VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

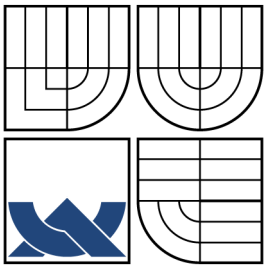
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ČÍSLICOVÉ ZPRACOVÁNÍ SIGNÁLŮ V REÁLNÉM ČASE

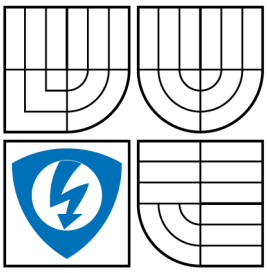
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE BC. ZDENĚK ZAMAZAL
AUTHOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND
COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ČÍSLICOVÉ ZPRACOVÁNÍ SIGNÁLŮ V REÁLNÉM ČASE
DIGITAL SIGNAL PROCESSING IN REAL TIME

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — BC. ZDENĚK ZAMAZAL
AUTHOR

VEDOUCÍ PRÁCE — ING. ONDŘEJ RÁŠO
SUPERVISOR

BRNO 2011

ZDE VLOŽIT LIST ZADÁNÍ

Z důvodu správného číslování stránek

ZDE VLOŽIT PRVNÍ LIST LICENČNÍ SMOUVY

Z důvodu správného číslování stránek

ZDE VLOŽIT DRUHÝ LIST LICENČNÍ SMOUVY

Z důvodu správného číslování stránek

ABSTRAKT

Tato práce se zabývá zpracováním signálů v oblasti adaptivní filtrace. Jsou zde nastíněny základní principy adaptivní filtrace a hlavní smysl této práce je vytvořit laboratorní úlohy v prostředí LabView, které se zabývají adaptivní filtrací. Tyto laboratorní úlohy mají sloužit studentům ke studiu a bude je možno využít v laboratorní výuce. Cílem je, aby bylo možné úlohy spojit s externím zařízením, které v těchto úlohách představuje mikrofon. Mikrofon je použit jako rozhraní ke snímání řečového signálu uživatele. V práci je nastíněna teorie Wienerova filtru a problém adaptivní filtrace. Jsou zde popsány současné algoritmy adaptivních filtrů a jejich aplikace. Důraz je kladem na algoritmus LMS a jeho mutace. Laboratorní úlohy se zabývají aplikacemi: Adaptivní potlačení echa, Aktivní potlačování rušení a Přímá identifikace systému. Každá z těchto úloh je samostatně spustitelná (v LabView nebo pomocí Run-time engine) a obsahuje potřebnou teorii i blokové schéma, proto je lze použít i bez návodu k použití.

KLÍČOVÁ SLOVA

adaptivní filtr, LabView, Wienerův filtr, LMS filtr, potlačení echa, potlačení rušení, identifikace systému, číslicové zpracování signálů, virtuální instrument, filtrování signálu

ABSTRACT

This work deals with digital signal processing in the field of adaptive filtering. Fundamental basics of adaptive filtering are described and primary aim is to create executable laboratory examples, using adaptive filtering, in LabView programming language. These laboratory examples are intended to be used by students fo studying and during laboratory lessons. Objective is to connect the examples with external devices, such as microphone. A microphone is used as an user's speech input acquiring interface. In the thesis is depicted Wiener's filter and problem of adaptive filtering is discussed. Contemporary adaptive algorithms are described and their applications as well. Most mentioned is the LMS algorithm and it's forms. Laboratory examples use following concepts: Adaptive Echo Cancellation, Active Noise Control and System Identification. Each of these examples is solely executable (need for LabView or Run-time engine), consisting also of theory with diagrams. Examples therefore are usable even without manual.

KEYWORDS

adaptive filter, LabView, Wieners filter, LMS filter, echo cancellation, noise cancellation, system identification, digital signal processing, virtual instrument, signal filtering

ZAMAZAL Z. *Číslicové zpracování signálů v reálném čase*. Brno: Vysoké Učení Technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav telekomunikací, Rok vydání 2011. Počet stran 66 s., Počet stran příloh 10 s. příloh. Diplomová práce. Vedoucí práce byl Ing. Ondřej Rášo.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „ČÍSLICOVÉ ZPRACOVÁNÍ SIGNÁLŮ V REÁLNÉM ČASE“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne ....

(podpis autora)

Rád bych poděkoval svému vedoucímu diplomové práce Ing. Ondřeji Rášovi za ochotu a projevenou trpělivost během vypracovávání této diplomové práce. Také děkuji svým přátelům Alžbětě Viznarové a Danielu Zvolenskému za obrovskou podporu, která mi pomohla dokončit tuto práci.

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

Tato diplomová práce se bude tématem číslicového zpracování signálů pomocí programu LabView. LabView je představitelem hybridního grafického programování. To znamená, že zdrojovým kódem programu není textový kód, ale proprietární formát. Programování v LabView má výhodu v tom, že je více intuitivní, než v jiných jazycích. Nevýhodou může být, že zdrojový kód nelze upravit mimo program LabView a problémem je zpětná kompatibilita kódu.

Tato práce se zabývá především oblastí adaptivního filtrování a klade na čtenáře nároky na základní znalosti zpracování číslicových signálů a číslicových filtrů.

V práci se zabýváme teorií Wienerova filtru a jeho adaptivní verzí. Představíme si několik adaptivních algoritmů. Je zde popsáno využití různých adaptivních filtrů pro různorodé aplikace mezi zpracováním signálů. Adaptivní filtry v mnoha variantách našly uplatnění v široké paletě aplikací. Používají se například pro zpracování bio signálů, jako signál EKG. Každodenně používáme adaptivní filtr například při telefonování, kde využijeme odstranění echa. Adaptivní filtry potlačují rušení ze zvuku či jiného signálu. Můžeme pomocí nich frekvenčně vyrovnat přenosový kanál. Největší využití našly tyto filtry v oblasti telekomunikací.

V této práci si klademe za cíl vytvořit laboratorní úlohy, které by představily možnosti adaptivní filtrace a názorně zobrazily jejich sílu ve zpracování signálů.

Jsou vytvořeny tři úlohy, které jsou schopny interagovat s uživatelem pomocí přívětivého uživatelského rozhraní. Úlohy se zabývají aplikacemi: a) Potlačování echa, b) Potlačování akustického rušení, c) Identifikace neznámého systému. Úlohy kladou důraz na zobrazení výsledků ve snadno srozumitelné formě. Je umožněno využít mikrofonový vstup pro získání signálu z řeči. A tam, kde je to možné, úloha pracuje v reálném čase.

1 PROSTŘEDÍ LABVIEW

LabVIEW je vysoce produktivní prostředí pro tvorbu uživatelských aplikací, které mohou pracovat s hardwarovým rozhraním ke sběru reálných dat z prostředí, využívá se v oblasti vědy a výzkumu. LabVIEW je moderní programovací vývojové prostředí, kde tvoříme program ve formě blokových diagramů. Obsahuje mnoho knihoven pro analýzu měřených dat. Data lze získávat přes seriové a paralelní rozhraní a s využitím měřicích karet z různých měřicích přístrojů. Obsahuje cca 600 ovladačů (a další přibývají či je lze najít na internetu). Je možná i síťová komunikace přes TCP/IP a mnoho dalších úloh.

Programy v LabVIEW se nazývají virtuální přístroje nebo VI, protože svým vzhledem a činností jsou obdobou skutečných přístrojů, jako jsou např. osciloskopy a multimetry. Termín VI je tedy obdobou termínu program nebo funkce v jiných programovacích jazycích. K vytváření takových programů obsahuje LabVIEW širokou sadu nástrojů pro sběr, analýzu, zobrazení a ukládání dat a také nástroje pro hledání a odstraňování chyb v programu.

Nyní existuje verze LabView 2010, nicméně práce je vytvořena ve verzi 2009. V nové verzi jsou pouze drobné změny a úlohy vytvořené v této práci jsou kompatibilní a je možné je spustit i v nové verzi programu.

1.1 Porovnání LabView, Matlab, C

Program LabView má výrazné přednosti, díky kterým vyniká oproti Matlabu. Na prvním místě figuruje grafická povaha, díky které máte možnost vidět, co se uvnitř děje. Nedochází zde pouze k interpretaci slov. Za druhé je tu možnost pomocí přídavných rozhraní spojit externí nástroje a hardware. LabView kombinuje tyto vlastnosti s mnoha vestavěnými užitečnými funkcemi, které umožňují provádět všechny druhy zpracovávání signálů.

Mnoho inženýrů se obrací na Matlab nebo jazyk C, aby mohli odsimulovat jejich návrh dříve, než začnou s výrobou produktu. Oba nástroje jsou výborné a velmi výkonné. Nicméně pro celou práci od hardwaru až k dílčím úkonům se hodí nejlépe LabView a zastiňuje tak obě dříve zmíněné cesty. Obecně grafické rozhraní programu LabView lze velmi snadno pochopit a porozumět mu. Při bližším zkoumání uvidíme, že LabView má velmi mocnou sadu nástrojů pro zpracování signálů, neklade značné požadavky na znalost programování ,jako například deklaraci proměnných. Nepotřebuje kompilovat, má pokročilé ovládání pro různé instrumenty a funkce pro získávání externích dat. Dále dokáže výborně zobrazovat analyzovaná data kdekoli v různých místech implementace, například komunikačního systému.

Postavením komunikačního systému založeném na LabView nám dovolí připojit další nové komunikační datové rozhraní a jednoduše je zaintegrovat. Řekneme, že LabView se za dobu vývoje dokázal dostat do stavu, kdy výpočty provádí velmi rychle, přesto se nedá říci, že by byl nástrojem pracujícím v reálném čase. Při vývoji komunikačního systému je důležité si uvědomit, že LabView je velmi pomalý pro výpočty probíhající v reálném čase, ale je výborným nástrojem k měření pomocí různých instrumentů. Tento nedostatek v rychlosti je vyvažován vysokou přizpůsobivostí, jednoduchostí programování, zobrazením dat a mimoto má LabView přívětivé uživatelské rozhraní.

1.2 Obecná doporučení pro práci v LabView

Jako v každém programovacím prostředí, existují drobná vylepšení, kterými můžete maximalizovat efektivitu vašeho programu. tento seznam není zdaleka vyčerpávající, ale obsahuje nejzásadnější postupy, které ušetří váš čas za malé množství námahy.

1. Dávejte pozor, co umísťujete do logických smyček: opakované výpočty by měly vždy mít místo vně smyčky. Toto by měl být obecně platný standard pro jakýkoliv programovací jazyk včetně LabView. Nesnažte se nutně vkládat hromadu operací (nebo sub-VI) dovnitř smyčky, pokud jejich výpočet nezávisí na obnovení v každé kole. také buďte opatrní, kam umísťujete řídící prvky instrumentu.

2. Používejte předem připravené výpočty co nejvíce to bude možné: tato rada platí pro koeficienty filtrů, komplexní hodnoty nebo jakoukoli jinou hodnotu, která se nebude měnit během průběhu programu. Nemá smysl například počítat filtru nebo Fourierovu transformaci (FFT) z neměnné posloupnosti dat.

3. Vyvarujte se globálních a lokálních proměnných, pokud možno. Používejte radši proměnné v rámci sekvence místo lokálních proměnných a nepoužívejte globální proměnné v aplikacích, které jsou kriticky závislé na rychlosti. Tímto stylem nebude sice kód velmi přehledný. Globální a lokální proměnné mají příjemnou vlastnost, že se sami pojmenovávají, je okamžitě vidět, zda jsou ke čtení nebo k zápisu a jejich použití je velmi přehledné. Ale jejich použití bývá neskutečně pomalé.

4. Co nejvíce snižte počet zobrazovaných prvků na čelním panelu. Množství vykreslovaných dat značně zpomaluje celou aplikaci. Zobrazování dat na čelním panelu bývá hodně pomalé. Pokud není ten či onen graf pro vaši aplikaci důležitý, radši jej úplně vynechte. Ponechte počet zobrazovaných prvků na minimu a všimnete si rozdílu v rychlosti vaší aplikace.

5. Využívejte co nejvíce zabudované VI. Pokud se podíváte blíže na většinu zabudovaných funkcí LabView, všimnete si, že základním kódem VI je volání rutiny jazyce C. Tyto volané rutiny jsou mnohem rychlejší, než jejich ekvivalentní protějšek složený do VI. Může lákat složit vlastní VI pomocí základních prvků, ale nejlepší je nejdřív se ujistit, zda už podobná funkce v LabView neexistuje.

6. Kompilujte spustitelné aplikace. LabView má výbornou schopnost vyrobit z vašeho VI samostatnou spustitelnou aplikaci. Tento zázrak je možný pomocí příslušné aplikaci dostupné z menu LabView, pod názvem Application Builder. Tato metoda umožňuje spustit váš virtuální instrument, bez nutné instalace programu LabView. Stačí si stáhnout dostupný kompatibilní Run-time engine. Pomocí této metody nemáte zaručeno, že rychlost vykonávání programu na vašem PC vzroste, ale sníží se náročnost na paměť, protože nemusíte mít spuštěnou celou aplikaci LabView. Další výhodou je, že nikdo nebude mít dovoleno změnit či zobrazit vnitřní kód nebo měnit blokový diagram.

2 ADAPTIVNÍ FILTRACE

V mnoha aplikacích, kde využíváme filtrování, nám není potřebná kmitočtová odezva filtru známa dopředu, anebo se může měnit v čase. Příkladem budiž potlačení harmonických složek zvuku od motoru auta pomocí stereo systému v autě. Adaptivní filtr, který se sám dokáže navrhnout a přizpůsobit se měnícímu se systému může být v takovýchto aplikacích naprosto nezbytný. Adaptivní filtry jsou součástí široké palety přístrojů, využívaných především v oblasti telekomunikací.

Adaptivní filtr mají velkou a rozmanitou škálu použití. Adaptivní filtry se používají, jak bylo již zmíněno k potlačování hluku v kabinách aut. Velice důležitá je tato aplikace také u motorových letounů, kde pilot musí komunikovat s ostatními přes rádiové spojení. Velice užitečné použití našly adaptivní filtry v telefonních přístrojích, kde dochází ke vzniku echa a adaptivní filtry jsou schopny tento efekt úspěšně potlačit. Dnes adaptivní potlačení echa nalezneme prakticky ve všech moderních mobilních ale i klasických telefonech. Využití také nalezly při metodách ekvalizace kanálu, kdy je potřeba správně vykompenzovat vlastnosti nedokonalého a nestacionárního komunikačního kanálu.

Adaptivní filtr lze využít i pro další inovativní a pokročilé aplikace jako například aktivní potlačování hluku v dětských inkubátorech. Nadměrný hluk v předporodních jednotkách pro péči i uvnitř inkubátorů může podle studií mít neblahý vliv na zdraví dítěte. Hluk inkubátoru je typicky širokopásmový a zahrnuje zdroje hluku jako jsou pumpy, větráky a ohřívací přístroje. Pro tuto aplikaci se vytváří počítačový model prostředí inkubátoru včetně integrovaného systému potlačování rušení. Tato metoda je dále rozšířena o "efekt dělohy" použitím vnitrotělních zvuků a zvuku tepajícího srdce matky dítěte. Těmto přídavným zvukovým projevům byly dokázány prospěšné účinky na zdraví potomka.

2.1 Krátká rekapitulace typů filtrů

Existují dva typy realizace číslicových filtrů, shrňme si je pro začátek:

FIR filtry

$$y[n] = \sum_{k=0}^{N} b_k x[n-k] \tag{2.1}$$

- Pro konečné koeficienty jsou vždy stabilní.
- Fázová odezva odpovídá prostému zpoždění – jejich fáze je lineární.
- Změny frekvenční odezvy způsobené malými změnami koeficientů jsou malé a snadno predikovatelné.

IIR filtry

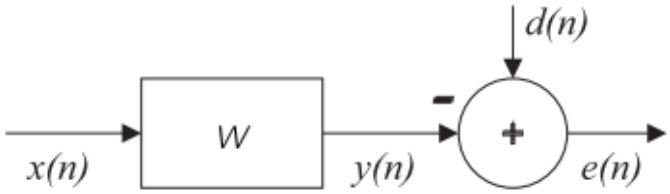
$$y[n] = \sum_{k=0}^{N} b_k x[n-k] - \sum_{k=1}^{M} a_k y[n-k] \tag{2.2}$$

- Nemusí být vždy stabilní.
- Jejich fáze nemůže být lineární.
- Malé změny koeficientů a_k mohou způsobit velké změny frekvenční odezvy.

2.2 Wienerův filtr

Struktury adaptivních filtrů předpokládají podobnost k Wienerovu filtru. Adaptivní filtry na rozdíl od stacionárního Wienerova filtru mění své koeficienty v čase tak, aby reflektovaly změny na vstupním signálu do vstupního signálu nebo změní parametry filtru.

Wienerův filtr je lineární časově invariantní filtr, který minimalizuje $E[e^2]$, varianci chybového signálu. Užitečným signálem je zde $d[n]$, řešíme, jaké mají být koeficienty Wienerova filtru tak, abychom ze signálu $x[n]$ získali signál $y[n]$ co nejvíce podobný signálu $d[n]$. Chyba $e[n] = d[n] - y[n]$ musí být minimální. Jak by se mohlo zdát, zabývat se tímto problémem je pošetilé, protože už máme k dispozici d_n. Naopak tento nápad je užitečný pro řadu aplikací.



Obr. 2.1: Wienerův filtr

Na obrázku vidíme vstupní užitečný signál $d[n]$. Vstupem filtru je signál $x[n]$ a výstupem $y[n]$. Jestliže užije FIR implementaci Wienerova filtru o K koeficientech, pak pro chybový signál platí rovnice:

$$e(n) = d(n) - y(n) = d(n) - \sum_{i=0}^{K-1} w_i x(n-i) \tag{2.3}$$

kde parametr n, respektive K-i reprezentují diskrétní čas vzorku signálu. Konstanta w_i je i-tý koeficient filtru. Druhá mocnina okamžité chyby je předepsán touto rovnicí:

$$e^2(n) = (d(n) - y(n))^2 = d^2(n) - 2d(n)y(n) + y^2(n) \tag{2.4}$$

Samotné koeficienty filtru W můžeme napsat jako matici s konečným počtem hodnot:

$$W(n) = [w_0(n)w_1(n) \dots w_{K-1}(n)]^T \tag{2.5}$$

2.2.1 Hledání optima

Pro nalezení optimálních koeficientů Wienerova filtru využíváme výraz (2.4). vzhledem k tomu, že se jedná o kvadratickou funkci, existuje pro no pouze jedno globální minimum bez lokálních extrémů.

Při hledání optima této funkce používáme gradient funkce variance $E[e^2(n)]$, což je vektor prvních derivací podle jednotlivých koeficientů:

$$grad\left\{E[e^2(n)]\right\} = \frac{\partial E[e^2(n)]}{\partial W(n)} \tag{2.6}$$

který má směr největší změny dané funkce. proto je hledané minimum v místě, kde je gradient roven nule. Respektive, kde jsou parciální derivace rovny nule. To představuje nulovou změnu a dokonalou adaptaci filtru s optimálními koeficienty.

2.2.2 Metoda největšího spádu

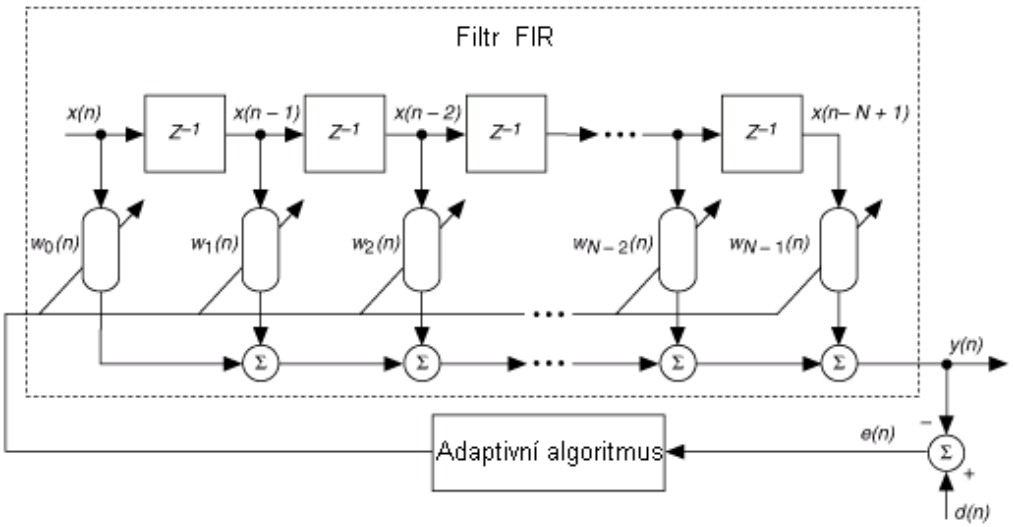
Výpočet optimálních koeficientů Wienerova filtru pomocí předchozích rovnic si žádá příliš velké nároky na výpočetní výkon. Navíc je nutné opětovně přepočítávat koeficienty filtru, v případě, že signály jsou statisticky nestacionární. Řešením se nabízí metoda největšího spádu, při které se koeficienty filtru iterativně posunují ve směru gradientu. Následující rovnice předpisuje výpočet nových hodnot koeficientů filtru pro novou iteraci:

$$W(n+1) = W(n) - \mu \cdot grad\left\{E[e^2(n)]\right\} \tag{2.7}$$

kde parametr μ je tzv. konvergenční koeficient.

2.3 Algoritmy adaptivních filtrů

Každý adaptivní filtr se skládá ze dvou částí: a) lineární filtr b) adaptivní algoritmus. Lze použít lineární filtry různých typů filtrů, jako jsou filtry s konečnou impulsní charakteristikou (FIR) nebo filtry s nekonečnou impulsní charakteristikou (IIR). Adaptive Filter Toolkit v Labview podporuje pouze třídu typu FIR. Následující diagram znázorňuje schéma adaptivního filtru.



Obr. 2.2: Blokový diagram FIR adaptivního filtru

Kde:
$x(n)$ je vstupní signál lineárního filtru v čase n
$y(n)$ je odpovídající výstupní signál
$d(n)$ je druhý vstupní signál adaptivního filtru
$e(n)$ je chybový signál, který udává rozdíl mezi $d(n)$ a $y(n)$
z^{-1} je jednotkové zpoždění
$w_i(n)$ je násobitel. Pro tento násobitel se používá pojem koeficient filtru
i je celé číslo v mezích [0, N-1]
Adaptivní algoritmus iterativně upravuje $w_i(n)$ tak, aby minimalizoval energii $e(n)$

VÝPOČET VÝSTUPNÍHO SIGNÁLU ADAPTIVNÍHO FILTRU
Adaptivní filtr počítá výstupní signál $y(n)$ podle následujícího vzorce:

$$y(n) = \vec{u}^T(n) \cdot \vec{w}(n) \tag{2.8}$$

Kde:
$\vec{u}(n)$ je vstupní vektor filtru. $\vec{w}(n) = [x(n)x(n-1)\dots x(n-N+1)]^T$
$\vec{w}(n)$ je vektor koeficientů filtru. $\vec{w}(n) = [w_0(n)w_1(n)\dots w_{N-1}(n)]^T$

2.3.1 ROZDĚLENÍ ALGORITMŮ ADAPTIVNÍCH FILTRŮ

Abychom mohli řídit jak filtr bude upravovat své koeficienty, můžeme použít různé algoritmy pro FIR adaptivní filtr. Adaptivní algoritmus upravuje koeficienty filtru, aby minimalizoval odchylkovou funkci J(n):

$$J(n) = E[e^2(n)] \tag{2.9}$$

Kde $E[e^2(n)]$ je předpokládaný odhad $e^2(n)$ a $e^2(n)$ je druhá mocnina chybového signálu v čase n. Podle toho, jak adaptivní algoritmus provádí výpočet odchylkové funkce J(n), lze rozdělit adaptivní algoritmy do dvou hlavních skupin:

Least Mean Squares (LMS)
LMS algoritmus počítá J(n) použitím následující rovnice:

$$J(n) = e^2(n) \tag{2.10}$$

Z této rovnice vidíme, že LMS algoritmus používá okamžité hodnoty $e^2(n)$ v čase n jako odhad $E[e^2(n)]$.

Recursive Least Squares (RLS)
RLS algoritmus počítá J(n) použitím následující rovnice:

$$J(n) = \frac{1}{N} \sum_{i=0}^{N-1} \lambda^i e^2(n-1) \tag{2.11}$$

Kde N je délka filtru a λ je zapomínací faktor. Tímto algoritmem z výpočtu získáme nejenom okamžitou hodnotu $e^2(n)$, ale také předchozí hodnoty $e^2(n-1), e^2(n-2), \ldots, e^2(n-N+1)$. Hodnota zapomínacího faktoru je v mezích 0 až 1. Pokud je zapomínací faktor menší než 1, znamená to, že algoritmus pokládá větší váhu na současnou hodnotu vzorku a méně na předchozí hodnoty. Výsledná $E[e^2(n)]$ algoritmu RLS je přesnější než u algoritmu LMS. Algoritmus LMS má méně výpočetních nároků a jeho požadavky na paměť jsou menší, než u algoritmu RLS. Nicméně činitel podmíněnosti vstupní korelační matice, neboli korelační matice vstupního signálu, může ovlivnit rychlost konvergence adaptivního filtru. Rychlost konvergence algoritmu RLS je mnohem větší, než u algoritmu LMS. Nicméně algoritmus RLS vyžaduje větší výpočetní výkon než algoritmus LMS.

Činitel podmíněnosti
Činitel podmíněnosti je definovaný následující rovnicí jako podíl maximální a minimální hodnoty vstupní korelační matice.

$$X = \lambda_{max}/\lambda_{min} \tag{2.12}$$

Kde λ_{max} a λ_{min} jsou maximum a minimum vlastních hodnot vstupní korelační matice. Vstupní korelační matice má rozměr $N \times N$, kde N je délka filtru. Vstupní korelační matice je definovaná následující rovnicí:

$$\mathbf{R} = E[\vec{u}(n)\vec{u}^T(n)] \tag{2.13}$$

Kde $\vec{u}(n)$ je vstupní vektor filtru a $E[x]$ je matematický odhad x. Vysoká hodnota činitele podmíněnosti vstupní korelační matice snižuje rychlost konvergence výsledného adaptivního filtru.

2.4 Algoritmus LMS

Algoritmus se stochasticky gradientní adaptací (Least Mean Squares „LMS“) modeluje koeficienty filtru tak, aby minimalizoval odchylkovou funkci. Na rozdíl od algoritmu RLS se zde neprovádí žádné operace s maticemi, proto algoritmus LMS má menší nároky na výpočetní výkon a paměťové místo. Samotná implementace je také mnohem jednodušší, než u algoritmu RLS. Rychlost konvergence algoritmu RLS je mnohem větší, než u algoritmu LMS. Nicméně činitel podmíněnosti vstupní korelační matice (korelační matice vstupního signálu) může ovlivnit rychlost konvergence adaptivního filtru.

Všechny algoritmy založené na LMS filtru mají tzv. krok, jehož velikost určuje míru korekce uskutečněné během jedné iterace v adaptaci filtru. Určení optimální délky kroku není vždy jednoduché. Příliš malý krok brání konvergenci v přijatelném čase, zatímco příliš velký krok může zapříčinit nepřesnost filtru. Filter Design Toolbox v LabView obsahuje algoritmy k určení největšího možného kroku při zachování konvergence filtru.

Standardní LMS

Standarní LMS algoritmus provádí následující operace nad signály pro určení koeficientů adaptivního filtru:
1) Výpočet výstupního signálu y(n) z adaptivního filtru.
2) Výpočet chybového signálu e(n) použitím rovnice $e(n) = d(n) - y(n)$
3) Výpočet nových koeficientů použitím následující rovnice:

$$\vec{w}(n+1) = \vec{w}(n) + \mu \cdot e(n) \cdot \vec{u}(n) \tag{2.14}$$

Kde μ je konstanta velikosti kroku adaptivního filtru, $\vec{w}(n)$ je vektor koeficientů filtru a $\vec{u}(n)$ je vstupní vektor filtru.

Normalizovaný LMS

Normalizovaný LMS (NLMS) je modifikovaná forma standardního LMS algoritmu. Algoritmus NLMS provádí výpočet koeficientů pomocí následující rovnice:

$$\vec{w}(n+1) = \vec{w}(n) + \mu \cdot e(n) \cdot \frac{\vec{u}(n)}{\|\vec{u}(n)\|^2} \tag{2.15}$$

Uvedenou rovnici můžeme taky přepsat následujícím způsobem:

$$\vec{w}(n+1) = \vec{w}(n) + \mu \cdot e(n) \cdot \vec{u}(n) \tag{2.16}$$

kde

$$\mu(n) = \mu / \|\vec{u}(n)\|^2 \tag{2.17}$$

Podle této rovnice je algoritmus NLMS totožný se standardním algoritmem LMS, až na to, že NLMS má časově proměnnou velikost kroku $\mu(n)$. Tato změna je schopna vylepšit rychlost konvergence adaptivního filtru.

Propustný LMS (Leaky LMS)

Odchylková funkce propustného LMS algoritmu je definována následující rovnicí:

$$J(n) = e^2(n) + \alpha \sum_{i=0}^{N-1} w_i^2(n) \tag{2.18}$$

Kde α je faktor propustnosti z rozmezí (0 až 0,1). Kvůli přítomnosti α je odchylková funkce propustného LMS algoritmu odlišná od standardního LMS algoritmu. Propustný LMS algoritmus zmírňuje problém přetékání koeficientů především u implementací s pevnou řádovou čárkou, protože odchylková funkce tohoto algoritmu platí jak pro $e^2(n)$, tak i pro koeficienty filtru. Algoritmus propustné LMS metody provádí výpočet koeficientů adaptivního filtru pomocí následující rovnice:

$$\vec{w}(n+1) = (1 - \mu\alpha) \cdot \vec{w}(n) + \mu \cdot e(n) \cdot \vec{u}(n) \tag{2.19}$$

Jestliže $\alpha = 0$, pak propustný LMS algoritmus bude totožný se standardním LMS algoritmem. Příliš velký faktor propustnosti avšak může zapříčinit vysokou odchylku v ustáleném stavu.

Normalizovaný propustný LMS algoritmus (NLMS)

Normalizovaný propustný LMS algoritmus je modifikovaná verze propustného LMS algoritmu. Tento algoritmus provádí výpočet koeficientů adaptivního filtru pomocí následující rovnice:

$$\vec{w}(n+1) = (1 - \mu a) \cdot \vec{w}(n) + \mu \cdot e(n) \cdot \frac{\vec{u}(n)}{\|\vec{u}(n)\|^2} \tag{2.20}$$

Existují i další algoritmy na bázi LMS:

- ZNAMÉNKOVÝ LMS
- RYCHLÝ BLOKOVÝ LMS

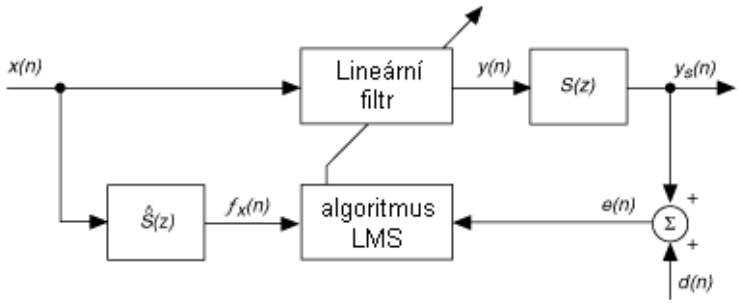
Ale nebudeme se jimi zde zabývat.

2.5 Algoritmus FXLMS (Filtered-x LMS)

U některých aplikací adaptivních filtrů musíme brát v úvahu sekundární cestu signálu, jako například u aktivního potlačení rušení. Sekundární cesta je cesta která se uplatňuje mezi výstupem adaptivního filtru $y(n)$ a výsledným chybovým signálem $e(n)$. Vlastnosti sekundární cesty zapříčiňují fázové posuny nebo zpoždění signálu při přenosu. Konvenční LMS algoritmus neumí vykompenzovat efekt sekundární cesty. U takového typu aplikace můžeme využít metodu FXLMS pro návrh adaptivního filtru.

Filtered-x LMS

Následující schéma zobrazuje adaptivní filtr vytvořený pomocí FXLMS algoritmu.



Obr. 2.3: Schéma algoritmu FXLMS

Kde:
$x(n)$ je vstupní signál lineárního filtru v čase n
$y(n)$ je odpovídající výstupní signál
$d(n)$ je druhý vstupní signál adaptivního filtru
$S(z)$ je impulsní odezva sekundární cesty
$y_s(n)$ je signál produkovaný ze sekundární cesty
$e(n)$ je chybový signál, který vznikne superpozicí $d(n)$ a $y(n)$
$\hat{S}(z)$ je odhad $S(z)$
$f_x(n)$ je výsledný výstupní signál z $\hat{S}(z)$
Algoritmus LMS iterativně upravuje koeficienty lineárního filtru tak, aby minimalizoval energii $e(n)$.
Poznámka: V reálných aplikacích se princip superpozice uplatňuje ve volném prostoru. Nemůžeme získat signály $d(n)$ a $y(n)$ zvlášť. Můžeme získat pouze superponovaný signál $e(n)$ a potom využít FXLMS adaptivní filtr k minimalizaci $e(n)$. Všimněme si těchto rozdílů mezi obrázkem na schématu a typickým adaptivním filtrem.

a) Je přítomen systém sekundární cesty. Abychom vykompenzovali efekt sekundární cesty musíme odhadnout impulsní odezvu sekundární cesty a vzít tento odhad v úvahu. Schéma typického adaptivního filtru sekundární cestu neobsahuje.

b) Podle schématu FXLMS filtru získáváme signál $e(n)$ přímo. Podle schématu typického adaptivního filtru získáváme signály $d(n)$ a $y(n)$ odděleně a počítáme $e(n)$ s použitím rovnice $e(n) = d(n) - y(n)$.

c) Vstupní signály pro FXLMS filtr jsou $x(n)$ a $e(n)$, kdežto vstupními signály pro typický filtr jsou $x(n)$ a $d(n)$.

Algoritmus FXLMS provádí k výpočtu koeficientů adaptivního filtru následující operace:

1. Vypočte výstupní signál $y(n)$ z adaptivního filtru.
2. Filtruje vstupní signál $x(n)$ pomocí $\hat{S}(z)$ a vytvoří $f_x(n)$.
3. Upraví koeficienty filtru podle následující rovnice:

$$\vec{w}(n+1) = \vec{w}(n) - \mu \cdot e(n) \cdot \vec{f_x}(n) \tag{2.21}$$

Kde μ je velikost kroku adaptivního filtru a $\vec{w}(n)$ je vektor koeficientů filtru.

Normalizovaný FXLMS

Normalizovaný FXLMS algoritmus je modifikovaná verze FXLMS algoritmu. Tato metoda kombinuje normalizované LMS a FXLMS algoritmy. Normalizovaný FXLMS algoritmus provádí výpočet koeficientů adaptivního filtru podle následující rovnice:

$$\vec{w}(n+1) = \vec{w}(n) - \mu \cdot e(n) \cdot \frac{\vec{f_x}(n)}{\left\|\vec{f_x}(n)\right\|^2} \tag{2.22}$$

2.6 Algoritmus s rekurzivní optimální adaptací (RLS)

V porovnání s algoritmem LMS tento algoritmus vyniká větší rychlostí konvergence a nevyskytuje se u něj problém činitele podmíněnosti. Nicméně algoritmus RLS vyžaduje komplikované matematické operace a má vyšší nároky na výpočty než algoritmus LMS.

Standardní RLS

Standardní RLS algoritmus k výpočtu koeficientů adaptivního filtru provádí následující výpočty:

1. Výpočet výstupního signálu adaptivního filtru.
2. Výpočet chybového signálu e(n) pomocí tohoto rozdílu: $e(n) = d(n) - y(n)$.
3. Upravení koeficientů filtru použitím následující rovnice:

$$\vec{w}(n+1) = \vec{w}(n) + e(n) \cdot \vec{K}(n) \tag{2.23}$$

kde $\vec{w}(n)$ je vektor koeficientů filtru a $\vec{K}(n)$ je vektor násobitele. $\vec{K}(n)$ je definován touto rovnicí:

$$\vec{K}(n) = \frac{P(n) \cdot \vec{u}(n)}{\lambda + \vec{u}^T(n) \cdot P(n) \cdot \vec{u}(n)} \tag{2.24}$$

kde λ je zapomínací faktor a $P(n)$ je inverzní korelační matice vstupního signálu. $P(n)$ má počáteční hodnotu $P(0)$:

$$\mathbf{P}(0) = \begin{bmatrix} \delta^{-1} & & & 0 \\ & \delta^{-1} & & \\ & & \ddots & \\ 0 & & & \delta^{-1} \end{bmatrix} \tag{2.25}$$

kde δ je regulační faktor. Standardní RLS používá následující rovnici pro výpočet inverzní korelační matice:

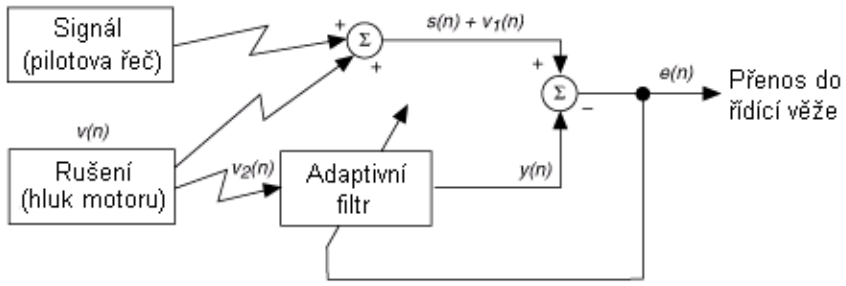
$$\mathbf{P}(n+1) = \lambda^{-1} P(n) - \lambda^{-1} \vec{K}(n) \cdot \vec{u}^T(n) \cdot P(n) \tag{2.26}$$

3 APLIKACE ADAPTIVNÍCH FILTRŮ

Na následujících stranách budou představeny zástupci metod využití adaptivních filtrů. Základem každé z nich je svým způsobem upravený adaptivní filtr přizpůsobený pro danou oblast aplikace. Některé z aplikací si dále představíme podrobněji v praktické části a ve virtuálních instrumentech v programu LabView. Úlohy využívají různé algoritmy, které jsme si popsali v předchozích kapitolách. Hlavními typy algoritmů zde popsaných jsou ty z rodiny LMS (Least Mean Squares), které jsou zároveň nejstarší a nejvíce používané.

3.1 Adaptivní potlačování rušení (ANC)

Účelem adaptivního potlačování rušení (Adaptive Noise Cancellation „ANC“) je zkvalitnit poměr signálu k šumu (SNR) odstraněním šumu na pozadí z přijatého signálu. Typickým příkladem využití této aplikace je komunikace pilota sedícího v tryskovém letadle a pozemní řídící věže. Motor letadla je schopný vyvinout hluk o síle až 140 dB, ale normální lidská řeč má energii menší než 50 dB. Pokud byste byli řídící věži, těžko byste rozuměli pilotovi v letadle čistě a jasně. V této situaci využijete adaptivní filtr, abyste mohli rozumět, co pilot říká. Následující diagram ukazuje systém potlačení rušení například motoru v letadle.



Obr. 3.1: Blokové schéma adaptivního potlačování rušení

Pilotova řeč $s(n)$ je signál, který chceme získat. Ale signál $s(n)$ nemůžeme získat přímým způsobem. Je možné pouze získat $s(n) + v_1(n)$, kde $v_1(n)$ je hluk motoru letadla. Nemůžeme získat ani $v_1(n)$ přímo. Abychom odstranili $v_1(n)$ ze signálu $s(n) + v_1(n)$, můžeme použít adaptivní filtr.

K získání hluku z letadla $v_2(n)$ musíme použít snímací senzor a tento signál dále necháme projít do adaptivního filtru. Pokud srovnáte tento diagram se schématem adaptivního filtru, pak signál $s(n) + v_1(n)$ odpovídá $d(n)$ a $v_2(n)$ odpovídá $x(n)$.

Podle diagramu, pokud není řeč $s(n)$ korelovaná s hlukem motoru letadla $v(n)$, a pokud oba signály $v_1(n)$ a $v_2(n)$ jsou velmi korelované s hlukem $v(n)$, pak adaptivní systém k odstranění rušení bude schopen odhadnout snímaný hluk motoru $v_1(n)$ tím, že iterativním přístupem upraví koeficienty adaptivního filtru. Když se výstupní signál $y(n)$ začne blížit $v_1(n)$, systém je schopen odstranit hluk motoru. Na diagramu výstup $e(n)$ označuje výsledný signál který se blíží signálu řeči $s(n)$.

3.2 Adaptivní potlačování echa (AEC)

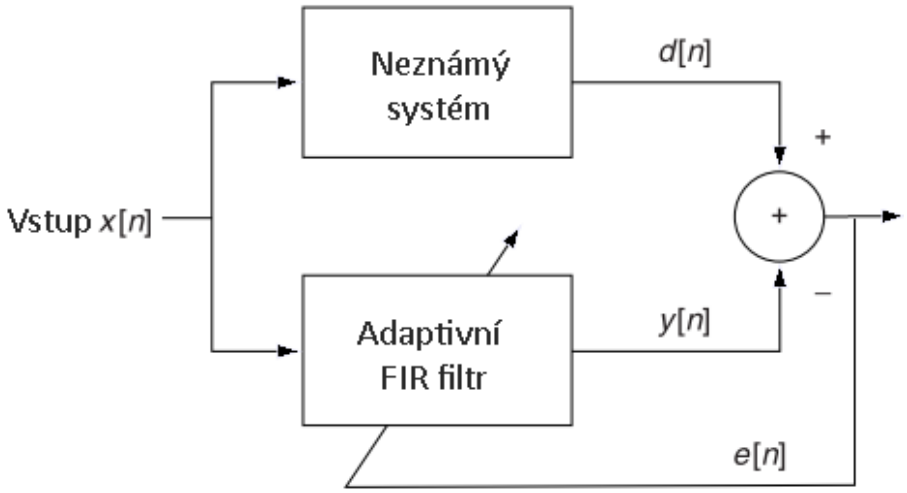
Smyslem potlačování echa (Adaptive Echo Cancellation „AEC“) je odstranění ech z komunikačních systémů. Linkové echo, také známé jako echo sítě, a akustické echo jsou dva různé typy echa. Linkové echo nastává na telefonních linkách vlivem nedokonalostí obvodů a impedančních nesrovnalostech mezi telefonními okruhy. Akustické echo vzniká například při audio konferencích a hands-free telefonování. Zvuk z reproduktoru telefonu je telefonem přijat a přenesen mikrofonem zpět.

Existují dva typické zdroje akustického echa. Akusticky izolované echo vzniká, když zdroj zvuku a mikrofon jsou od sebe slabě izolovány jeden od druhého. V moderních přístrojích vzniká například na nekvalitně navržených handsetech, headsetech a bluetooth headsetech. Druhým typem akustického echa je echo z okolí. Tento typ akustického echa vzniká, pokud hovor probíhá v akusticky odrazivém prostředí. V tomto případě mikrofon v headsetu nejdřív přijme původní zvukový projev a potom i následující signál odražený od zdí. Akustické echo okolí je běžným jevem u hands free souprav.

Dále se budeme aplikací potlačování echa zabývat v jedné z praktických úloh.

3.3 Přímá identifikace systému

Smyslem přímé identifikace systému (System Identification) je najít matematický model neznámého dynamického systému použitím vstupního stimulujícího signálu a výstupní odezvy na tento signál. Schéma na obrázku ilustruje identifikační proces, kde se využívá adaptivní filtr.



Obr. 3.2: Blokové schéma přímé identifikace

Na obrázku je znázorněn vstupní signál $x[n]$, který budí neznámý systém i adaptivní filtr. Signál $d[n]$ je odezva neznámého systému a signál $y[n]$ je výstup adaptivního filtru, signál $e[n]$ je chybový signál, jež určuje rozdíl mezi $d[n]$ a $y[n]$.

Během každé iterace adaptivní filtr upraví koeficienty filtru tak, aby minimalizoval velikost chyby $e[n]$. Výsledkem je, že chybový signál $x[n]$ se stává menším a přenosové vlastnosti adaptivního filtru se blíží vlastnostem původně neznámého systému.

Praktickou aplikací přímé identifikace systému se budeme blíže zajímat v laboratorní úloze se stejným názvem.

3.4 Aktivní potlačení rušení (AN Control)

Účelem aplikace aktivního potlačení rušení (Active Noise Control „AN Control“) je generovat anti-rušící signál reproduktorem v místě hluku tak, abychom snížili úroveň původního hluku. Tradiční metody potlačování rušení využívají pasivní přístup k minimalizaci úrovně hluku. Například automobilový průmysl provádí snižování hluku v kabinách aut použitím pasivních tlumičů, které pohlcují hluk motoru. Tradiční techniky pohlcování rušení používají metody, které snižují hluk na středních a vysokých kmitočtech. Jenže tyto metody neumí odstraňovat hluk na nízkých kmitočtech. Pomocí aktivního potlačení rušení lze vytvořit anti-rušící signál se stejným nízkým kmitočtem jako rušící signál. Fáze anti-rušícího signálu je opačná k signálu rušení. V momentě, kdy se oba signály setkají, uplatní se princip superpozice a oba signály se navzájem vyruší.

V tomto systému k potlačování rušení potřebujeme dva mikrofony a jeden reproduktor. Jeden mikrofon umístíme ke zdroji rušení a druhý mikrofon dáme do zvukového pole, kde chceme potlačit hluk. Mikrofon blíže zdroje rušení označujeme jako primární mikrofon. Mikrofon ve zvukovém poli označujeme chybový mikrofon. Použitý reproduktor určený k přenosu anti-rušícího signálu z adaptivního filtru nazýváme řídící reproduktor.

Výstup adaptivního filtru se značuje $y[n]$. Během procesu potlačování rušení adaptivní filtr upravuje své koeficienty a přenáší výstupní signál do řídícího reproduktoru. Pro tuto aplikaci se využívá x-filtrující LMS (Filtered-x Least Mean Squares „FXLMS“) adaptivní filtr, který se liší od typického adaptivního filtru především rozdíly ve vstupních signálech. Uvědomme si, že nemá vstupní signál $d[n]$. Namísto toho je vstupem filtru chybový signál $e[n]$. Tento chybový signál je snímán sekundárním chybovým mikrofonem.

Cesta z výstupu adaptivního filtru do zvukového pole, kde je umístěn chybový mikrofon se nazývá sekundární cesta, která se obvykle seskládá z DA převodníku, řídícího reproduktoru, akustické cesty z řídícího reproduktoru do chybového mikrofonu a AD převodníku. DA (číslicově analogový) převodník mění digitální signál $y(n)$ na analogový signál proudící do řídícího reproduktoru. AD (analogově číslicový) převodník mění analogový signál snímaný chybovým mikrofonem na digitální signál.

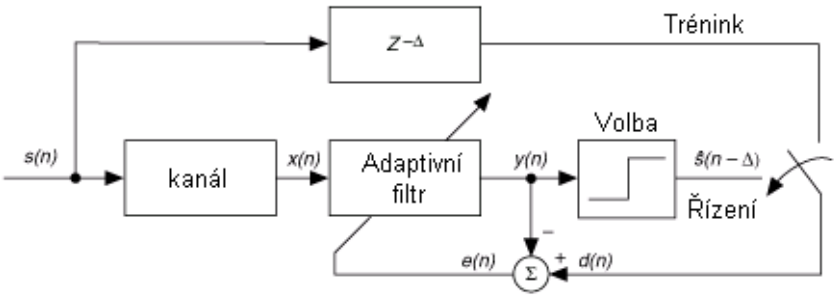
$\hat{S}(z)$ je impulzní odezva sekundární cesty. Adaptivní filtry neodhadují impulzní odezvu sekundární cesty. Musíme odhadnout impulzní odezvu sekundární cesty ještě před implementací systému potlačování rušení.

Dále se budeme touto metodou zabývat v laboratorní úloze aktivní potlačení rušení. Tam nalezneme i blokové schéma aplikace.

3.5 Adaptivní ekvalizace kanálu (ACE)

Účelem adaptivní ekvalizace kanálu (Adaptive Channel Equalization „ACE“)je vykompenzovat narušení signálu v komunikačním kanále. Komunikační systémy přenáší signál z jednoho bodu do druhého přes komunikační kanál. Kanál je realizován elektrickým kabelem, optickým kabelem nebo bezdrátovým rádiovým spojem. Během přenosu signál obsahující informaci může být narušen. K tomu, abyste vykompenzovali toto rušení můžete použít adaptivní filtr na komunikační kanál. Adaptivní filtr pracuje jako adaptivní ekvalizátor kanálu.

Následující obrázek znázorňuje schéma systému pro adaptivní ekvalizaci kanálu.



Obr. 3.3: Blokové schéma adaptivní ekvalizace kanálu

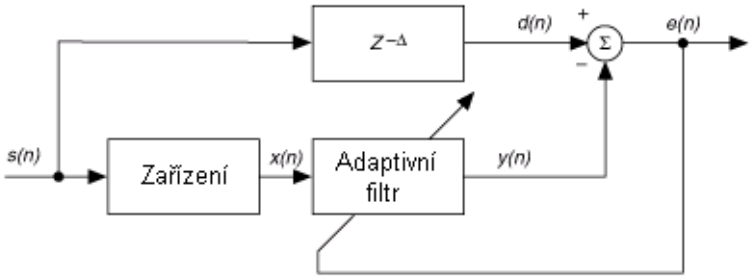
Popis diagramu: $s(n)$ je signál, který přenášíme přes komunikační kanál a $x(n)$ je narušený výstupní signál. Abychom vykompenzovali zarušení, systém adaptivní ekvalizace kanálu musí projít následujícími módy:

1. Trénovací mód Tento mód pomáhá určit vhodné koeficienty adaptivního filtru. Pokud přenášíme signál $s(n)$ přes komunikační kanál, také aplikujeme zpožděnou verzi toho stejného signálu do adaptivního filtru. Na obrázku $z^{-\Delta}$ značí zpožďovací funkci a $d(n)$ je zpožděný signál. Signál $y(n)$ je výstup adaptivního filtru a $e(n)$ je chybový signál neboli rozdíl mezi $d(n)$ a $y(n)$. Adaptivní filtr iterativně upravuje své koeficienty, aby minimalizoval chybu $e(n)$. Poté co energie signálu $e(n)$ konverguje k minimu, signály $y(n)$ a $d(n)$ se stanou skoro identické, což znamená, že můžeme využít výsledné koeficienty adaptivního filtru ke kompenzaci signálového rušení.

2. Rozhodovací mód Poté, co určíme hledané koeficienty adaptivního filtru, můžeme přepnout systém adaptivní ekvalizace do rozhodovacího módu. V tomto módu systém adaptivní ekvalizace dekóduje signál $y(n)$ a vytváří nový signál $\hat{S}(n-\Delta)$, který je odhadem signálu $s(n)$ až na zpoždění o Δ vzorků.

3.6 Adaptivní řízená inverze (AIC)

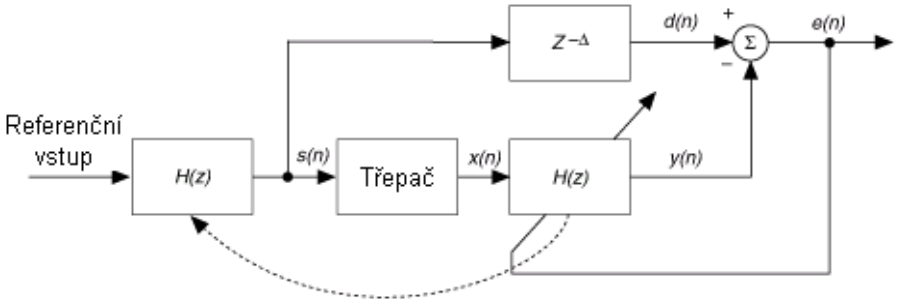
Účelem adaptivní řízené inverze (Adaptive Inverse Control „AIC“) je vytvořit inverzní model neznámého zařízení v řídícím systému. Můžeme kontrolovat neznámý systém umístěním inverzního modelu zařízení. Následující schéma znázorňuje systém adaptivní řízené inverze.



Obr. 3.4: Blokové schéma adaptivní řízené inverze

Na schématu stimulující signál $s(n)$ budí neznámé zařízení. Zařízení generuje signál $x(n)$, který je vstupním signálem adaptivního filtru. Systém adaptivní řízené inverze také vytváří zpožděný signál $d(n)$ ze stimulačního signálu a porovnává $d(n)$ s $y(n)$, což je výstupní signál adaptivního filtru. Funkce $z^{-\Delta}$ je funkce zpoždění, $e(n)$ je chybový signál. Adaptivní filtr iterativně upravuje své koeficienty, aby minimalizoval energii $e(n)$. Když energie $e(n)$ dosáhne minimální hodnoty, adaptivní filtr představuje inverzní model neznámého zařízení.

Jedním příkladem adaptivní řízené inverze jsou vibrační zkoušky, které testují spolehlivost výrobků, které jsou vystaveny prostředí s velkým množství vibrací. Například můžeme použít simulátor silnice, abychom ověřili, zda spotřební zboží může přežít silniční dopravu. Většina simulátorů silnic jsou třepače, které simulují vibrace v dopravním prostředí tím, že kopírují průběh vibrací, které lze zaznamenat v‘reálném dopravním prostředí. Pokud aplikujeme záznam vibrace na zařízení třepače, zjistíme, že tento třepač není schopen vytvořit identické vibrace kvůli dynamickým vlastnostem třepače. V tomto případě můžeme využít adaptivní filtr k řízení třepače, aby vytvářel identické vibrace k těm zaznamenaným v reálném světě. Následující schéma znázorňuje vibrační testování.



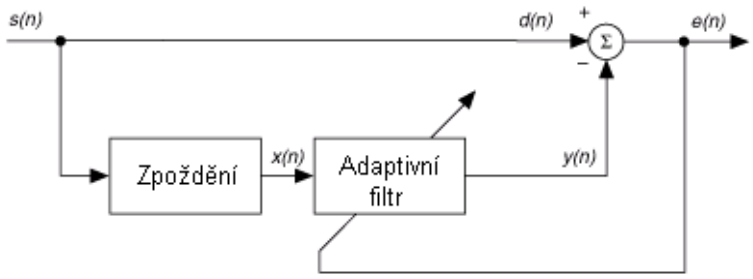
Obr. 3.5: Blokové schéma vibračního testování

Toto schéma je modifikovaná verze schématu adaptivní řízené inverze. Podle obrázku je referenční vstupní signál křivka, kterou očekáváme, že třepač vyrobí a $H(z)$ je inverzní přenosová funkce třepače. Pokud pošleme referenční vstupní signál do $H(z)$ a použijeme výstupní signál z $H(z)$ k simulaci třepače, $x(n)$ bude identický signál ke vstupnímu signálu až na zpoždění o Δ vzorků.

3.7 Adaptivní lineární predikce (ALP)

Účelem adaptivní lineární predikce (Adaptive Linear Prediction „ALP“) je využít adaptivní filtr k odhadu budoucích hodnot signálu založeném na předchozích hodnotách signálu. Adaptivní lineární predikce je taky velmi užitečná pro kompresi řečových a obrazových dat, například metodu lineárního predikčního kódování (LPC).

Následující diagram znázorňuje schéma adaptivního lineárního predikčního systému.



Obr. 3.6: Blokové schéma adaptivní lineární predikce

Popis schématu: signál $s(n)$ je posloupnost v čase n a $x(n)$ je zpožděná verze $s(n)$, což je vstupní signál adaptivního filtru, $y(n)$ je výstupní signál adaptivního filtru. Systém adaptivní lineární predikce počítá chybový signál $e(n)$, což je rozdíl mezi $s(n)$ a $y(n)$. Systém iterativně upravuje koeficienty adaptivního filtru, aby minimalizoval energii chyby $e(n)$. Pokud energie $e(n)$ dosáhne minimální hodnoty, pak adaptivní filtr předpovídá časovou posloupnost založenou na minulých hodnotách.

Adaptivní lineární predikce je způsobem jak odhadnou autoregresní (AR) modely neznámých zařízení. Můžeme vynásobit koeficienty adaptivního filtru hodnotou -1 a získáme AR koeficienty neznámého zařízení. Rozdílem mezi adaptivní lineární predikcí a jinými metodami odhadu AR modelů je ten, že adaptivní lineární predikce může pracovat v online režimu. Například můžeme využít adaptivní lineární predikci k detekci klepání motoru, které vzniká při selhání zapalování. Aplikací adaptivní lineární predikce na vibrační signál z motoru, můžeme sledovat amplitudu chybového signálu v reálném čase. Pokud amplituda chybového signálu $e(n)$ vykazuje přechodné změny, tyto přechodné změny značí, že v motoru dochází ke klepání.

4 LABORATORNÍ ÚLOHY

Hlavním cílem této diplomové práce je vypracovat laboratorní úlohy k číslicovému zpracování signálů. Úlohy se týkají oblasti adaptivního filtrování. Kladen je důraz na to, aby bylo možné využít externí zdroj signálu, například mikrofon. Mělo by být možném pokud to programové prostředí dovolí, aby úloha pracovala v reálném čase.

V této práci uvedeme 3 na sobě nezávislé jedinečné aplikace využití adaptivních filtrů. Jsou to:

- Adaptivní potlačení echa
- Aktivní potlačení rušení
- Přímá identifikace systému

Všechny úlohy mají propracované uživatelské rozhraní. Lze je spustit přímo v LabView (soubory s příponou .vi) anebo jako samostatně spustitelnou aplikaci s pomocí LabView run-time engine. Tato možnost je žádoucí, protože spustitelná aplikace je předkompilována a proto svižnější než samotný zdrojový kód. Další výhodou je nemožnost zasáhnout do zdrojového kódu. Nicméně součástí této diplomové práce jsou jak spustitelné .exe soubory, tak i zdrojové kódy jednotlivých úloh.

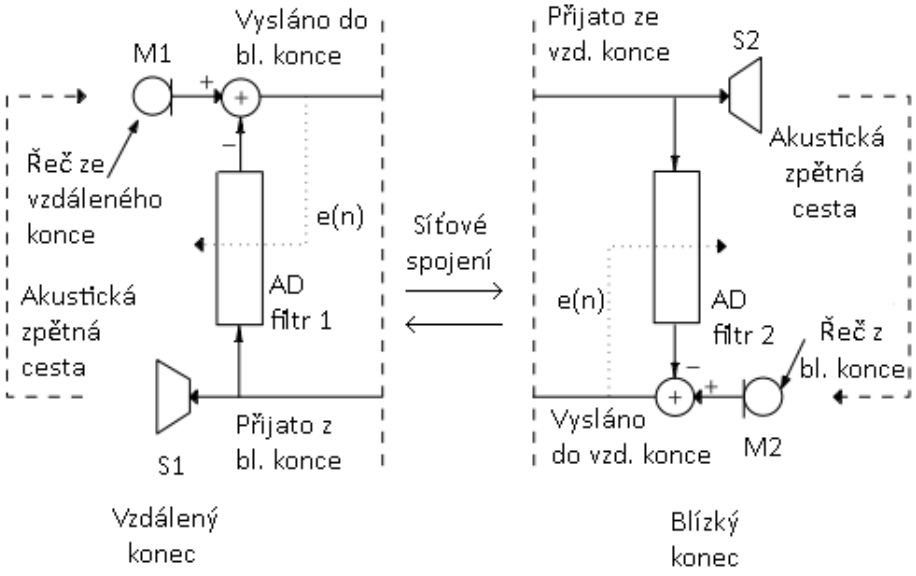
4.1 Adaptivní potlačení echa

4.1.1 Úvod

Cílem úlohy je seznámit se a vyzkoušet aplikaci adaptivní potlačování echa. Příklad aplikace je vytvořen v LabView s použitím normalizovaného LMS (NLMS) filtru. Součástí úlohy je možnost nahrát vlastní řečový záznam, na kterém lze provádět experimenty s potlačováním echa. Vlastní řečový záznam lze uložit do souboru ve (.wav) formátu . Lze uložit nahrávku před potlačením i po potlačení echa. K práci s úlohou lze stejně tak dobře použít i externí zvukové soubory ve (.wav) formátu. Pro tuto úlohu nebylo vhodné realizovat potlačování v reálném čase, protože v LabView nedosáhneme optimálních výsledků.

4.1.2 Teoretický základ

Účelem potlačení akustického echa je odstranění echa z komunikačního systému. Linkové echo, nebo také echo na síti, a akustické echo jsou dva různé typy echa, které vznikají na koncích spoje. Linkové echo vzniká na telefonních linkách vlivem nedokonalostí a impedančních nerovnováhách na telefonních okruzích. Akustické echo vzniká zpětným snímáním zvuku z reproduktoru na blízkém konci komunikačního systému. Budeme se zde zabývat pouze tématem potlačování akustického echa. Následující diagram popisuje potlačení akustického echa.



Obr. 4.1: Schéma adaptivního potlačování echa

Schéma zobrazuje vznik akustického echa v komunikačním systému. Řeč osoby na blízkém konci ("Near End") je snímána mikrofonem M2 a přenesena do repro-

duktoru S1 na vzdáleném konci ("Far End"). Mikrofon M1 potom sejme hlas z reproduktoru S1 a přenese jej zpět do blízkého konce na reproduktor S2. Když osoba na blízkém konci mluví do mikrofonu M2, slyší svůj vlastní hlas z reproduktoru S2. Zkreslený a zpožděný signál hlasu je akustické echo, které negativně ovlivňuje komunikaci, pokud je příliš velké. Na každém konci systému je použit adaptivní filtr, který odhaduje přenosové vlastnosti mezi mikrofonem a odpovídajícím reproduktorem. Jak je znázorněno na schématu, vyslaný signál z mikrofonu M2 na blízkém konci po přenosu zároveň vstupuje jak do reproduktoru S1, tak do adaptivního filtru na vzdáleném konci. Adaptivní filtr pak upraví své koeficienty tak, aby odhadl parametry zkreslené a zpožděné řeči. Pokud je odhadovaný signál podobný tomu snímanému z mikrofonu M1, pak echo na reproduktoru S2 bude potlačeno a nebude slyšet.

Metrika používaná k určení hlasitosti echa na lince je Echo Return Loss (ERL), někdy se používá i název Echo Path Loss. ERL se vyjadřuje v jednotkách dB (decibel) relativně k úrovni hlasitosti původního signálu.

$$\mathrm{ERL}(dB) = \mathrm{Puvodni\_signal}(dBm) - \mathrm{Echo}(dBm) \tag{4.1}$$

Nízká hodnota ERL znamená hlasitější echo, zatímco vysoká hodnota ERL odpovídá slabému echu. Například ERL o velikosti 0 dB znamená stejně silné echo jako byl původní signál (v praxi nemožné), zatímco ERL 55 dB je velmi tiché echo.

Echo Return Loss Enhancement

Doporučení ITU-T G.168 definuje Echo Return Loss Enhancement (ERLE) jako "Utlumení signálu echa když vychází ze zařízení k potlačení echa. Tato definuje specificky vylučuje jakékoli nelineární zpracování signálu na výstupu zařízení k potlačení echa, které by měly za následek další utlumení echa."

Například pokud echo na lince má ERL 15 dB a zařízení k potlačení echa je schopno 30 dB, pak výsledné zbytkové echo bude mít o (15 + 30) 45 dB méně, než původní signál před nelineárním zpracováním.

Celkovou míru potlačení echa udává hodnota Residual Echo Return Loss (RERL), která znamená relativní úroveň hlasitosti potlačeného echa k úrovni původního signálu. Platí:

$$\mathrm{RERL}(dB) = \mathrm{ERL} + \mathrm{ERLE} \tag{4.2}$$

Křížová korelace

Křížová korelace je standardní metoda výpočtu, jak moc se dva signály podobají pokud jeden z nich je v čase posunutý. Při zpracování signálů je křížová korelace mírou podobnosti dvou signálů.

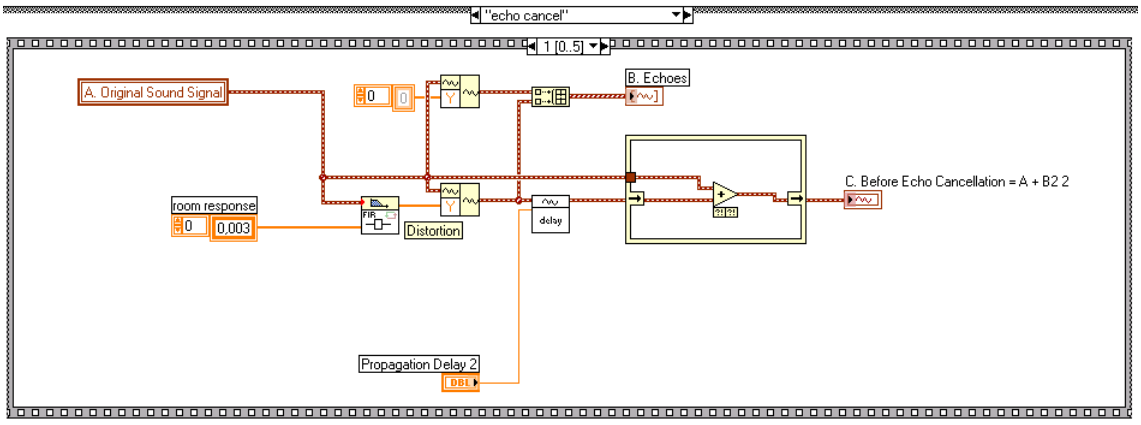
$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f * [m]g[n+m] \tag{4.3}$$

Při autokorelaci, což je obdoba křížové korelace signálu se sebou samým, bude špička vždy v bodě 0, pokud signál není nulový.

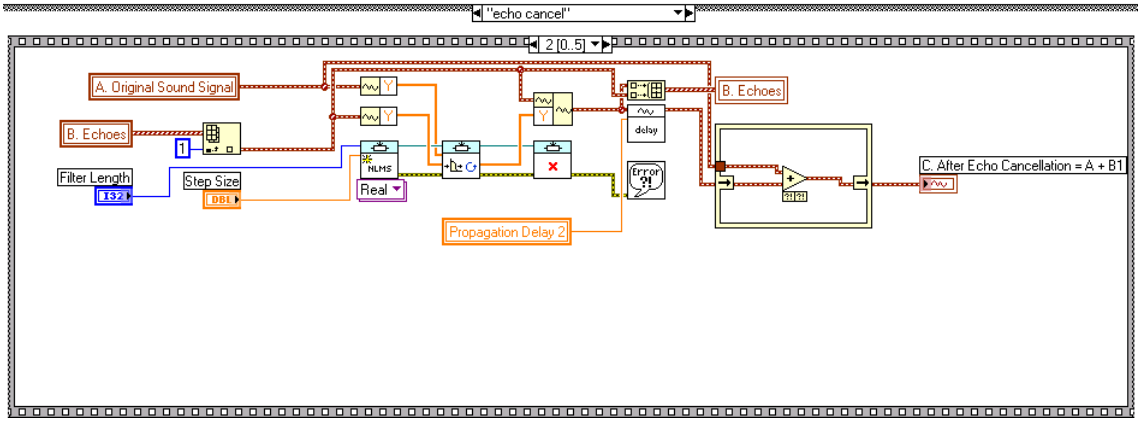
4.1.3 Realizace

K realizaci této úlohy byl použit NLMS filtr. Je součástí sady doplňků programu LabView, Adaptive Filter Toolkit (AFT). V tomto příkladu nemáme k dispozici zdroj řečového signálu z blízkého konce. Místo toho použijeme simulaci odezvy místnosti na daný signál s aditivním šumem, čímž simulujeme požadovaný signál. Odezva místnosti je zde simulována odezvou filtru FIR (120 koeficientů) na vstupní signál. Ze vstupního signálu získáme filtrací utlumený signál, dále jej zpozdíme o hodnotu "Propagation Delay". Tento a původní signál sečteme a takto vzniklý signál je vstupní signál adaptivního filtru. Na tento signál aplikujeme adaptivní filtraci NLMS algoritmem. Důležitým faktorem je velikost kroku NLMS algoritmu (Step size), který lze nastavit před adaptivní filtrací.

Bloková schémata produkce echa a jeho potlačení jsou na následujících schématech:



Obr. 4.2: Blokový diagram produkce echa



Obr. 4.3: Blokový diagram potlačení echa

4.1.4 Postup práce

Ke splnění úlohy postupujte následujícími kroky:

1. !Volitelné! Na záložce "Acoustic Echo Apply" spusťte nahrávání vlastní řeči. Definujte délku nahrávky Duration (s) a spusťte nahrávání stisknutím "Record Wave File". Nahraný zvuk si můžete přehrát stisknutím ikony reproduktoru u grafu časového grafu nahrávky.

2. !Volitelné! Zvolte zpoždění echa táhlem "Propagation Delay" a stiskněte "Add echo". Tímto vložíte echo do nahrávky. Opět si jej můžete přehrát. Všimněte si hodnoty ERL (Echo Return Loss), udává rozdíl energie původního signálu a odvozeného echa.

3. !Volitelné! Původní nahrávku lze jako soubor wave uložit na disk. Stiskněte tlačítko "Save original speech".

4. Přepněte záložku na "Adaptive Echo Cancellation". Nahrajte zvukový soubor ve formátu wave stisknutím "Open Wave File"

5. Nastavte velikost kroku adaptivního LMS filtru "Step Size". Doporučené hodnoty jsou v rozmezí 0,1 až 0,5.

6. Zvolte délku adaptivního filtru. "Filter Length". Tuto hodnotu není nutné měnit, pokud nemusíte, ponechte implicitní nastavení. Doporučené hodnoty jsou

128 až 1024.

7. Zvolte zpoždění echa "Propagation Delay 2". Tato hodnota nemá na adaptaci filtru vliv. Zpoždění echa je zřejmé při přehrání zvukového signálu. Volte mezi 0,2 a 0,6 s.

8. Stiskněte tlačítko "Apply Echo Cancellation" pro aplikaci LMS filtru k odstranění echa. Všimněte o kolik se snížila energie echa na ukazateli ERLE (Echo Return Loss Enhancement)

9. Změňte nastavení a opakujte odstranění echa. Celkové míra potlačení Echa udává RERL (Residual Echo Return Loss) [dB].
   Platí ERL + ERLE = RERL.

10. Ukončete aplikaci standardně stisknutím tlačítka Exit.

Poznámky:

a *Použijte malou velikost kroku abyste dosáhli nejmenší konečné chyby. Na druhou stranu menší krok výrazně snižuje rychlost konvergence adaptivního filtru.*

b *Zvětšete velikost kroku, abyste zvýšili rychlost konvergence. Nicméně přehnaná velikost může způsobit nestabilitu adaptivního filtru.*

c *Volba ideální délky filtru je procesem pokus-omyl. Musíte sami určit nejlepší délku pomocí simulace.*

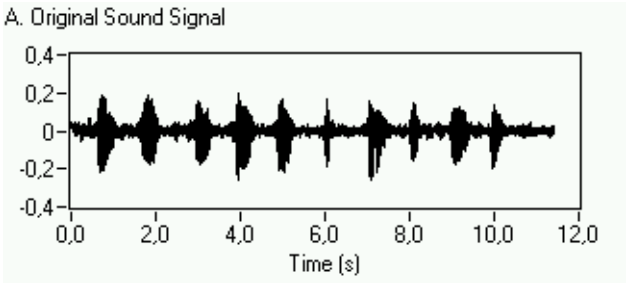
d *Délka fitru musí být delší než počet koeficientů filtru simulující neznámý systém, zde 120.*

e *Dlouhý filtr minimalizuje konečnou chybu. Ale příliš dlouhý filtr nemusí dosáhnout ideálního řešení. Dlouhý filtr také vyžaduje vysoké výpočetní nároky.*

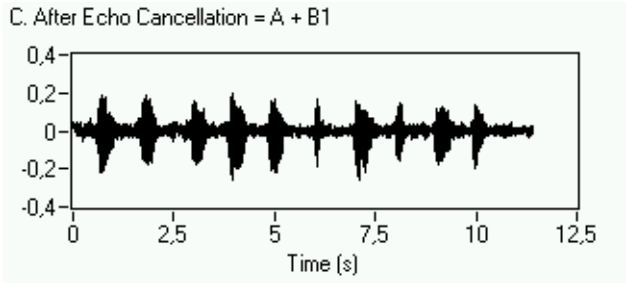
f Použijte nejmenší délku fitru, která uspokojuje požadavky aplikace. Menší délka zvyšuje rychlost konvergence, a taky šetří výpočetní zdroje.

4.1.5 Výsledky měření

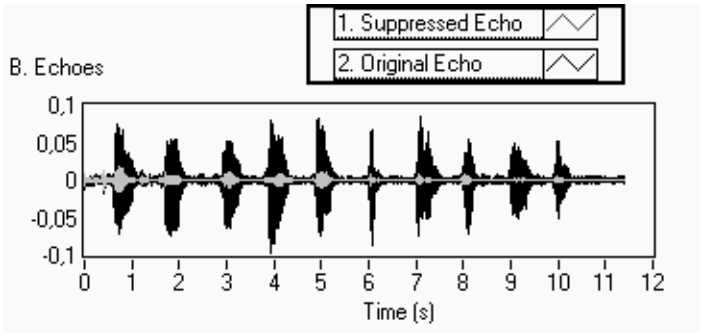
Měření lze provést na libovolné testovací nahrávce řeči či jiného zvuku. Ideální je použít nestacionární zvukový signál, například řeč. Testovacím signálem byly slova "'raz, dva, tři … deset"'. Podívejme se na grafy signálových průběhů typického měření.



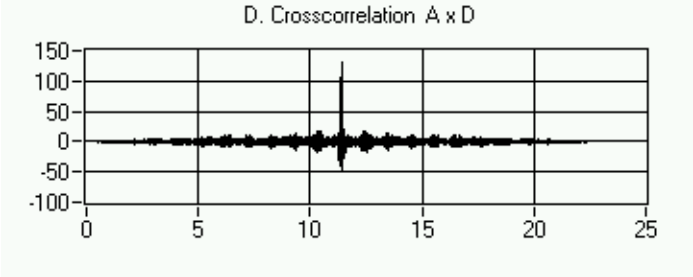
Obr. 4.4: Původní signál řeči 1,2,3 … 10



Obr. 4.5: Signál s potlačeným echem



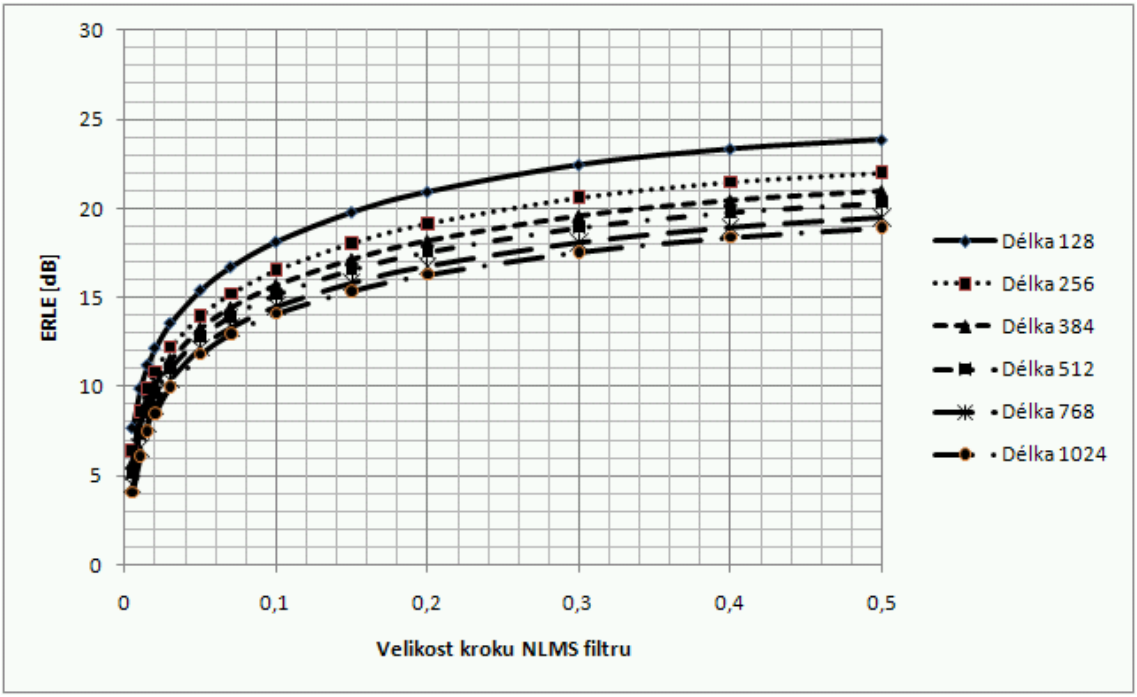
Obr. 4.6: Původní a potlačené echo



Obr. 4.7: Křížová korelace původního a filtrovaného signálu

4.1.6 Měření závislosti

V této úloze lze dobře dokázat závislost potlačení echa na délce adaptivního filtru a velikosti kroku algoritmu NLMS. Výsledky měření lze shrnout do tohoto grafu:



Obr. 4.8: Závislost ERLE [dB] na velikosti kroku, různé délky filtru

4.1.7 Shrnutí

Tato laboratorní úloha slouží k seznámení s adaptivním NLMS filtrem a jeho využití k odstranění akustického echa. Úloha poslouží k získání povědomí o možnostech adaptivního filtrování a uvede zájemce do vlastností adaptivních algoritmů. Úloha je zajímavá možností nahrát vlastní řeč a přehrávat si tento záznam s echem i po potlačení echa. Všechny dílčí zvuky lze uložit pro pozdější použití, případně analýzu.

Je zřejmé z výsledků měření, jak je schopnost adaptivního filtru potlačit echo závislá na délce filtru a velikosti kroku. Naměřená závislost je orientační, protože je vázaná na testovaný signál. Nicméně může sloužit jako opěrný bod pro ideální stanovení parametrů filtru.

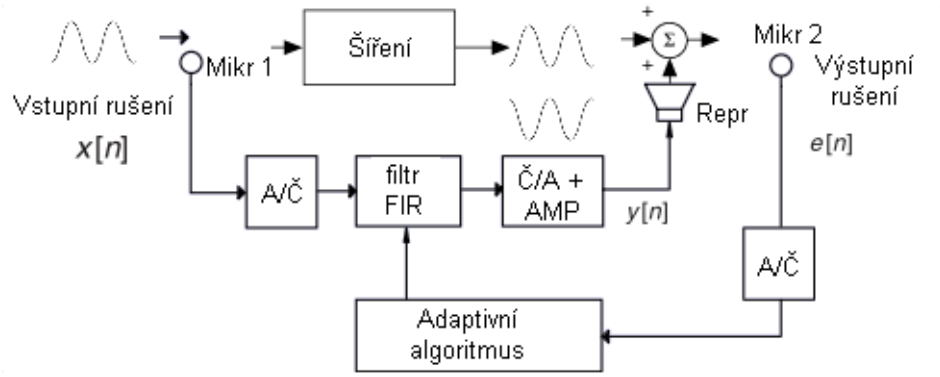
4.2 Aktivní potlačení rušení

4.2.1 Úvod

Cílem úlohy aktivní potlačení rušení je představit aplikaci adaptivního filtru k odstranění nežádoucího rušení principem superpozice. Pro tuto aplikaci je využit x-filtrující Least Mean Squares „FXLMS“ adaptivní filtr, který se liší od typického adaptivního filtru tím, že nemá vstupní signál d[n]. Namísto toho je vstupem filtru chybový signál e[n]. Součástí úlohy nahrávání řečového signálu přímo z mikrofonu. K řečovému signálu se v úloze generuje aditivní rušení, jehož amplituda a kmitočet lze podle potřeby měnit. Úloha při nahrávání řeči v reálném čase zobrazuje časový průběh řeči a hustotu energie řečového signálu i aditivního rušení. Typ rušení lze měnit, je možné vybrat signál sinusový, trojúhelník, pila, šum... Při nahrávání je možné zároveň nahrávaný signál přehrávat výstupním zvukovým zařízením. Samotné potlačování rušení nebylo vhodné zpracovat v reálném čase, protože realizací v LabView bychom nedosáhli optimálních vlastností.

4.2.2 Teoretický základ

Účelem aplikace aktivního potlačení rušení je generovat anti-rušící signál reproduktorem v místě hluku tak, abychom snížili úroveň původního hluku. Tento princip využívá přídavný zdroj signálu v místě hluku tak, aby vyrušil původní nežádoucí hluk. Další zdroj zvuku vytváří signál s opačnou fází a tak v místě rušení dochází k minimalizaci hluku. Ke generování anti-rušícího signálu lze použít adaptivní filtr. Následující obrázek znázorňuje schéma systému pro aktivní potlačování rušení.



Obr. 4.9: Schéma aktivního potlačení rušení

Na diagramu je výstup adaptivního filtru označen y[n]. Během procesu potlačování rušení adaptivní filtr upravuje své koeficenty a přenáší výstupní signál do pracovního reproduktoru. Pro tuto aplikaci je využit x-filtrující Least Mean Squares

„FXLMS“ adaptivní filtr, který se liší od typického adaptivního filtru. Všimněte si, že nemá vstupní signál d[n]. Namísto toho je vstupem filtru chybový signál e[n]. Tento chybový signál je snímán sekundárním mikrofonem. V tomto příkladu nemáme k dispozici dva mikrofonové vstupy, proto vstup mikrofonu Mikr 2 bude simulovat prostým součtem výstupních signálů. Cesta zvukového signálu z adaptivního filtru do místa, kde je mikrofon je známá pod pojmem sekundární cesta, která obvykle zahrnuje Č/A převodník, reproduktor, akustickou cestu k mikrofonu snímající chybový signál a A/Č převodník. Tuto sekundární cestu simulujeme FIR filtrem se 100 koeficienty.

Poznámka: Aktivní potlačení rušení pracuje nejlépe, pokud máme potlačovat monotónní hluk v prostorově omezeném a jednoduchém zvukovém poli.

Metrika použitá k určení průměrného potlačení rušení je Signal to Noise Ratio (SNR), jednotka je v dB. SNR udává podíl energie signál vůči energii rušení.

K určení celkové míry potlačení rušení zavedeme novou veličinu Signal to Noise Ratio Enhancement (SNRE) [db]. Bude platit:

$$\mathrm{SNRE}(dB) = \mathrm{SNR}_{after} - \mathrm{SNR}_{before} \tag{4.4}$$

kde:

SNR_{before} (dB) je poměr signálu k šumu před potlačením rušení.

SNR_{after} (dB) je poměr signálu k šumu po potlačení rušení.

4.2.3 Realizace

K vytvoření úlohy byl použit Filtered-x LMS (FXLMS) adaptivní filtr. Je součástí sady doplňků programu LabView, Adaptive Filter Toolkit (AFT). Máme k dispozici standardně pouze jeden mikrofonový vstup, proto vstup chybového signálu, mikrofonu 2 budeme simulovat součtem výstupních signálů. Odezva sekundární cesty podle schématu je v programu nahrazena filtrem typu FIR se 100 koeficienty. Zpoždění způsobené šířením signálu prostředím simulujeme použitím zpožďovacího bloku, který signál posune o 200 vzorků. Ke vstupnímu signálu řeči můžeme v úloze generovat aditivní rušení. Toto rušení je zpožděné způsobené simulovaným šířením, takto zpožděný signál je vstupem adaptivního FXLMS filtru. Výstup z adaptivního filtru je dále filtrován FIR filtrem, který simuluje sekundární cestu signálu. Výstupem sekundární cesty je anti-rušící signál, který se sčítá se zpožděným rušícím signálem. Výsledným výstupním signálem je zbytkový signál e[n], který vznikne superpozicí signálů. Toto je zbytková chyba, která narušuje užitečný signál. Důležitým faktorem je velikost kroku FXLMS algoritmu (Step size) a délka filtru. Obě hodnoty lze za běhu programu nastavit.

Blokové schéma (Active Noise Control) je zobrazeno na následujících obrázcích:

4.2.4 Postup práce

Ke splnění úlohy postupujte následujícími kroky:

1 Zvolte typ rušícího signálu (Waveform type), který budete používat. Nejvíce je vhodný sinusový signál (Sine). Nicméně máte na výběr další možnosti (Square, Sawtooth, Triangle, Gaussian White Noise, Uniform White Noise).

2 Nastavte kmitočet (Frequency) a Amplitudu rušení (Noise Volume). Kmitočet volte libovolný, hlasitost volte přiměřeně výkonu signálu řečového signálu, který snímáte mikrofonem. Spíše ovšem v dolních mezích rozsahu.

3 Pro začátek nahrávání stiskněte tlačítko RECORD. Jakmile budete spokojeni s nahrávkou, stiskněte tlačítko STOP. Upozornění: Maximální délka nahrávky je omezena na 60 s.
Poznámka: Během nahrávání můžete i současně přehrávat nahrávaný signál. Zvolte z nabídky Playback samotný vstup z mikrofonu (speech signal), vstup s rušícím signálem (speech & noise) nebo bez zvuku (no sound). Hlasitost můžete měnit táhlem Volume. Nahraný signál řečového vstupu s rušením lze přehrát kliknutím na ikonu reproduktoru.

4 Přepněte na záložku Apply Noise Control. Zde probíhá demonstrace aktivního odstranění rušení.

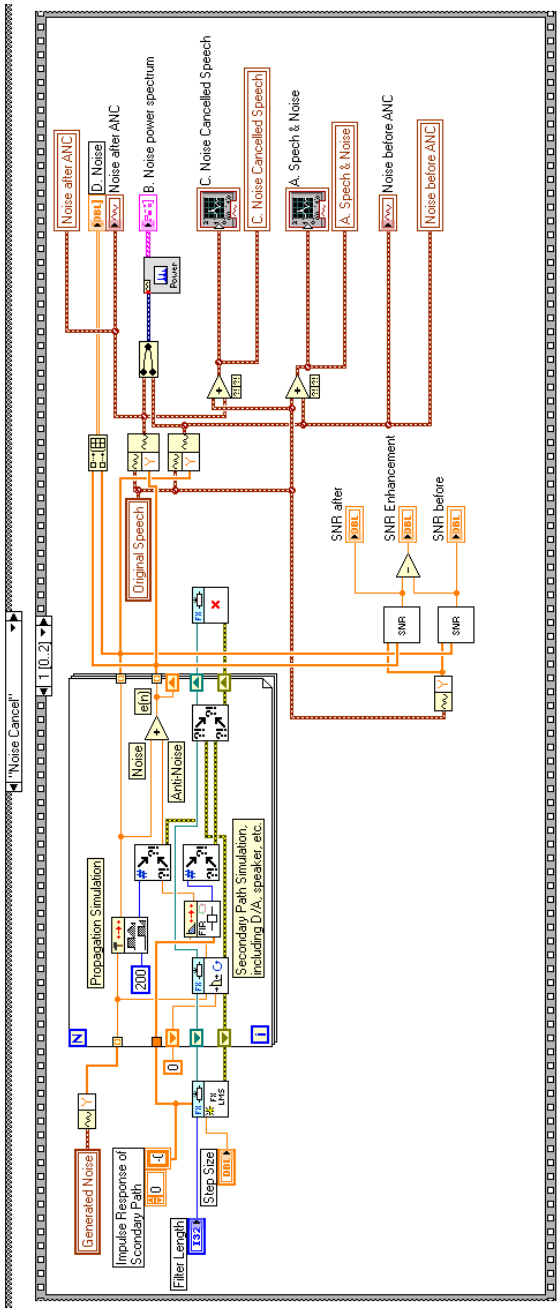
5 Nastavte délku filtru (Filter Length) FXLMS. Délka by měla být delší než délka filtru simulující sekundární cestu, tj. 100, ale není to nutné. Volte rozmezí 64 až 1024. Délku filtru pro první pokus ponechte s původním nastavením.

6 Nastavte velikost kroku (Step Size) FXLMS filtru. Pro první pokus ponechte nastavení.

7 Stiskněte tlačítko Cancel Noise. Tímto proběhne adaptace filtru na nahraný signál. Rušící signál bude minimalizován s tím, jak rychle se adaptuje FXLMS filtr. Sledujte ukazatele SNR - poměru signálu k šumu. Řečový signál po odstranění rušení by měl mít vždy větší odstup SNR.

8 Měňte nastavení velikosti kroku a délky filtru. Sledujte jak se mění rychlost konvergence adaptivního filtru a hodnoty SNR. Opakujte body 5 až 7.

9 Aplikaci ukončíte standardně tlačítkem EXIT.

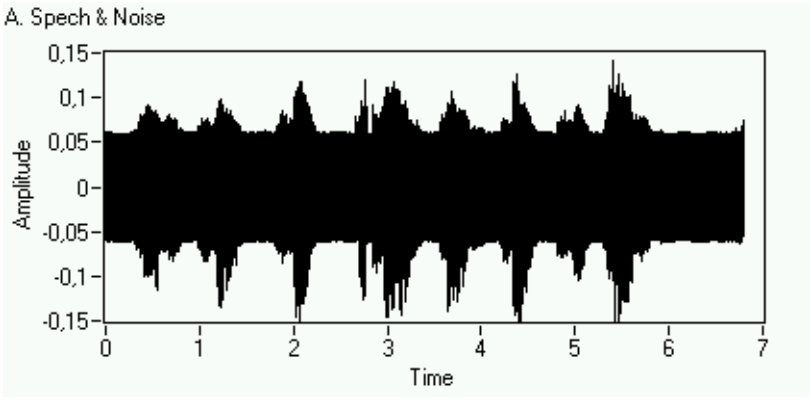


Obr. 4.10: Blokový diagram aktivního potlačování rušení

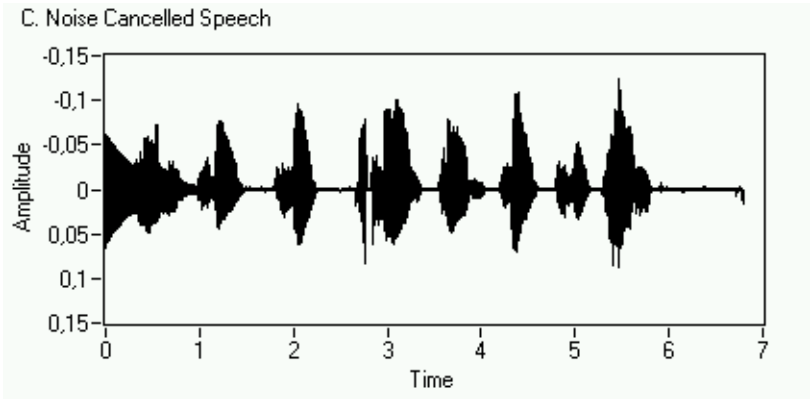
4.2.5 Výsledky měření

Měření lze provést na libovolném vstupním řečovém signálu a kombinaci aditivního rušení. Optimálních výsledků dosáhnete použitím rušení sinusovým signálem o stanovené frekvenci. Na dostatečně dlouhém (alespoň několik vteřin) záznamu lze provést realizaci potlačení rušení. Energie rušení by měla být nižší než užitečného signálu, proti volíme amplitudu rušení podle síly našeho řečového signálu. Příkladné měření bylo provedeno na náhodném řečovém signálu o délce zhruba 8 sekund.

Na následujících obrázcích pozorujme průběhy měření:



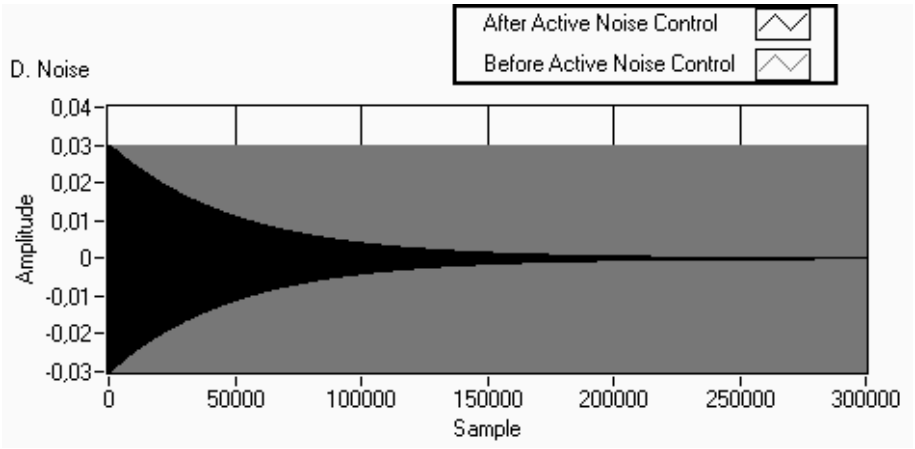
Obr. 4.11: Časový průběh řeči s aditivním rušením



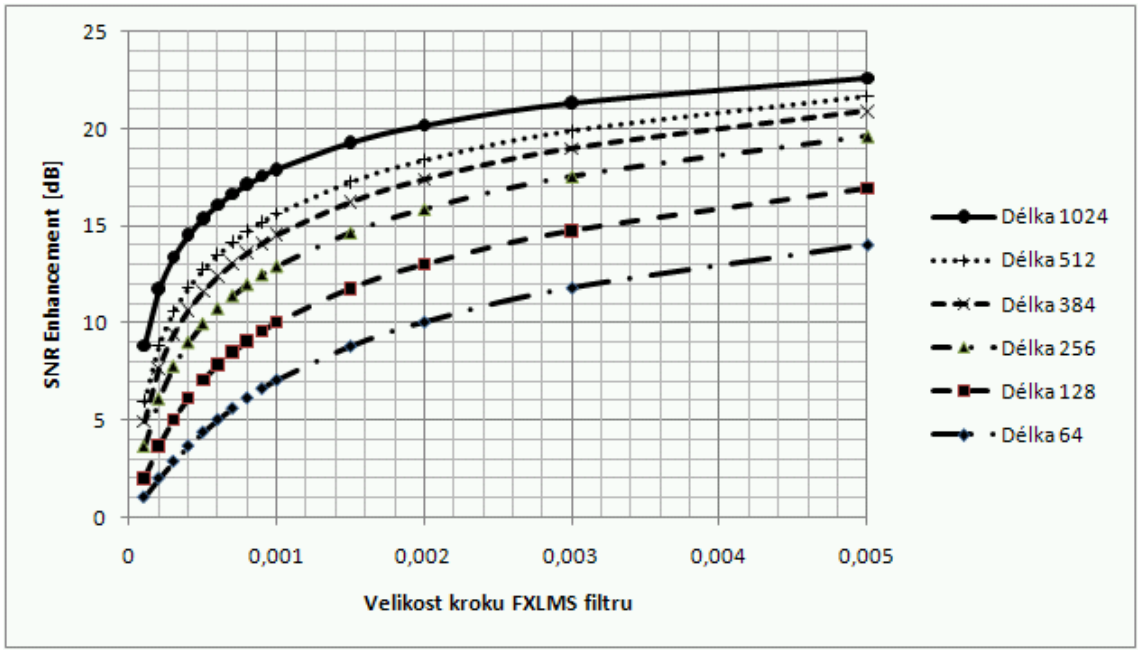
Obr. 4.12: Časový průběh signálu s potlačeným rušením

4.2.6 Měření závislosti

U této úlohy lze dobře dokázat závislost potlačení rušení na délce adaptivního filtru a velikosti kroku algoritmu FXLMS. Výsledky měření lze shrnout do grafu závislosti SNR na velikosti kroku.



Obr. 4.13: Časové průběhy rušení před a po potlačení



Obr. 4.14: Závislost SNRE [dB] na velikosti kroku, různé délky filtru

4.2.7 Shrnutí

Tato laboratorní úloha slouží k seznámení se s adaptivním FXLMS filtrem a jeho použitím na aktivní potlačení aditivního rušení. Zajímavou možností je pro realizaci měření možnost použít vlastní řeč, kdy vidíme časový průběh řeči a spektrální rozložení energie řeči a rušení během nahrávání signálu. Vlastní potlačení rušení je realizováno algoritmem FXLMS, který se liší od ostatních algoritmů svými vstupy. Úloha poskytuje uživatelsky přívětivé grafické prostředí pro analýzu funkce adaptivního filtru. Nahraný signál s rušením i signál s potlačeným rušením lze během programu přehrát. Je také možné uložit zvukový záznam do souboru .wav pro pozdější analýzu. Export grafů umožňuje samotný LabView. Měřením jsme získali závislost SNRE na velikosti kroku FXLMS algoritmu při různých délkách filtru. Naměřená závislost je orientační, protože je vázaná na testovaný signál. Nicméně může sloužit jako opěrný bod pro ideální stanovení parametrů filtru.

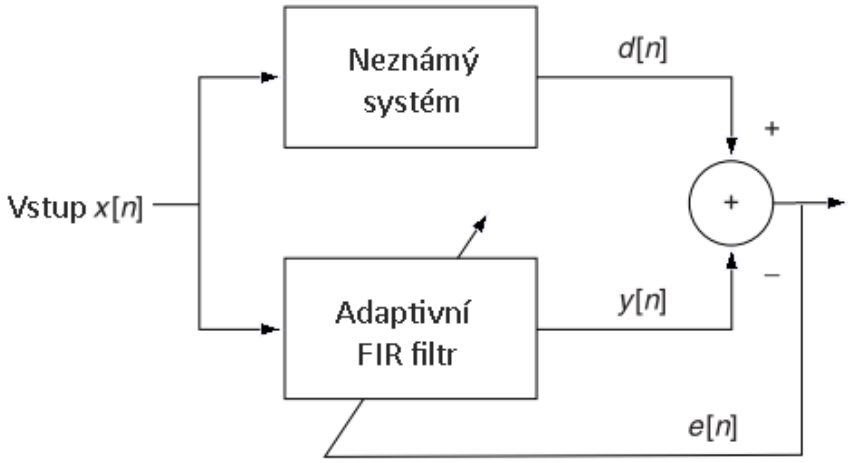
4.3 Přímá identifikace systému

4.3.1 Úvod

Cílem této úlohy je seznámit se s aplikací přímé identifikace systému. Aplikace je vytvořena v LabView pomocí LMS filtru, který je složen ze základních bloků VI. Jeho blokový diagram je zobrazen v úloze. Součástí úlohy je možnost nastavení typu neznámého systému a vstupního signálu. Úloha má dvě části. Jedna část realizuje identifikaci systému FIR. Druhá část realizuje identifikaci systému IIR filtru. Samotný LMS filtr je typu FIR, proto přímo porovnávat koeficienty filtru lze pouze v úloze identifikace systému FIR. Neznámý systém typu FIR nebo IIR lze zvolit z nabídky: dolní propust, horní propust, pásmová propust a pásmová zádrž. Je možné nastavit řád filtru systému i řád LMS adaptivního filtru. Takže je možné porovnat, jak adaptivní filtr je schopen konvergovat koeficienty, pokud má nižší či vyšší řád, než neznámý systém.

4.3.2 Teoretický základ

Smyslem aplikace je najít matematický model neznámého dynamického systému použitím vstupního stimulujícího signálu a výstupní odezvy na tento signál. Schéma na obrázku ilustruje identifikační proces, kde se využívá adaptivní filtr.



Obr. 4.15: Blokové schéma identifikace systému

Na obrázku je znázorněn vstupní signál x[n], který budí neznámý systém i adaptivní filtr. Signál d[n] je odezva neznámého systému a signál y[n] je výstup adaptivního filtru, signál e[n] je chybový signál, jež určuje rozdíl mezi d[n] a y[n]. Během

každé iterace adaptivní filtr upraví koeficienty filtru tak, aby minimalizoval velikost chyby e[n]. Výsledkem je, že chybový signál x[n] se stává menším a koeficienty adaptivního filtru se blíží koeficientům původně neznámého systému.

4.3.3 Realizace

K realizaci této úlohy byl použit LMS filtr sestavený ze základních bloků VI. Čerpali jsme z literatury . Blokové schéma LMS filtru bude zobrazeno v úloze. K návrhu filtrů a jejich analýze jsme použili Digital Filter Design Toolkit (DFDT) ze sady doplňků LabView.

Tato úloha pracuje v reálném čase. Koeficienty filtru jsou počítány ve smyčce iterativně.

Metrikou je zde použita střední kvadratická odchylka (Mean Square Error „MSE“):

$$\mathrm{MSE}(\hat{x}) = E[(\hat{x} - x)^2] \tag{4.5}$$

V úloze je zobrazena průměrná hodnota MSE, která se počítá z posledních 200 vzorků signálu.

blokové diagramy jsou zobrazeny na následujících obrázcích.

4.3.4 Postup práce

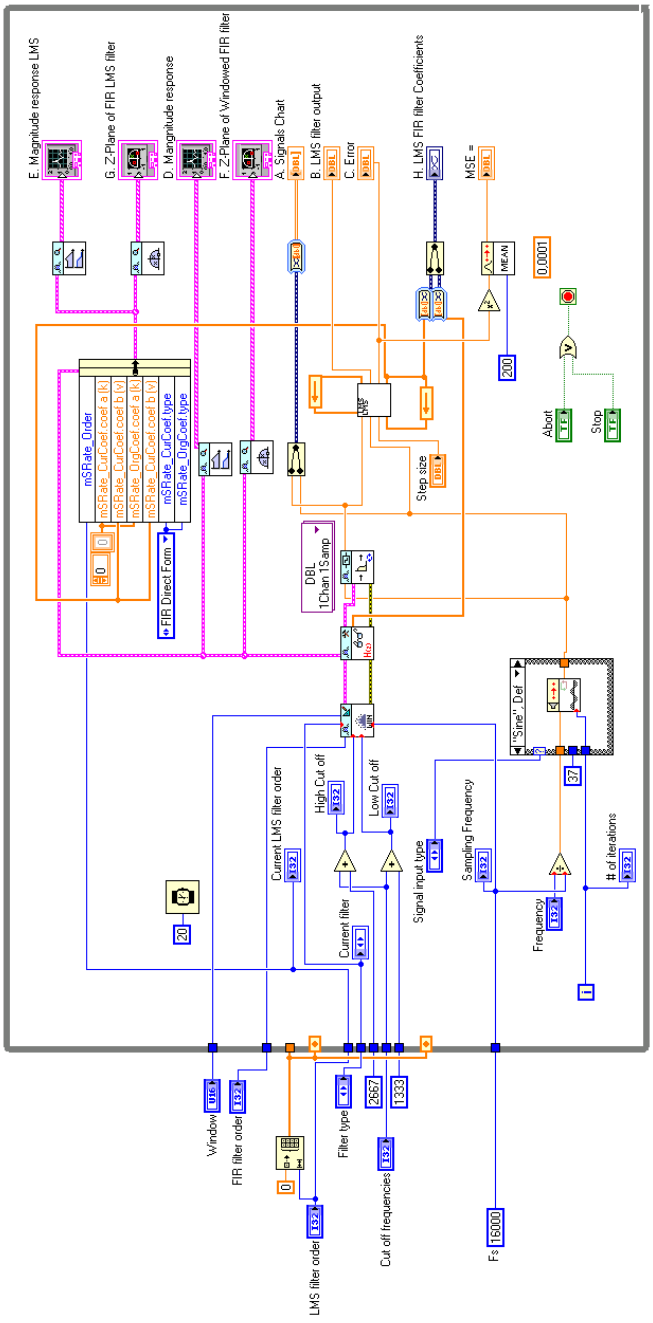
Přímá identifikace systému typu FIR

K práci s úlohou postupujte následujícími kroky:

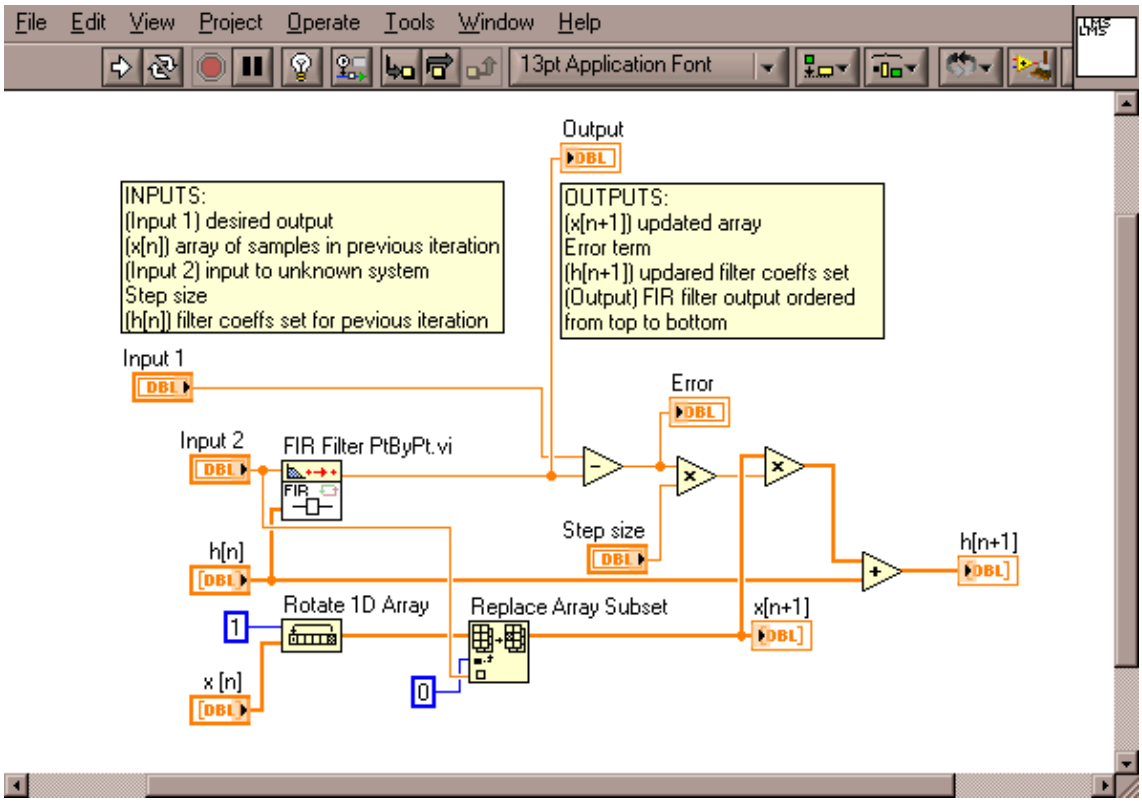
1 Nastavení parametrů FIR filtru - bude zde simulovat neznámý systém pro adaptivní filtr.
a. Zvolte typ filtru. Máte zde na výběr z nabídky: Dolní propust (Lowpass), Horní propust (Highpass), Pásmová propust (Bandpass) a Pásmová zádrž (Bandstop).
Poznámka : U FIR filtrů typu Bandstop nebo Highpass nesmí být řád filtru liché číslo! LabView ukáže chybové hlášení a ukončí program. Nastavte tedy proto u těchto typů řád filtru na sudé číslo.
b. Zvolte řád filtru (FIR filter order) podle svého uvážení. Doporučené hodnoty jsou nižší než 40. Volte nejlépe v rozmezí 4 - 32.
c. Zvolte typ okna použitého při návrhu FIR filtru. Nejlépe ponechejte nastavení na žádné (none). I žádné okno je vlastně okno.
d. Pomocí táhla upřesněte mezní frekvence. Dovolené hodnoty dovolují rozmezí až 666 Hz. Tyto hodnoty není nutné měnit.

2 Nastavení parametrů adaptivního LMS filtru.
a. Zvolte řád filtru (LMS filter order) podle vašeho přání. Měli byste se orientovat podle řádu FIR filtru.
b. Nastavte velikost kroku (Step size) pro LMS filtr. Doporučené rozmezí je 0,001 až 0,020.

3 Nastavení zdroje signálu.
a. Zdrojem signálu je zde generátor posloupnosti číslicového signálu. Lze zde nastavit jeden z pěti typů signálového vstupu: Sinusoida (Sine), Trojúhelník (Triangle), Pila (Sawtooth), Čtverec (Square) a uniformní bílý šum (Uniform White Noise).
Poznámka: Signál Sine obsahuje pouze 1 kmitočtovou složku, ostaní signály mají složek více. Bílý šum obsahuje rovnoměrně rozprostřené spektrum všech kmitočtových složek.
b. Frekvence neboli kmitočet signálu - základní harmonické složky lze volit od 0 Hz do 8000 Hz, tedy do poloviny vzorkovací frekvence 16 000 Hz.



Obr. 4.16: Blokový diagram identifikace systému typu FIR



Obr. 4.17: Blokový diagram LMS filtru VI

4 Spuštění aplikace
Zmáčkněte tlačítko Start. Některá tlačítka se stanou nedostupné. Pro ukončení momentální adaptace zmáčkněte tlačítko Stop. Po zmáčknutí Stop bude umožněno upravit některá nastavení. Opětovné stlačení Start uvede instrument opět do činnosti.
Poznámka: Řád FIR filtru lze nastavit pouze při prvním spuštění aplikace. Pro nové nastavené ukončete aplikaci a spusťte znovu.

Přímá identifikace systému typu IIR

K práci s úlohou postupujte následujícími kroky:

1 Nastavení parametrů Butterworthova IIR filtru - bude zde simulovat neznámý systém pro adaptivní filtr.
a. Zvolte typ filtru. Máte zde na výběr z nabídky: Dolní propust (Lowpass), Horní propust (Highpass), Pásmová propust (Bandpass) a Pásmová zádrž (Bandstop).
Poznámka : U FIR filtrů typu Bandpass nebo Bandstop nesmí být řád filtru

liché číslo! LabView ukáže chybové hlášení a ukončí program. Nastavte tedy proto u těchto typů řád filtru na sudé číslo.
b. Zvolte řád filtru (Butterworth filter order) podle svého uvážení. Doporučené hodnoty jsou nižší než 40. Volte nejlépe v rozmezí 4 - 32.
c. Pomocí táhla upřesněte mezní frekvence. Dovolené hodnoty dovolují rozmezí až 666 Hz. Tyto hodnoty není nutné měnit.

2 Nastavení parametrů adaptivního LMS filtru.
a. Zvolte řád filtru (LMS filter order) podle vašeho přání. Měli byste se orientovat podle řádu IIR filtru.
b. Nastavte velikost kroku (Step size) pro LMS filtr. Doporučené rozmezí je 0,001 až 0,020.

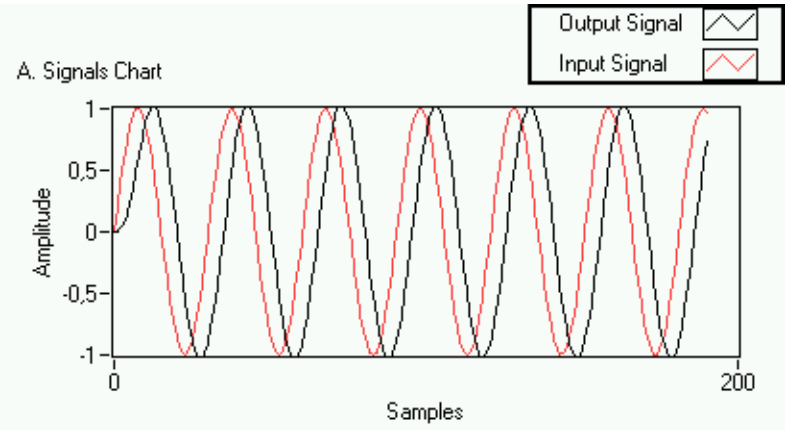
3 Nastavení zdroje signálu.
a. Zdrojem signálu je zde generátor posloupnosti číslicového signálu. Lze zde nastavit jeden z pěti typů signálového vstupu: Sinusoida (Sine), Trojúhelník (Triangle), Pila (Sawtooth), Čtverec (Square) a uniformní bílý šum (Uniform White Noise).
Poznámka: Signál Sine obsahuje pouze 1 kmitočtovou složku, ostaní signály mají složek více. Bílý šum obsahuje rovnoměrně rozprostřené spektrum všech kmitočtových složek.
b. Frekvence neboli kmitočet signálu - základní harmonické složky lze volit od 0 Hz do 8000 Hz, tedy do poloviny vzorkovací frekvence 16 000 Hz.

4 Spuštění aplikace
Zmáčkněte tlačítko Start. Některá tlačítka se stanou nedostupné. Pro ukončení momentální adaptace zmáčkněte tlačítko Stop. Po zmáčknutí Stop bude umožněno upravit některá nastavení. Opětovné stlačení Start uvede instrument opět do činnosti.
Poznámka: Řád IIR filtru lze nastavit pouze při prvním spuštění aplikace. Pro nové nastavené ukončete aplikaci a spusťte znovu.

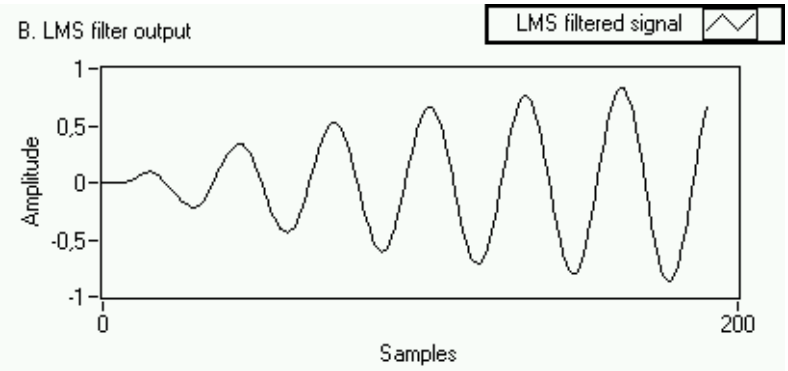
4.3.5 Výsledky měření

Měření lze provádět pří různých nastavení typů filtrů, jejich řádu a různém nastavení vstupního signálu. Standardní výsledky měření si můžeme prohlédnout na následujících grafech. Pro oba typy systémů, FIR i IIR byl použit řád filtru 10 a vstupem

byl sinusový signál o zvolené frekvenci. Běh programu lze kdykoli přerušit tlačítkem STOP a potom je možné exportovat výsledné grafy.



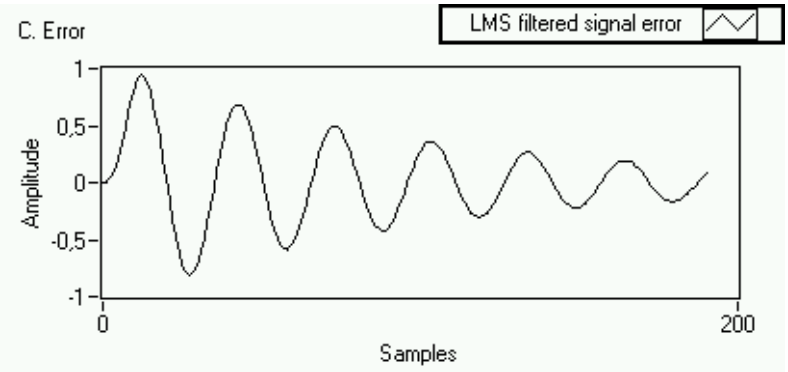
Obr. 4.18: Vstupní signály



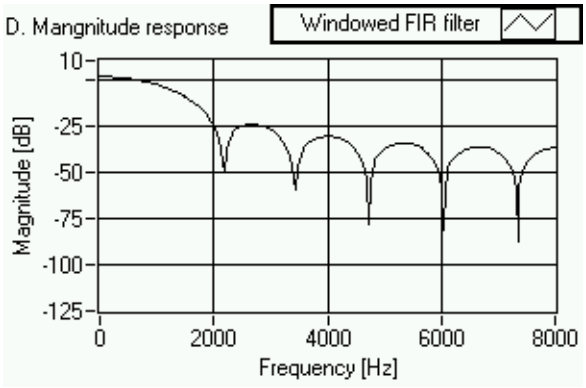
Obr. 4.19: Výstupní signál LMS filtru

4.3.6 Měření závislosti

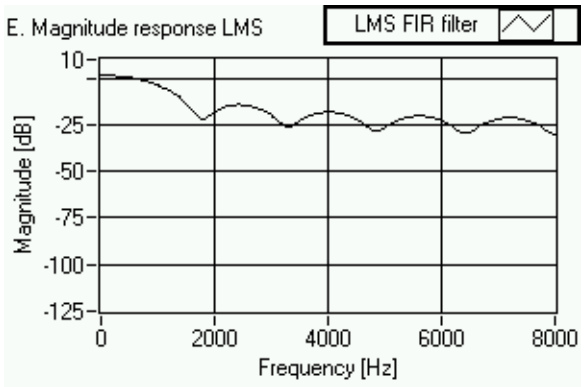
V této úloze jsme se rozhodli dokázat závislost rychlosti adaptace LMS filtru na zvolené velikosti kroku (Step size) adaptivního filtru a na typu vstupního signálu. Dočasně jsme upravili blokový diagram tak, aby iterativní adaptace skončila, když průměrná chyba MSE posledních 200 vzorků bude mít hodnotu 0.0001. Naměřené závislosti jsme vynesli do grafů závislosti pro systém typu FIR a systém typu IIR. Na ose y je počet iterací filtru nutných ke konečné konvergenci koeficientů filtru, kdy výsledná chyba MSE bude pod přijatelnou úrovní.



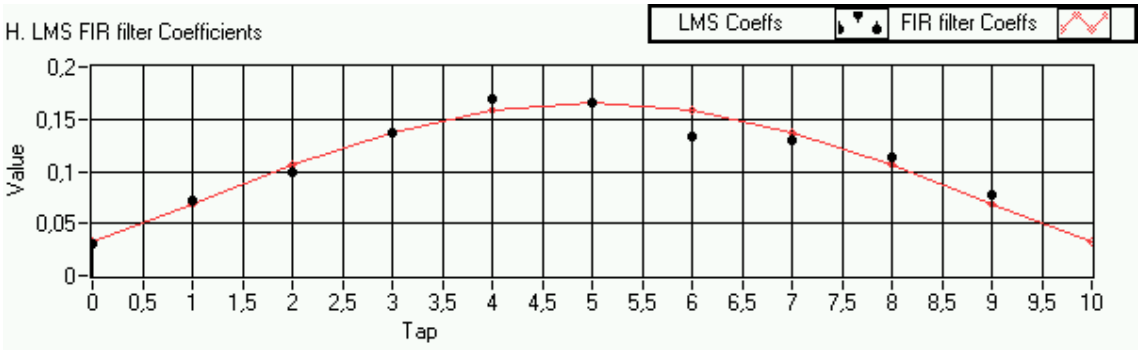
Obr. 4.20: Chybový rozdílový signál



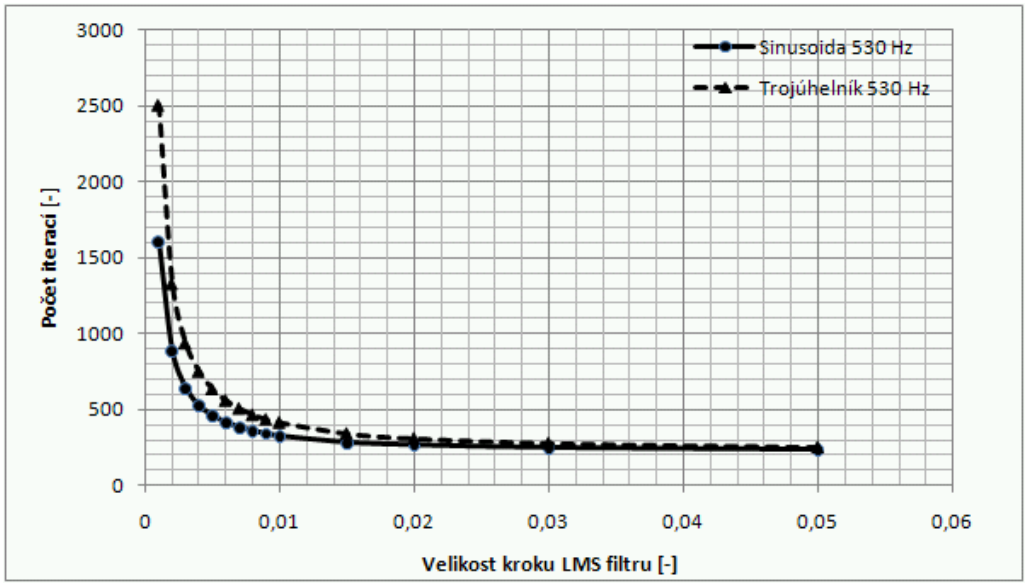
Obr. 4.21: Frekvenční charakteristika FIR systému



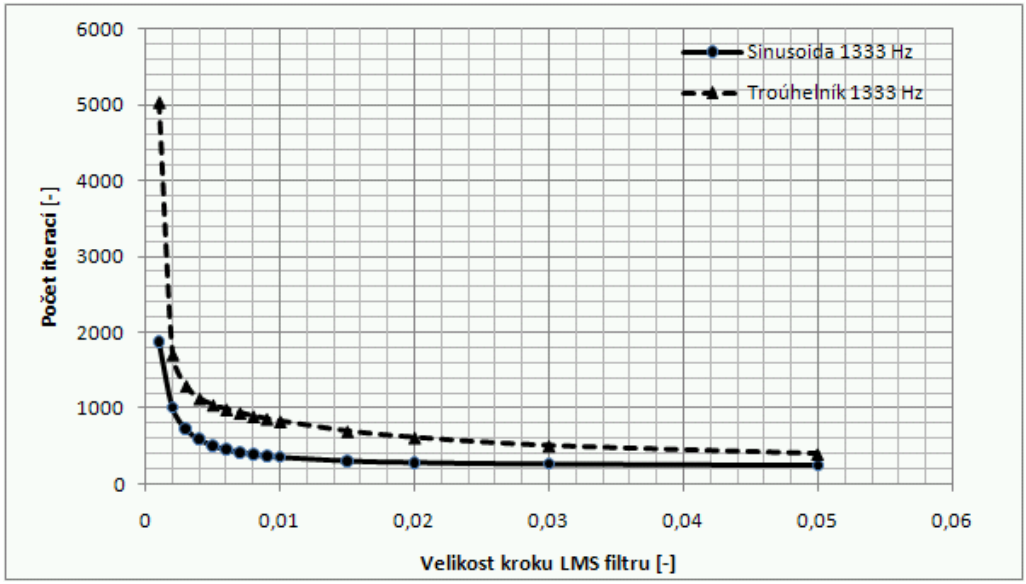
Obr. 4.22: Frekvenční charakteristika LMS filtru



Obr. 4.23: Koeficienty filtrů



Obr. 4.24: Závislost počtu iterací na velikosti kroku, typ systému FIR



Obr. 4.25: Závislost počtu iterací na velikosti kroku, typ systému IIR

4.3.7 Shrnutí

Tato úloha slouží k poznání možnosti využít LMS adaptivní filtr k přímé identifikaci systému. Byla zde představena možnost sestavit adaptivní filtr pomocí elementárních VI v aplikaci LabView. Úloha běží v reálném čase s vzorkovací frekvencí 16000. Jelikož ovšem chceme zachovat přehlednost příkladu, je každá iterace výpočtu zdržena o 20 ms.

Výsledné naměřené grafy ilustrují závislost rychlosti konvergence adaptivního LMS filtru na velikosti kroku při různých vstupních signálech. Závislost je pouze orientační, protože velmi záleží na ostatních faktorech, jako je řád filtru, typu vstupního signálu a jeho kmitočtu.

5 ZÁVĚR

Výsledkem této práce jsou tři laboratorní úlohy demonstrující aplikace adaptivních filtrů ke zpracování číslicových signálů. Z nich právě jedna (Přímá identifikace systému) pracuje v reálném čase a ostatní dvě mají možnost připojit externí zařízení v podobě mikrofonu, který dělá úlohu zajímavou.

První úloha „Adaptivní potlačování echa“ demonstruje použití adaptivního normalizovaného LMS filtru. Úloha používá jako vstup řečový signál nahraný v úloze, nebo lze využít i externí zvukový wave soubor. Protože je složité určit správné nastavení adaptivního NLMS algoritmu, provedli jsme měření závislosti ERLE na velikosti kroku algoritmu při různých délkách filtru. Hodnota ERLE (dB) udává relativní úroveň signálu echa proti původnímu signálu.

Druhá úloha „Aktivní potlačení rušení“ představuje možnost použít adaptivní filtr k potlačení aditivního rušícího signálu principem superpozice. Úloha je realizována algoritmem FXLMS. Vstupním signálem je řeč uživatele a rušení je uskutečněno signálem generovaným přímo v úloze. Typ a amplitudu rušícího signálu lze zvolit. Protože není k dispozici návod, jak zvolit ideální nastavení adaptivního algoritmu, provedli jsme měření závislosti SNRE (dB) na velikosti kroku algoritmu při různé délce filtru. Hodnota SNRE udává rozdíl hodnot SNR před a po adaptivní filtraci.

Třetí úloha „Přímá identifikace systému“ je rozdělena do dvou větví. Jedna část demonstruje adaptaci filtru LMS na systém FIR, druhá provádí adaptaci na IIR systém. Úloha je realizována filtrem typu LMS sestaveným z jednoduchých bloků v LabView. Úloha pracuje v reálném čase, ale pro optimální zobrazení výsledků je každá iterace (vzorek) zpožděna o 20 ms. Vstupní signál lze nastavit z několika voleb a lze měnit jeho kmitočet.

Všechny tři úlohy lze využít k demonstraci práce adaptivních filtrů. Vhodné využití mohou naleznout v laboratorní výuce v původním stavu, nebo s drobnými úpravami.

LITERATURA

[1] CLARK, Cory L., *LabVIEW Digital Signal Processing and Digital Communications*. 2005. McGraw-Hill, ISBN 0-07-146966-4

[2] ISEN, Forester W., *DSP for MATLAB and LabVIEW Volume IV: LMS Adaptive Filtering*. 2009. Morgan & Claypool. ISBN: 9781598299007

[3] JAN, J. , *Číslicová filtrace, analýza a restaurace signálů*. 2002. Vysoké učení technické v Brně, nakladatelství VUTIUM, ISBN 80-214-1558-4.

[4] JONES, Douglas L., *Adaptive Filters*. 2005. Rice University Houston, Texas. Dostupné z URL: <http://cnx.org/content/col10280/1.1/>

[5] KEHTARNAVAZ, N., *Digital Signal Processing System Design: LabVIEW-Based Hybrid Programming*. 2008. University of Texas at Dallas. Elsevier Inc. ISBN: 978-0-12-374490-6

[6] *LabView Help*. LabView 2009. Adaptive Filter Toolkit. National Instruments Corporation.

[7] *Echo Basics Tutorial*. 2011. Ditech networks. Dostupné z URL: <http://www.ditechnetworks.com/learningCenter/echoBasics.html>

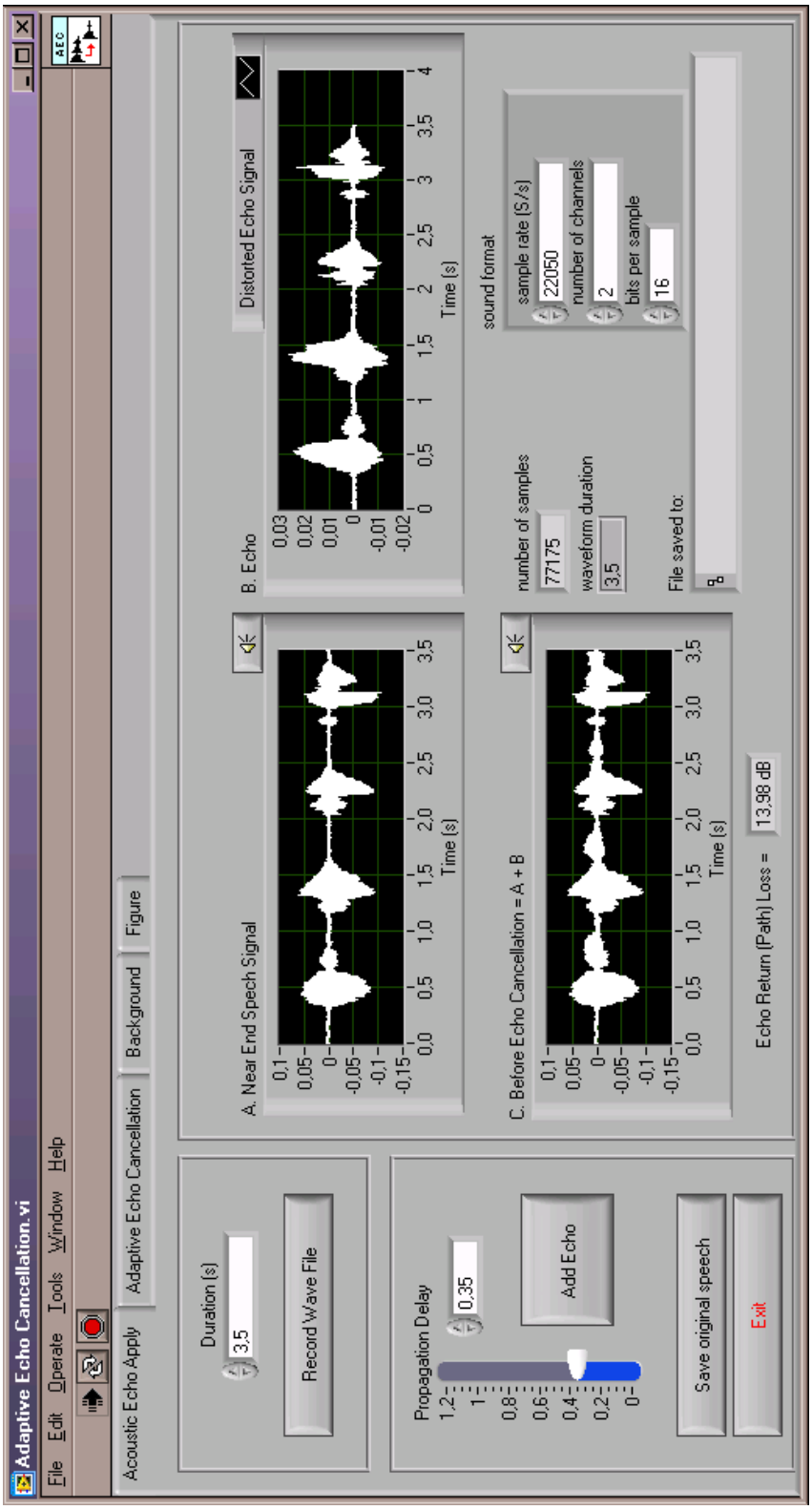
[8] *Wiener Filter*. Wikipedia. Dostupné z URL:<http://en.wikipedia.org/wiki/Wiener_filter>

[9] VLACH, J., HAVLÍČEK, J., VLACH, M., *Začínáme s LabVIEW*. 2008. BEN - technická literatura, ISBN: 978-80-7300-245-9

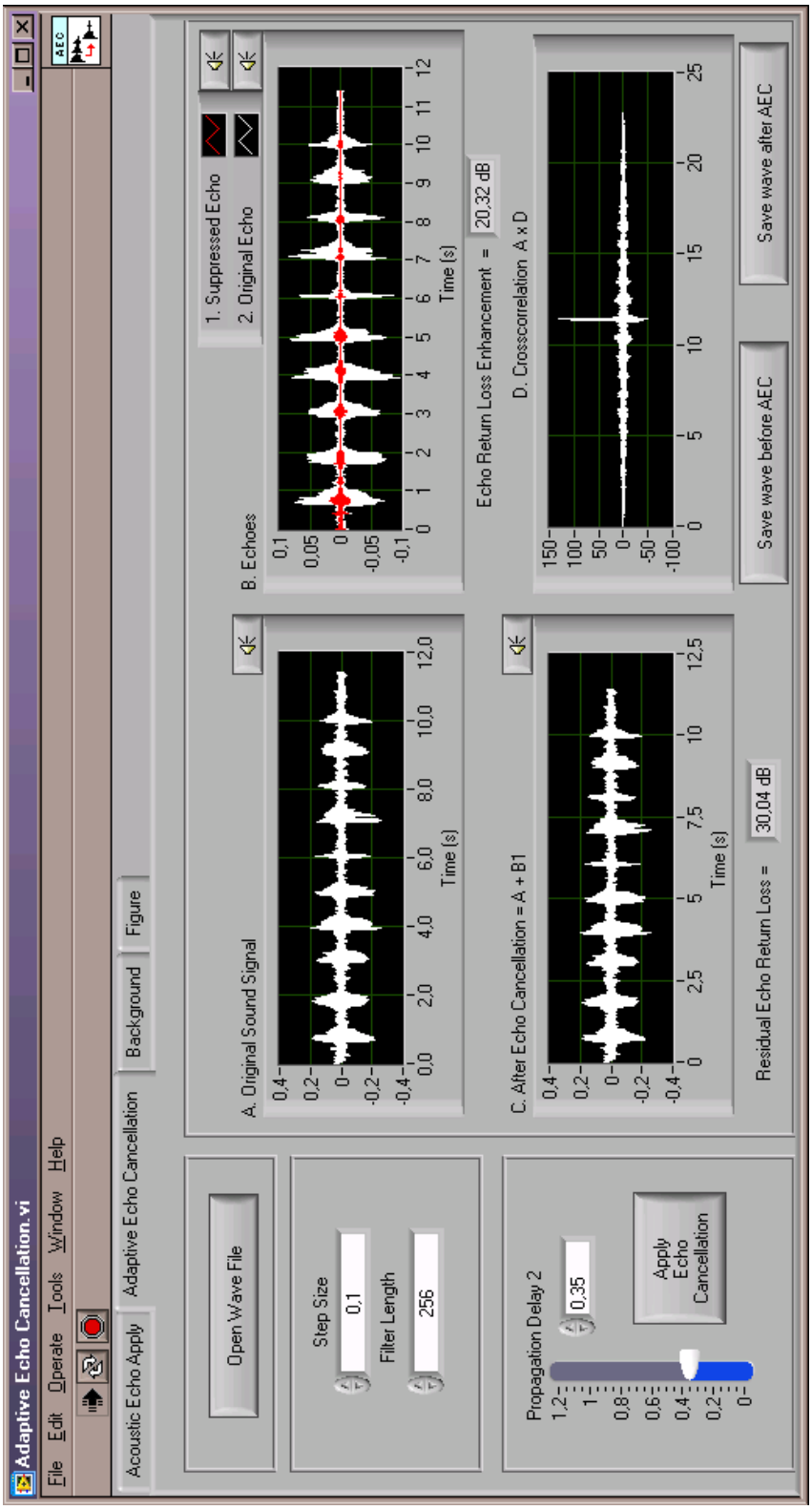
SEZNAM PŘÍLOH

A PŘÍLOHA: ADAPTIVNÍ POTLAČENÍ ECHA

A.1 Uživatelské prostředí úlohy



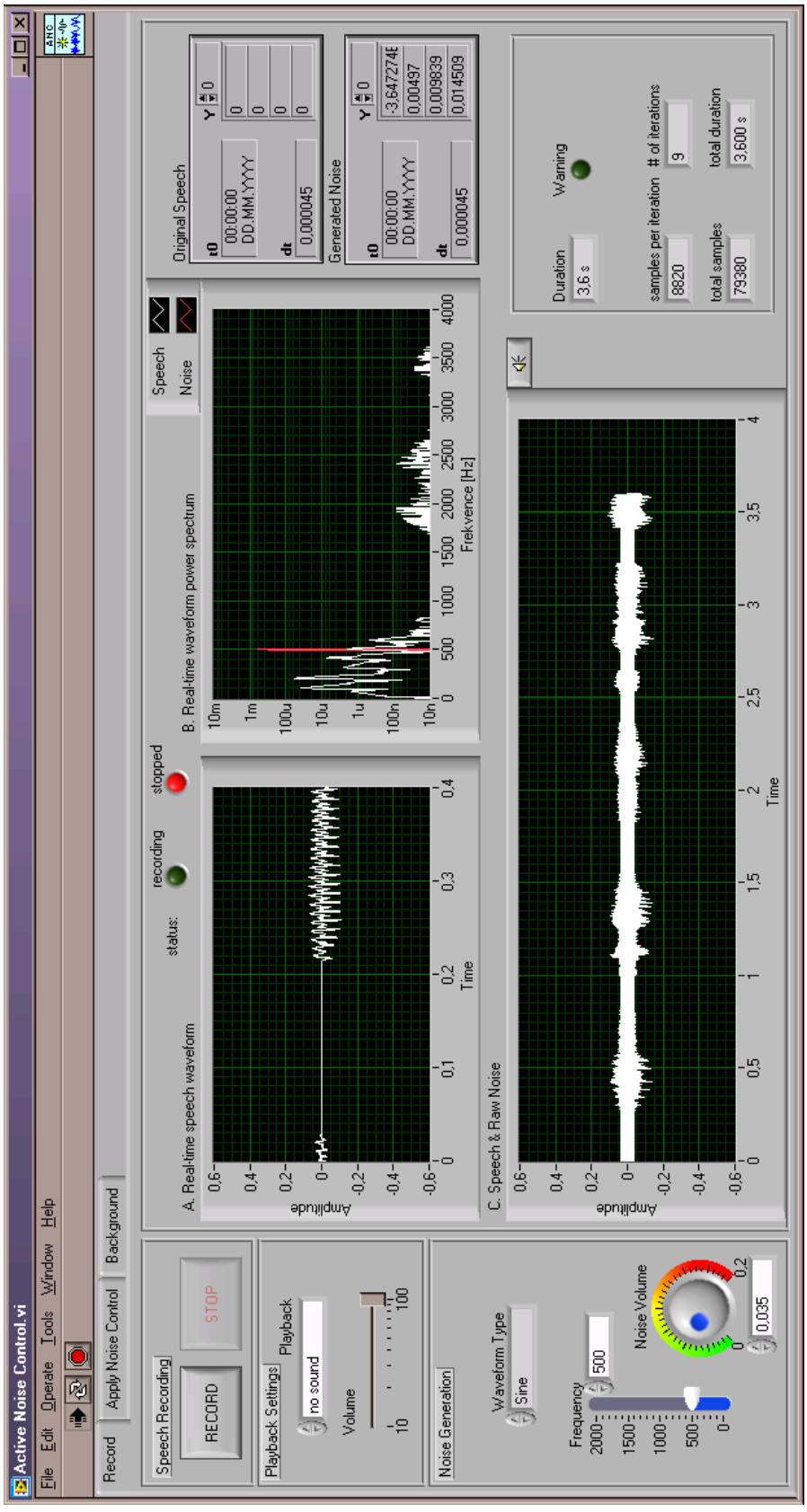
Obr. A.1: Čelní panel Adaptive Echo Apply



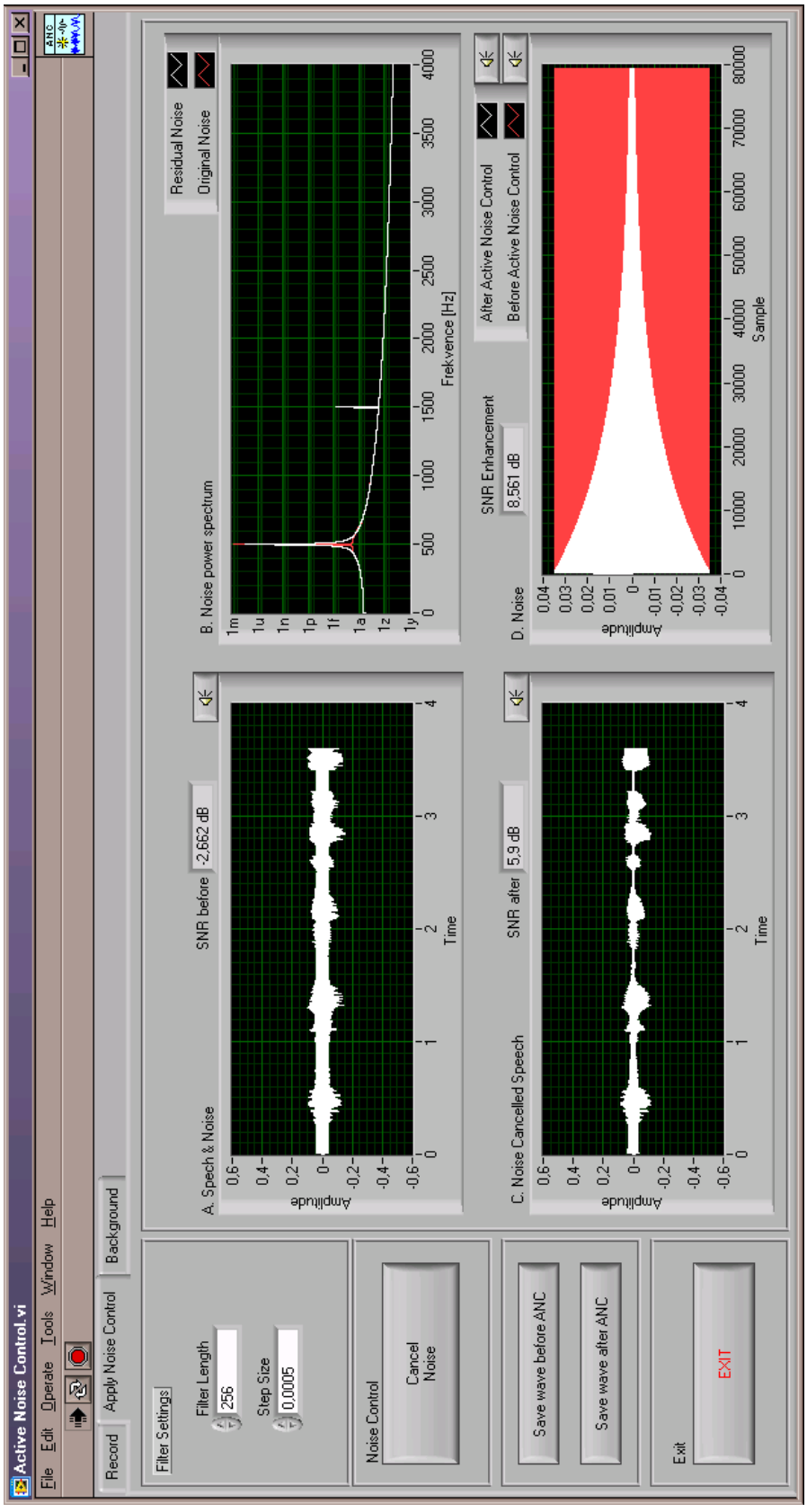
Obr. A.2: Čelní panel Adaptive Echo Cancellation

B PŘÍLOHA: AKTIVNÍ POTLAČENÍ RUŠENÍ

B.1 Uživatelské prostředí úlohy



Obr. B.1: Čelní panel Record

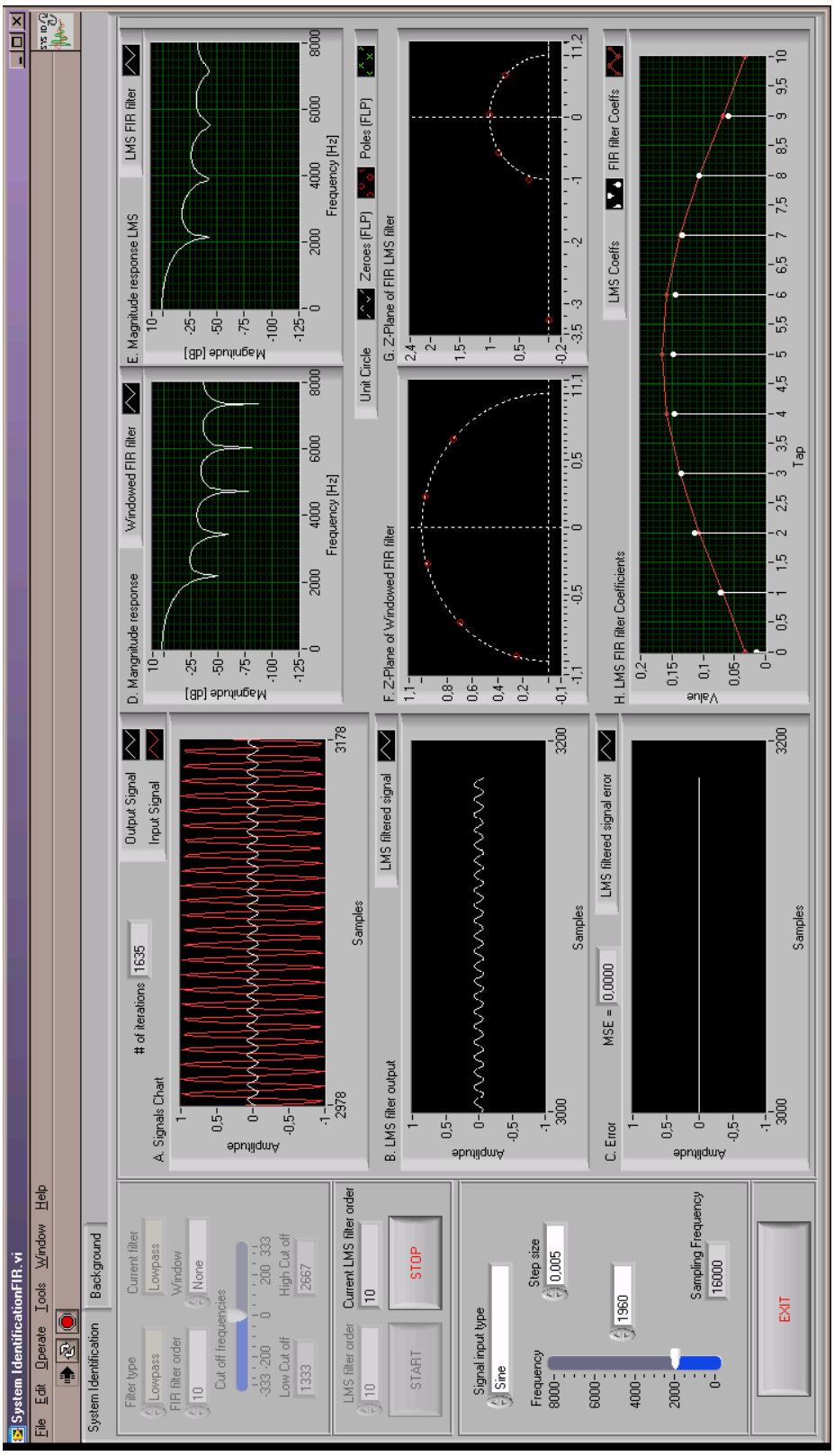


Obr. B.2: Čelní panel Active Noise Control

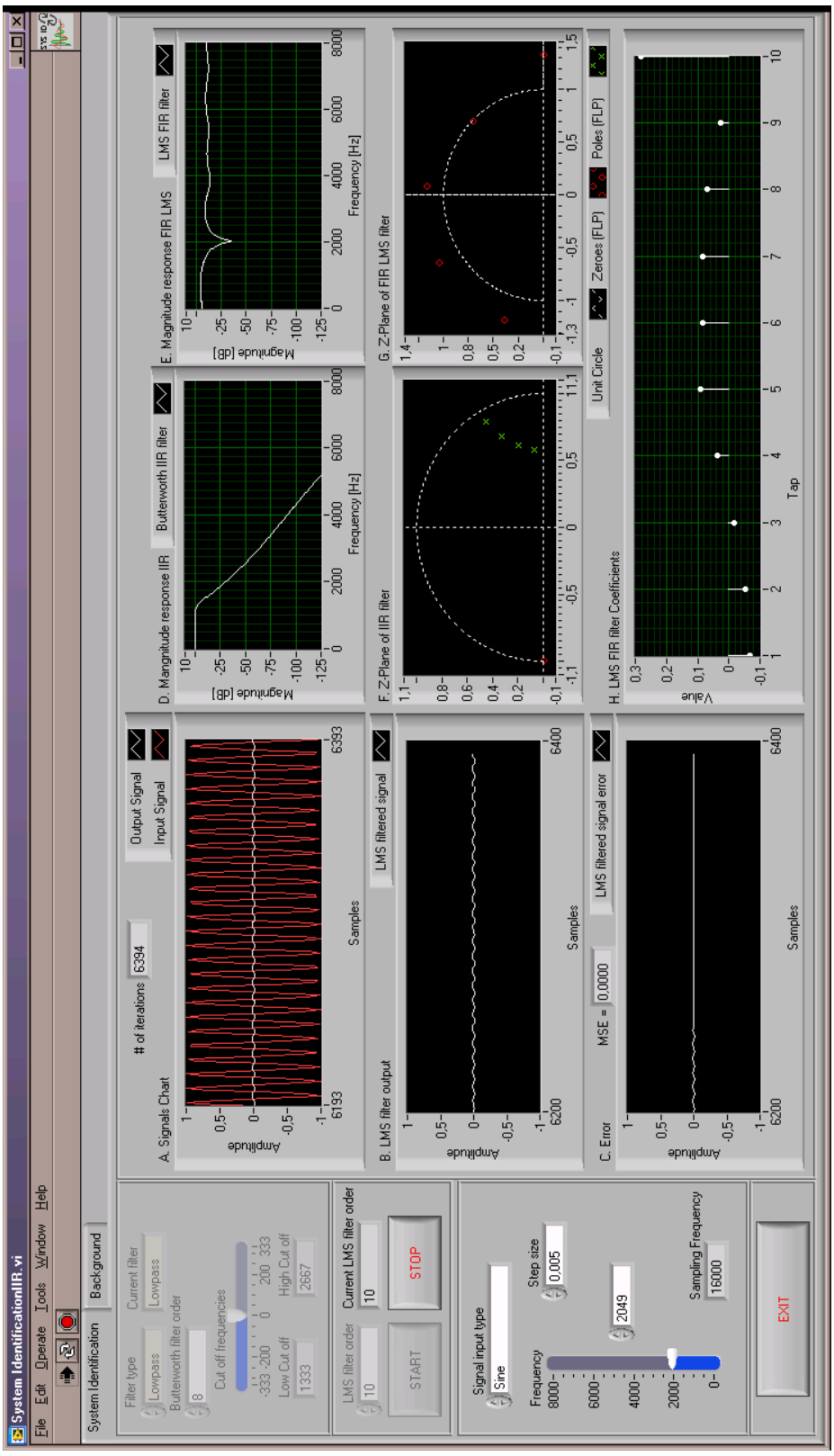
C PŘÍLOHA: IDENTIFIKACE SYSTÉMU

C.1 Uživatelské prostředí se systémem FIR

C.2 Uživatelské prostředí se systémem IIR

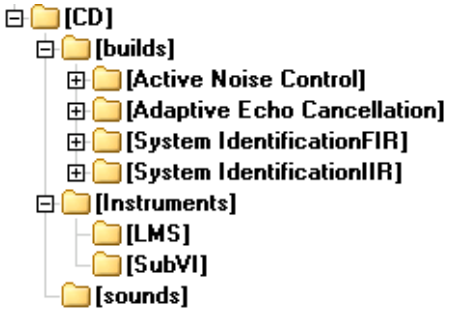


Obr. C.1: Čelní panel identifikace systému FIR



Obr. C.2: Čelní panel identifikace systému IIR

D OBSAH PŘILOŽENÉHO CD



Obr. D.1: Struktura adresářů na přiloženém CD

Na přiloženém CD nalezneme zdrojové kódy a spustitelné aplikace laboratorních úloh.

Sktruktura adresářů je následující:

Adresář „builds“ obsahuje podsložky se spustitelnými úlohami. Pro spuštění úloh je třeba mít nainstalovaný LabView Run-time engine verze 2009 nebo novější.

Adresář „Instruments“ obsahuje zdrojové kódy laboratorních úloh. Úlohy mají příponu souboru .vi. Projekt k úloze má příponu .lvproj. Zdrojové kódy jsou kompatibilní s LabView verzí 2009 a novějšími. Pro korektní otevření úloh je třeba nainstalovat Adaptive Filter Toolkit a Digital Filter Design Toolkit.

Poznámka: při otevření instrumentů může být nutné specifikovat cestu k jiným sub-VI, které se nachází v adresářích „LMS“ a „SubVI“.

Adresář „sounds“ obsahuje nějaké testovací zvukové soubory. Doporučeno je ovšem používat vlastní nahrávky řeči nahrané s pomocí úloh.