő pólus-zérus képe a 12.4. ábrán látható. Megfigyelhető, hogy mivel a szűrő véges impulzusválaszú, pólusai nincsenek (vagy másként fogalmazva minden pólusa nulla), zérusai közül pedig találunk az $(1 + 0j)$ hely környezetében.



12.1. ábra. Az $e(n)$ hibajel alakulása az LMS algoritmus tesztelése közben, $\mu = 0,1$

12.2.2. Az ALE működésének vizsgálata

A rendszer működtetésére szolgáló MATLAB alapú grafikus felhasználói felület képe a 12.5. ábrán látható.

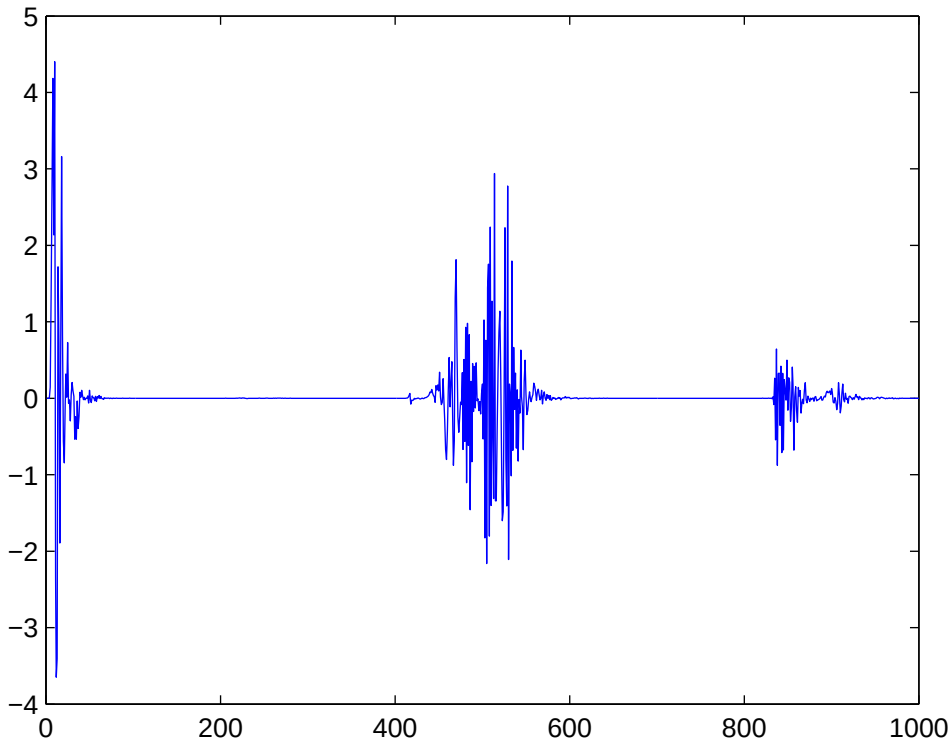
Ha az ALE bemenetére zajjal terhelt szinuszos jelet adunk, a kimeneten a szinusz-komponens változatlan amplitúdóval megjelenik, míg a zaj mindenütt kb 20-30 dB csillapodást szenved a bátorsági tényező értékétől függően. A frekvenciát úgy érdemes megválasztani, hogy az 100 Hz-nél nagyobb legyen, de ne érje el a decimált mintavételi frekvencia felét. A struktúra ugyanis 8 kHz-re decimált mintavételi ütemben végzi a feldolgozást. Ebben a tartományban azonban a kimeneti jel frekvenciától függetlenül a fentieknek megfelelően áll elő.

A struktúra bemeneti jele, az $y(n)$ hasznos jelet és zajt is tartalmaz, míg a szűrő $\hat{y}(n)$ kimenete megfelelő működés esetén a hasznos jelet változatlanul tartalmazza, de a zajkomponenst jelentősen csillapítja. Az $e(n)$ hibajel a két előbbi különbsége, azaz tartalma a maradó zaj.

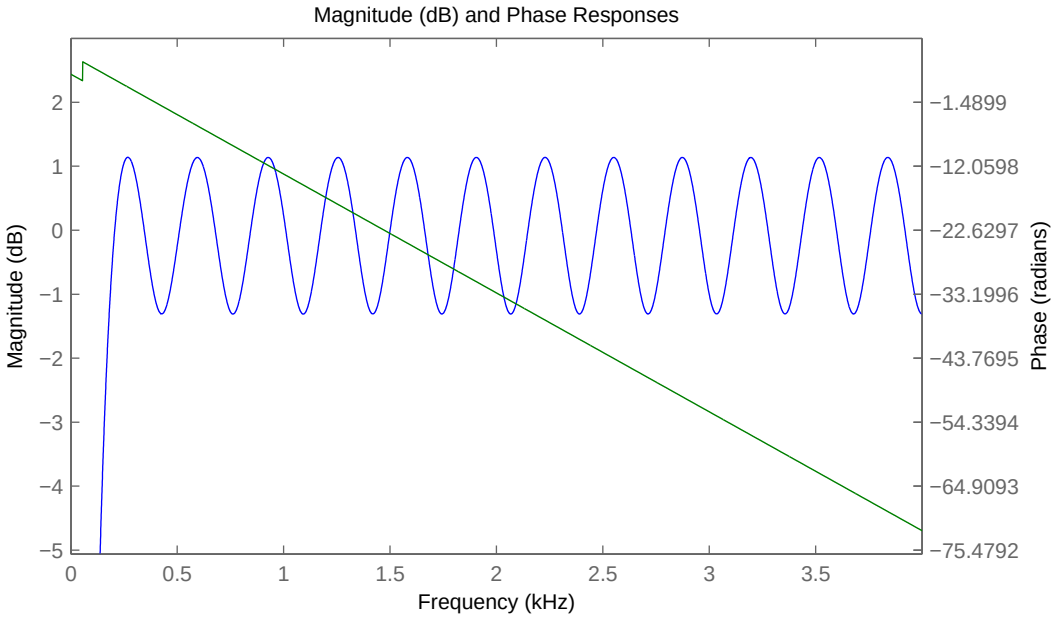
A struktúra beállási tulajdonságai a hasznos jel ki- és bekapcsolásával, illetve frekvenciájának módosításával vizsgálhatók. Nagyobb μ gyorsabb beállást de nagyobb maradó zajt eredményez. Túl nagy bátorsági tényező instabilitást eredményez.

Zajjal terhelt szinuszos jel előállítására szolgáló MATLAB kód:

```
1 fs = 8000;
2 f = 1000;
3 t = 30;
4 u = sin(2*pi*f*[0:1/fs:t]);
```

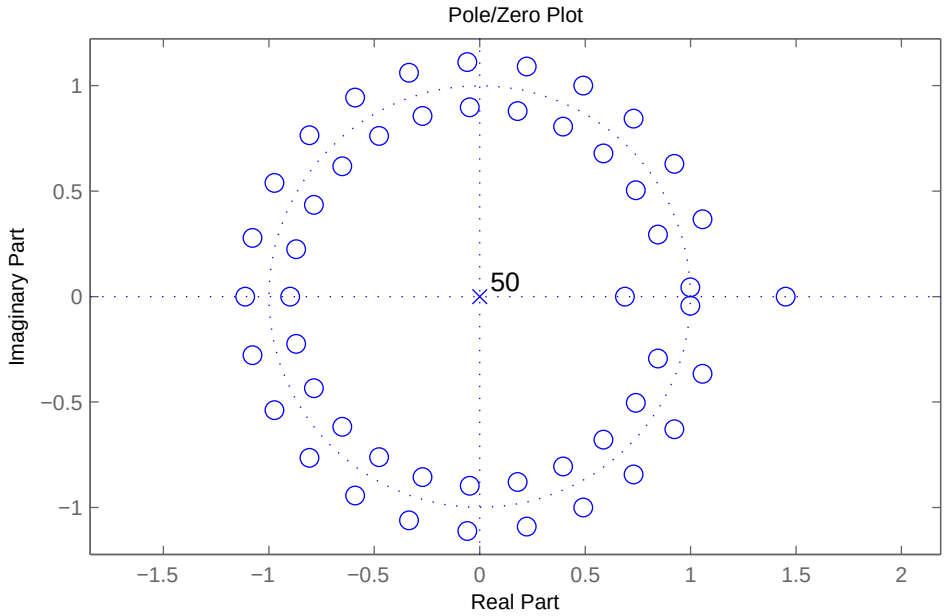


12.2. ábra. Az $e(n)$ hibajel alakulása az LMS algoritmus tesztelése közben, $\mu = 0,16$



12.3. ábra. A hálózati zavarok elnyomására szolgáló felülátereszt ő szűrő amplitúdó- és fáziskarakterisztikája

```
5 v = randn(size(u)) * 1.5; % nagyobb amplitúdó
6 y = u + v;
```



12.4. ábra. A hálózati zavarok elnyomására szolgáló felüláteresztő szűrő pólus-zérus képe

12.5. ábra. Az ALE rendszer felhasználói felülete

```
7 soundsc(y,fs);
```

Zajjal terhelt, egymás után következő két különböző frekvenciájú szinuszos jel előállítására szolgáló MATLAB kód:

```
1 fs = 8000;
2 f1 = 1000;
3 f2 = 2000;
4 t = 20;
5 u1 = sin(2*pi*f1*[0:1/fs:t]);
6 u2 = sin(2*pi*f2*[0:1/fs:t]);
7 u = [u1 u2];
8 v = randn(size(u)) * 1.5; % nagyobb amplitúdó
```

```
y = u + v;
soundsc(y,fs);
```

A chirp jel használatával a struktúra követési tulajdonságai vizsgálhatók. A chirp jel változási gyorsaságának és a bátorsági tényező értékének kapcsolatát meghallgatással lehet a legegyszerűbben vizsgálni. Füllel jól hallható, mely esetben képes a struktúra gyorsabb adaptációra, mint a hasznos jel frekvencia-változásának sebessége.

Zajjal terhelt chirp jel előállítására szolgáló MATLAB kód:

```
fs = 8000;
t = 30;
fstart = 500;
t1 = 10;
f1 = 1000;
u = chirp([0:1/fs:t], fstart, t1, f1);
v = randn(size(u)) * 1.5; % nagyobb amplitúdó
y = u + v;
soundsc(y,fs);
```

Zenéhez kevert szinuszos jel esetében a helyzet más – ekkor a zene a hasznos jel, a szinusz pedig a zavarás. A struktúra kimenetén az értékes jel ebben az esetben az $e(n)$ jel. Megjegyezzük, hogy a feldolgozás során alkalmazott decimált mintavételi frekvencia 8 kHz, így a zene erős sávkorlátozást szenved. Célszerű ezért meghallgatás útján az $y(n)$ és $\hat{y}(n)$ jeleket összevetni.

12.3. Adaptív visszhangcsökkentés (AEC) – az NLMS algoritmus

12.3.1. Az $ERLE(n)$ tényező kiszámítása

A feladat megoldása a következő:

```
function er = erle(y,e);
P_y = filter(1, [1,-0.995], y .^2);
P_e = filter(1, [1,-0.995], e .^2);
er = 10 * log10 (P_y ./ P_e);
```

12.3.2. Az NLMS algoritmus megvalósítása

A feladat megoldása a következő:

```
function [e,w]=nlms(mu,M,x,y,a);

%   Bemeneti paraméterek:
%
%   mu = lépésköz-paraméter (bátorsági tényező) [1x1]
```

```
6  %   M = a szűrő hossza [1x1]
7  %   x = a rendszer bemeneti jele [Nx1]
8  %   y = a rendszer kimeneti jele [Nx1]
9  %   a = korrekció [1x1]
10
11 %   Kimeneti változók:
12 %
13 %   e = becslési hiba [Nx1]
14 %   w = végső szűrőegyüttható-készlet [Mx1]
15
16 %   zérus kezdeti értékek
17     w = zeros(M,1);
18
19 %   a bemeneti jel mintáinak darabszáma
20     N = length(x);
21
22 %   x és y oszlopvektorok
23     x = x(:);
24     y = y(:);
25
26 %   LMS algoritmus
27     for n = M:N
28         xx = x(n:-1:n-M+1);
29         e(n) = y(n) - w' * xx;
30         w = w + (mu / (a + xx' * xx)) * xx * e(n);
31     end
32 %   e oszlopvektor
33     e = e(:);
```

NLMS algoritmust realizáló MATLAB függvény

12.3.3. A visszhangcsökkentő struktúra vizsgálata

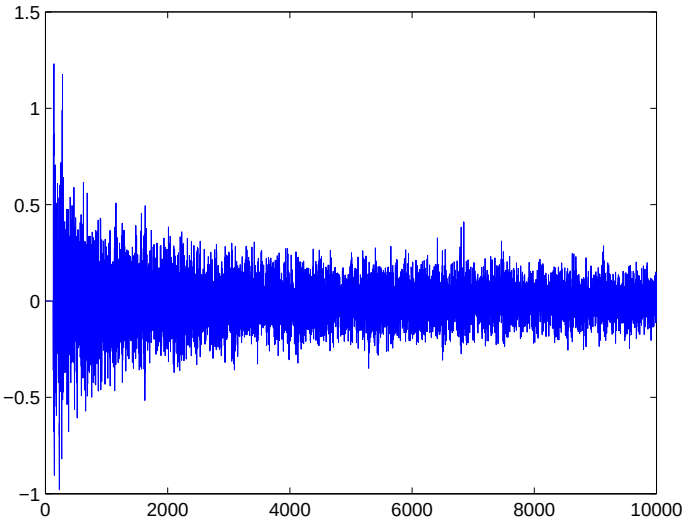
1. A regisztrátumok betöltése:

```
1 % a regisztrátumok betöltése:
2 % x1: a hibrid bemeneti jele
3 % y1: a hibrid kimeneti jele
4 load hibrid;
```

2. Az optimális bátorsági tényező értéke $\mu = 0,0001$ körül adódik. $\mu = 0,0002$ esetén az algoritmus még stabil, de tulajdonságai már rosszabbak. Az $e(n)$ grafikonja a 12.6. ábrán látható.

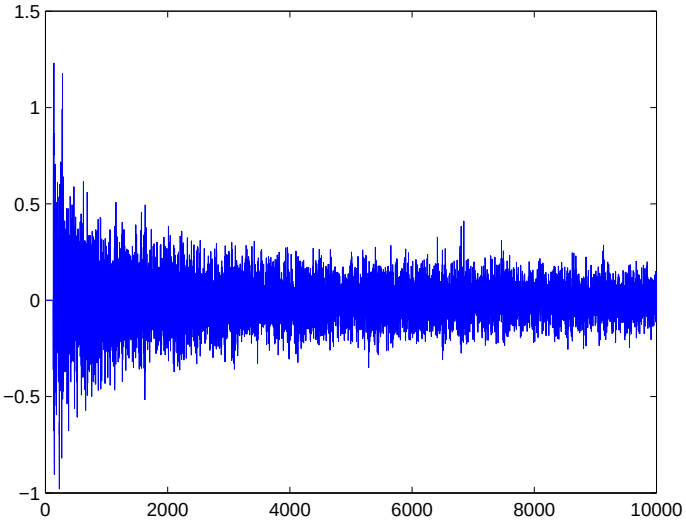
```
1 % az LMS vizsgálata, optimális mu keresése
2 [e_lms_1,w_lms_1] = lms(0.0001,128,x1,y1);
3 plot([1:length(y1)], e_lms_1);
```

3. Az optimális bátorsági tényező értéke $\tilde{\mu} = 1,2$ körül adódik. Az $e(n)$ grafikonja a 12.7. ábrán látható.



12.6. ábra. Az LMS algoritmus vizsgálata, $e(n)$, $\mu = 0,0001$

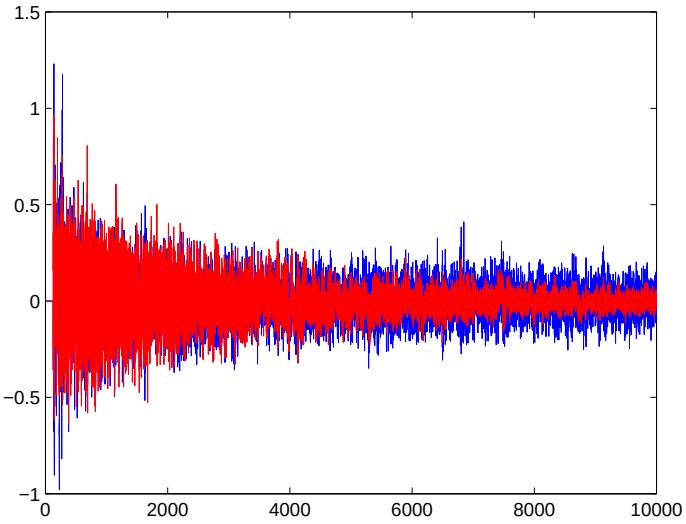
```
% az NLMS vizsgálata, optimális mu~ keresése
[e_nlms_1,w_nlms_1] = nlms(1.2,128,x1,y1,0.01);
plot([1:length(y1)], e_nlms_1);
```



12.7. ábra. Az NLMS algoritmus vizsgálata, $e(n)$, $\tilde{\mu} = 1,2$

4. A hibajelek összehasonlítása a 12.8. ábrán látható.

```
% a hibajelek összehasonlítása
plot([1:length(y1)],e_lms_1,'b',[1:length(y1)],e_nlms_1,'r');
```



12.8. ábra. Az $e(n)$ hibajelek összehasonlítása, kék: LMS, $\mu = 0,0001$, piros: NLMS, $\tilde{\mu} = 1,2$

5. Az $ERLE(n)$ függvények és az $e(n)$ hibajelek összehasonlításakor észrevehető, hogy az NLMS algoritmus segítségével kisebb maradó hiba érhető el, beállásának időállandója kisebb, de emiatt kezdetben a pillanatnyi bátorsági tényező magasabb, így ekkor nagyobb hiba jellemzi. Az $ERLE(n)$ függvények grafikonjai a 12.9. ábrán láthatók.

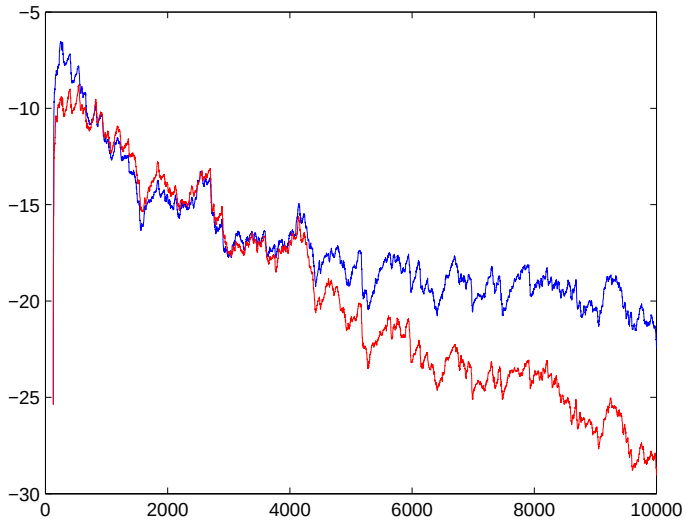
```
% az ERLE időfüggvények kiszámítása
erle_lms_1 = erle(e_lms_1,y1);
erle_nlms_1 = erle(e_nlms_1,y1);
plot([1:length(y1)],erle_lms_1,'b',[1:length(y1)],erle_nlms_1,'r');
```

6. Az $e(n)$ hibajelek vizsgálata nem sok eredményre vezet, ugyanis csak az látszik, hogy a csökkentett amplitúdójú esetben a hibajel amplitúdója is kisebb – amint az várható volt (12.6. ábra). Az $ERLE(n)$ 12.11. ábrán történő vizsgálatából azonban jól látszik, hogy ebben az esetben az algoritmus futásának tulajdonságai rosszabbak. Ennek oka, hogy a szűrőparaméterek hangolásának mértéke függ a bemeneti jel szintjétől.

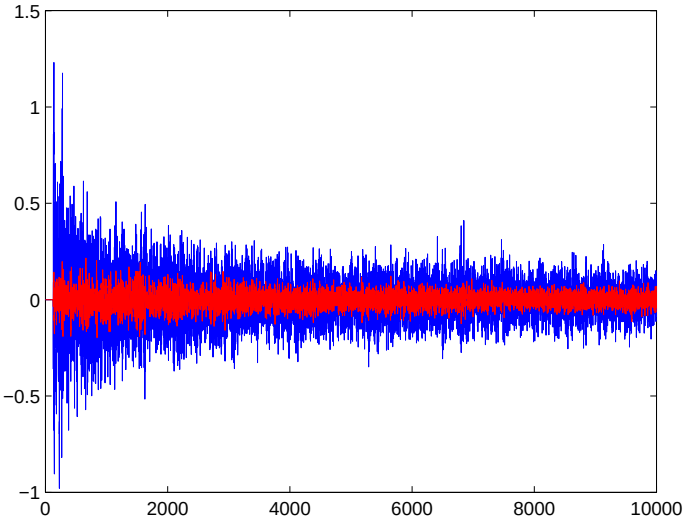
```
% csökkentett amplitúdójú regisztrátum-vektorok:
x2 = x1 / 10;
y2 = y1 / 10;

%  az LMS vizsgálata az eredeti mu értékkel
[e_lms_2,w_lms_2] = lms(0.0001,128,x2,y2);
plot([1:length(y1)],e_lms_1,'b',[1:length(y2)],e_lms_2,'r');
erle_lms_2 = erle(e_lms_2,y2);
plot([1:length(y1)],erle_lms_1,'b',[1:length(y2)],erle_lms_2,'r');
```

7. Tizedakkora amplitúdók mellett a teljesítmények századolódnak, így az eredetivel megegyező



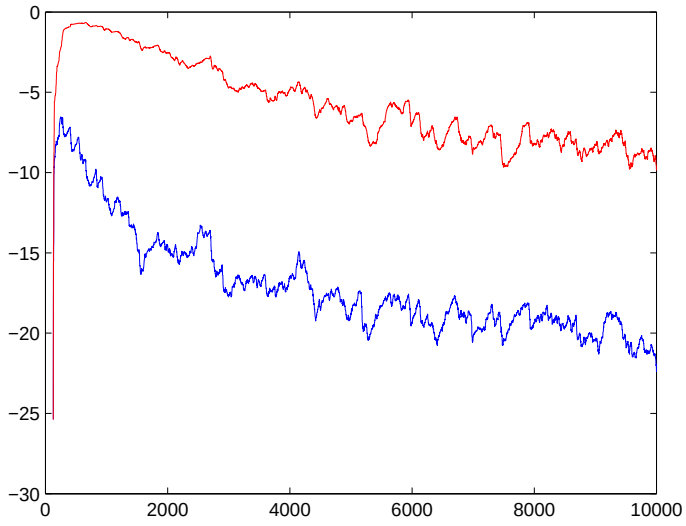
12.9. ábra. Az $ERLE(n)$ függvények összehasonlítása, kék: LMS, $\mu = 0,0001$, piros: NLMS, $\tilde{\mu} = 1,2$



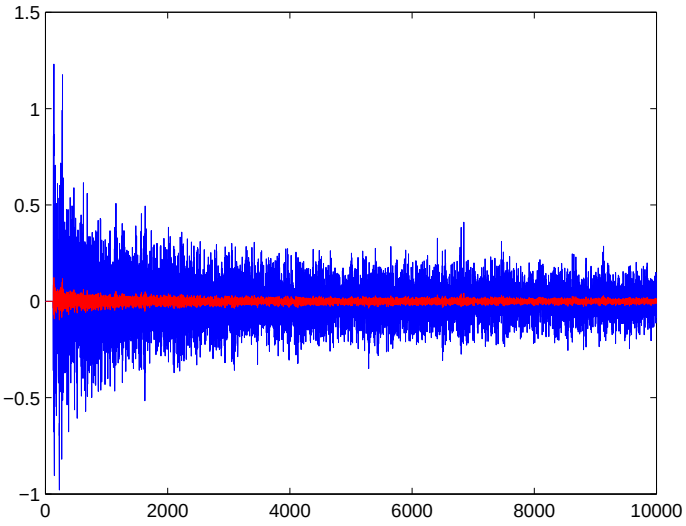
12.10. ábra. Az LMS algoritmus vizsgálata, $e(n)$, kék: eredeti bemeneti jel esete, piros: csökkentett amplitúdójú bemenet esete, $\mu = 0,0001$

működés eléréséhez $100 \cdot \mu = 100 \cdot 0,0001 = 0,01$ beállítása szükséges. Ebben az esetben a 12.13. ábrán látható $ERLE(n)$ függvények grafikonjai egybeesnek az eredetikkel. Az $e(n)$ hibajel természetesen kisebb marad, amint az a 12.12. ábrán is látszik.

```
% az LMS vizsgálata a megnövelt mu értékkel
[e_lms_2,w_lms_2] = lms(0.01,128,x2,y2);
plot([1:length(y1)],e_lms_1,'b',[1:length(y2)],e_lms_2,'r');
erle_lms_2 = erle(e_lms_2,y2);
plot([1:length(y1)],erle_lms_1,'b',[1:length(y2)],erle_lms_2,'r');
```



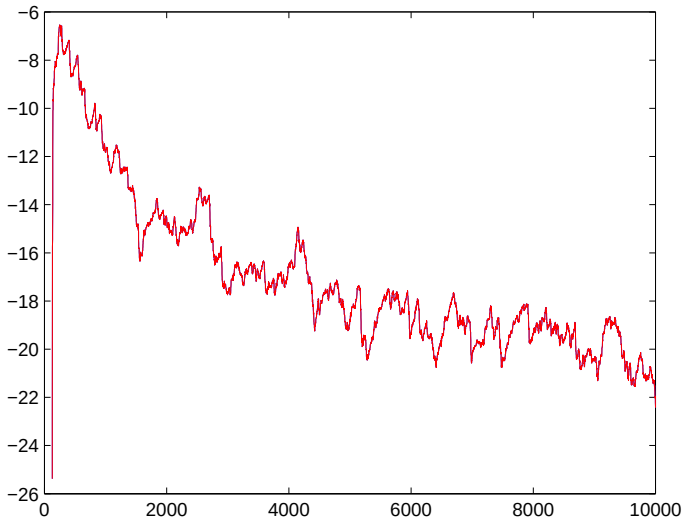
12.11. ábra. Az LMS algoritmus vizsgálata, $ERLE(n)$, kék: eredeti bemeneti jel esete, piros: csökkentett amplitúdójú bemenet esete, $\mu = 0,0001$



12.12. ábra. Az LMS algoritmus vizsgálata, $e(n)$, kék: eredeti bemeneti jel esete, $\mu = 0,0001$, piros: csökkentett amplitúdójú bemenet esete, $\mu = 0,01$

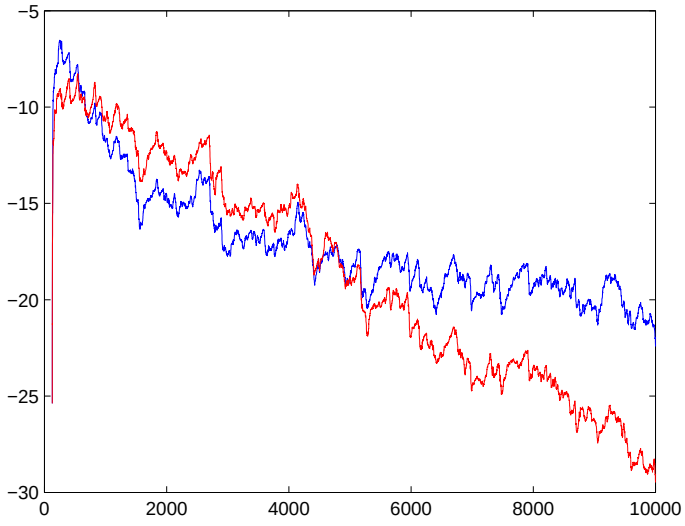
8. A 12.14. ábrán megfigyelhető, hogy az NLMS algoritmus – mivel beépített normalizációt tartalmaz a referenciajel teljesítménye szerint – nem szorul bátorsági tényező korrekcióra annak érdekében, hogy megváltozott feltételek mellett, tehát csökkentett jelszintek esetében is változatlan tulajdonságokkal üzemeljen.

```
% az NLMS vizsgálata az eredeti mu értékkel
[e_nlms_2,w_nlms_2] = nlms(1.5,128,x2,y2,0.01);
erle_nlms_2 = erle(e_nlms_2,y2);
```



12.13. ábra. Az LMS algoritmus vizsgálata, $ERLE(n)$, kék: eredeti bemeneti jel esete, $\mu = 0,0001$, piros: csökkentett amplitúdójú bemenet esete, $\mu = 0,01$, a két görbe egybeesik

```
4 plot([1:length(y2)],erle_lms_2,'b',[1:length(y2)],erle_nlms_2,'r');
```



12.14. ábra. Az $ERLE(n)$ függvények összehasonlítása csökkentett bemeneti amplitúdók esetén, kék: LMS, $\mu = 0,01$, piros: NLMS, $\tilde{\mu} = 1,2$

9. Általánosságban elmondható, hogy a szűrő hosszának növelésével kisebb bátorsági tényező beállítására van szükség. A jelenség oka, hogy a szűrő hosszának változtatásával eltérő számú bemeneti mintát dolgozunk fel, így a bemeneti adatokon csúszó vektor hossza is változik. Az algoritmusban kiszámított skaláris szorzatok értéke a szűrő hosszának növelésével ily módon szükségszerűen növekszik.

12.4. Az FxLMS algoritmus

12.4.1. Az FxLMS algoritmus megvalósítása

A feladat megoldása a következő:

```
function [e,w] = fxlms(mu,M,x,y,c);

%   Bemeneti paraméterek:
%
%   mu = lépésköz-paraméter (bátorsági tényező) [1x1]
%   M = a szűrő hossza [1x1]
%   x = a rendszer bemeneti jele [Nx1]
%   y = a rendszer kimeneti jele [Nx1]
%   c = a másodlagos átvitel becslőjének impulzusválasza [Ix1]

%   Kimeneti változók:
%
%   e = becslési hiba [Nx1]
%   w = végső szűrőegyüttható-készlet [Mx1]

%   zérus kezdeti értékek
    w = zeros(M,1);

%   a bemeneti jel mintáinak darabszáma
    N = length(x);

%   x, y és c oszlopvektorok
    x = x(:);
    y = y(:);
    c = c(:);

    I = length(c);

%   FxLMS algoritmus
    for n = M+I:N
        xc = [];
        for i = 0:M-1
            xc = [xc; c' * x(n-i:-1:n-i-I+1)];
        end
        xc = flipud(xc);
        e(n) = y(n) - w' * xc;
        w = w + 2 * mu * xc * e(n);
    end

%   e oszlopvektor
    e = e(:);
```

FxLMS algoritmust realizáló MATLAB függvény

12.4.2. A megvalósított algoritmus ellenőrzése

1. A feladat egy lehetséges megoldása 50 ütemes késleltetés esetében:

```
% FIR szűrő együtthatói
h = [zeros(50,1); (0.5.^[0:9])'];
```

2. A feladat megoldása:

```
% bemeneti jel
x = randn(1000,1);
% kimeneti jel
y = conv(h,x);
```

3. A feladat egy lehetséges megoldása 20 ütem késleltetés megválasztásával:

```
% másodlagos út
c = [zeros(20,1);1;0.1;0.01;0.001]
```

4. A feladat egy lehetséges megoldása 60 hosszú szűrő és $\mu = 0,01$ mellett a következő:

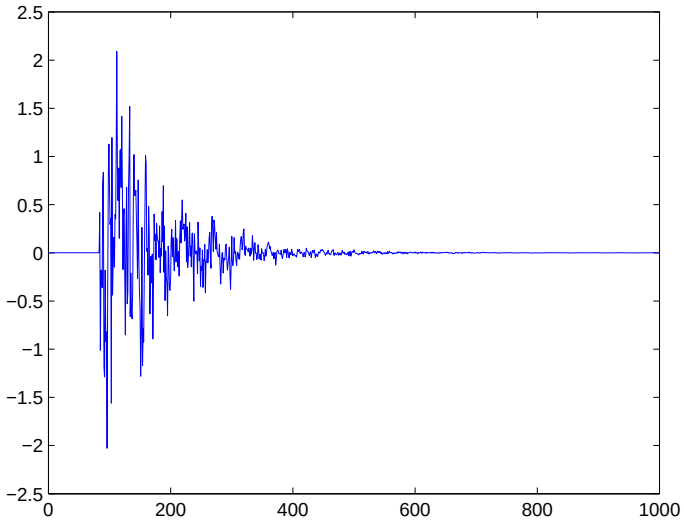
```
% FxLMS algoritmus
[e_fxlms,w_fxlms] = fxlms(0.01,60,x,y,c);
```

5. A feladat megoldása a következő. A hibajel időfüggvénye a 12.15. ábrán látható. Figyeljük meg, hogy a hibajel kezdetben a késleltetések miatt zérus.

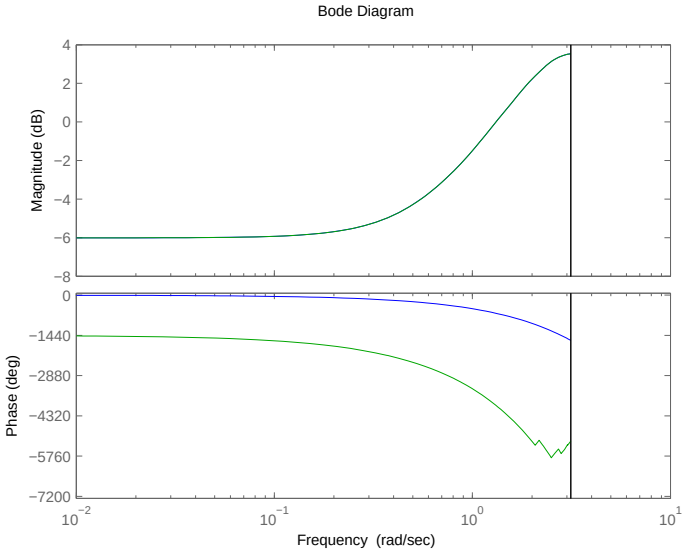
```
plot(e_fxlms);
```

6. A megoldásban egyrészt az eredeti átvitelt kell ábrázolni Bode diagramon, majd az adaptív szűrő és másodlagos út együttes átvitelét. A diagram a 12.16. ábrán látható. Megfigyelhető, hogy a két átvitel amplitúdókarakterisztikája jól egyezik, tehát a megoldás helyes.

```
% eredeti átvitel
dbode(1,h',1)
hold on;
% fxlms adaptív szűrő és másodlagos átvitel
h1 = conv(w_fxlms,c)';
dbode(1,h1,1)
```



12.15. ábra. Az FxLMS algoritmus vizsgálata, $e(n)$, $\mu = 0,01$



12.16. ábra. Az elsődleges út (kék); valamint az adaptív szűrő és másodlagos út együttes átvitele (zöld)

12.5. Az aktív zajcsökkentő struktúra

A rendszer működtetésére szolgáló MATLAB alapú grafikus felhasználói felület képe a 12.17. ábrán látható.

12.5.1. Az aktív zajcsökkentő struktúra vizsgálata

A rendszer vizsgálata alapvetően kísérletezésből áll, melynek során a hallgatók változtatják a zaj paramétereit (jelalak, frekvencia), valamint a rendszer paramétereit (bátorsági tényezők, erősítések) egyaránt. A feladat célja, hogy lehetőség nyíljon annak megtapasztalására, milyen nehézségeket

12.17. ábra. Az ANC rendszer felhasználói felülete

rejt magában egy adaptív szűrőt alkalmazó gyakorlati jelentőségű struktúra üzemeltetése. A gyakorlatban számos probléma jelentkezhet. Ha a hibamikrofon vagy a beavatkozó hangszóró jele nem elégséges szintű, az a rendszer instabilitásához vezethet. Hasonló mondható el a bátorsági tényező túl nagyra történő megválasztásáról is. Nagyobb μ esetén a beállás gyorsabb lesz, de a maradó zaj is nagyobb mértékű mint kisebb bátorsági tényező esetében.