VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2020

Bc. MATOUŠ VRBÍK

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

MODELOVÁNÍ LINEÁRNÍHO ZKRESLENÍ ZVUKOVÝCH ZAŘÍZENÍ
MODELING OF LINEAR DISTORTION OF AUDIO DEVICES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Matouš Vrbík**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Jiří Schimmel, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Audio inženýrství**

Ústav telekomunikací

Student: Bc. Matouš Vrbík ***ID:*** 182721
Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Modelování lineárního zkreslení zvukových zařízení

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte a porovnejte číslicové metody modelování modulové kmitočtové charakteristiky zvukových zařízení. Zaměřte se zejména na metody využívající paralelních sekcí IIR filtrů nízkého řádu. Vybranou metodu implementujte v jazyce C, C++ nebo C# jako soubor funkcí. Navrhněte a implementujte také metodu extrapolace modulové kmitočtové charakteristiky mimo změřenou oblast. Následně navrhněte algoritmus automatické optimalizace parametrů metody za účelem minimalizace slyšitelného rozdílu mezi původním zvukovým zařízením a jeho číslicovou simulací a vytvořte aplikaci, která bude provádět simulaci v reálném čase.

DOPORUČENÁ LITERATURA:

[1] BANK, B. "Perceptually Motivated Audio Equalization Using Fixed-Pole Parallel Second-Order Filters". In IEEE Signal Processing Letters, Vol. 15, 2008, pp. 477-480. DOI: 10.1109/LSP.2008.921473, ISSN: 1070-9908

[2] KARJALAINEN, M., PIIRILÄ, E., JÄRVINEN, A., HUOPANIEMI, J. "Comparison of Loudspeaker Equalization Methods Based on DSP Techniques", In AES 102nd Convention 1997, Munich, Germany, Preprint 4437.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 1.6.2020

Vedoucí práce: doc. Ing. Jiří Schimmel, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

V této práci jsou probrány metody úpravy a modelování kmitočtové charakteristiky zvukového zařízení číslicovými filtry. Mimo klasických metod návrhu číslicových filtrů je pozornost věnována pokročilejším numerickým metodám, zejména Pronyho a Steiglitz-McBride.
Práce je zaměřena na strukturu filtru využívající paralelní sekce IIR filtrů druhého řádu. Jsou ukázány a implementovány metody výpočtu koeficientů této paralelní struktury. Pro vybranou metodu s nejlepšími výsledky využívající tzv. dvojité borcení kmitočtové osy je implementován algoritmus automatického výpočtu parametrů potřebných k návrhu filtru pomocí iterativní numerické metody optimalizace hejnem částic.
Je také představeno celkem šest způsobů vyhodnocení přesnosti návrhu a jsou porovnány jejich výsledky.
Funkce realizující návrh filtru jsou implementovány v jazyce C++, MATLAB a Python. Výstupem je také VST modul, který předvede simulaci navrženého filtru v reálném čase.

KLÍČOVÁ SLOVA

C++, číslicové filtry, MATLAB, modelování kmitočtové charakteristiky, optimalizace hejnem částic, paralelní filtry, Steiglitz-McBride, warped filtry

ABSTRACT

Methods used for correction and modeling of frequency response of sound devices are discussed in this paper. Besides classic methods of digital filter design, more advanced and complex numerial methods are reviewed, Prony and Steiglitz-McBride in particular.
This paper focuses on structure utilizing parallel sections of second-order IIR filters. Methods for calculating coefficients of this structure are presented and later implemented. For selected method, utilizing dual frequency warping, an interative algorithm for automatic calculation of parameters necessary to filter design is implemented – so called Particle Swarm Optimization. Six ways of evaluation filter design precision are presented and the results are compared.
Functions realizing filter design are implemented in C++, MATLAB and Python. A VST module simulating the filter in real time is also provided.

KEYWORDS

C++, digital filters, frequency response modeling, MATLAB, Particle Swarm Optimization, parallel filters, Steiglitz-McBride, warped filters

VRBÍK, Matouš. *Modelování lineárního zkreslení zvukových zařízení*. Brno, 2020, 102 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: doc. Ing. Jiří Schimmel, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Modelování lineárního zkreslení zvukových zařízení“ 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/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu doc. Ing. Jiřímu Schimmelovi, Ph.D. za odborné vedení, konzultace, ochotu, trpělivost a podnětné návrhy k práci.
Poděkování patří i spolužákům, díky kterým mezi námi neutichala zdravá soutěživost a chuť se stále zdokonalovat.
V neposlední řadě děkuji svým rodičům a přítelkyni, kteří za mnou vždy stáli a podporovali mě.

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Chceme-li si dopřát kvalitní poslech z pozice posluchače, nebo naopak potřebujeme precizní reprodukci zvuku např. v nahrávacím studiu, potřebujeme ideálně takové zvukové zařízení, které reprodukuje všechny složky spektra se stejným zesílením. Mnoho zařízení však těmito vlastnostmi nedisponuje a v některých případech je jejich nevyrovnaná kmitočtová charakteristika tak zajímavá, že uživatel preferuje použití kvůli jejich charakteru.

Při úpravě tohoto lineárního zkreslení uplatňujeme zpravidla dva přístupy: požadujeme upravit nedokonalou kmitočtovou charakteristiku zařízení takovým způsobem, aby byla co nejvyrovnanější, nebo chceme modelovat nevyrovnanou charakteristiku nějakého zařízení, které je pro nás zajímavé (např. kytarový reprobox). Rozšířením druhého přístupu je možné docílit simulace nejen jiného zvukového zařízení, ale i modelování charakteru rezonátoru hudebního nástroje.

Po změření impulsové odezvy zařízení se pro výpočet výstupního signálu nabízí klasická konvoluce, avšak při zpracování v reálném čase při dlouhých impulsových odezvách narážíme na nedostatečnou rychlost zpracování. Přístup k úpravě kmitočtové charakteristiky popsaný v této práci staví na paralelním zpracování vstupního signálu IIR filtry druhého řádu, což zajistí několikanásobné snížení procesního zpoždění při výpočtu výstupního signálu.

V prvních dvou kapitolách jsou představeny diskrétní systémy a jejich realizace. Jsou popsány návrhy číslicových filtrů klasickými metodami a je zmíněna metoda využívající borcení kmitočtové osy, čímž lze dosáhnout návrhu, který je více v souladu se způsobem vnímání zvuku člověkem.

Ve třetí a čtvrté kapitole jsou popsány pokročilejší metody návrhů číslicových filtrů, zejména pak filtr s paralelní strukturou IIR sekcí druhého řádu. Obsažené poznatky potom naleznou uplatnění při implementaci funkcí realizujících výpočet koeficientů filtru.

Pátá kapitola definuje parametry, které jsou potřeba k návrhu paralelního filtru a jsou představeny různé přístupy k návrhu. Dále je popsaný proces automatické optimalizace vstupních parametrů k návrhu filtru pomocí algoritmu *optimalizace hejnem částic* [20]. Pro využití tohoto algoritmu je nutné objektivně vyhodnotit přesnost návrhu filtru, a tak je představeno několik možností, jak toho docílit.

Šestá kapitola uplatňuje poznatky z předchozích kapitol a pojednává o implementaci algoritmu v programovacích jazycích a obecných či specifických problémech při implementaci. Je vysvětlen princip algoritmu a jsou porovnány výsledky vypracované v prostředích MATLAB, C++ a Python. Je také diskutována možnost zapojit do návrhu váhovací vektor a způsoby jeho vytvoření.

Poslední kapitola popisuje výsledky této práce, vytvořené funkce a aplikace. Součástí této práce je také zásuvný VST modul pro simulaci paralelního filtru v reálném čase.

1 DISKRÉTNÍ SIGNÁLY A SYSTÉMY

Signál je obecně sledování nějakého jevu, který se mění v čase, v prostoru nebo dle jiné veličiny. Matematicky je signál popsán jako funkce jedné nebo více nezávislých proměnných. Nejčastější nezávislou proměnnou v oblasti zpracování zvukových signálů je čas. Zvukový signál je tedy nejčastěji popsán změnou fyzikální veličiny (např. akustického tlaku, napětí na výstupu elektroakustického měniče či kvantované hodnoty tohoto napětí) v čase.

Signály a systémy používané při zpracování zvuku se rozdělují dle definičních oborů nezávislé a závislé proměnné na:

- **spojité** – nezávislá je z množiny reálných čísel a závislá je z množiny reálných nebo komplexních čísel,
- **diskrétní** – nezávislá je z množiny celých čísel a závislá je z množiny reálných nebo komplexních čísel,
- **číslicové** – nezávislá je z množiny celých čísel a závislá je z množiny použité číselné soustavy nebo číselné reprezentace, která je *konečná*.

Diskrétní signál $x(n)$ bude po vzorkování původního spojitého signálu $x(t)$ popsán jako

$$x(n) = x(t)\Big|_{t=nT_{\mathrm{vz}}}, \tag{1.1}$$

kdy diskrétní signál $x(n)$ je chápán jako posloupnost číselných hodnot vyjadřující velikost signálu $x(t)$ v okamžicích odpovídajících n-násobku vzorkovací periody T_{vz}.

Klasifikace diskrétních signálů dle jejich vlastností není předmětem této práce, potřebné informace k tomuto tématu lze nalézt např. v [1, kap. 3.1].

1.1 Popis diskrétních systémů

V mnoha aplikacích hledáme obvod nebo algoritmus, který provede předepsané operace s diskrétními nebo číslicovými signály. Vybraný algoritmus může být naprogramovaný v počítači nebo implementován např. v signálovém procesoru anebo může jít přímo o hardwarové zařízení nebo obvod. Systém zpracovávající diskrétní signály se nazývá diskrétní systém, číslicový systém potom pracuje s číslicovými signály. V telekomunikační technice se většinou vyskytuje systém, který má jeden vstup a jeden výstup; je také nejsnadnější jej popsat [2, kap. 2].

Kategorizovat diskrétní systémy lze několika způsoby: dle platnosti principu superpozice (lineární, nelineární), dle počtu vstupů a výstupů, dle chování v čase (časově proměnné – variantní, časově neproměnné – invariantní), dle setrvačnosti systému (s pamětí, bez paměti), dle chování systému v závislosti na spektru vstupního signálu (kmitočtově závislé, nezávislé) nebo dle možnosti jeho řízení [1].

Nejjednodušší pro popis je systém *lineární časově invariantní* (LTI – *Linear Time Invariant*), tedy splňující podmínku principu superpozice a neměnný v čase. Předmětem

této práce bude modelování LTI systému, jelikož u modelovaného zařízení budeme předpokládat, že se pohybuje ve své lineární oblasti a jeho parametry se skutečně s časem nebudou měnit.

Na diskrétní systém je možné pohlížet dvěma odlišnými způsoby. Můžeme se na něj dívat jako na neznámý objekt (*black box*), o jehož vnitřní struktuře nic nevíme, tedy známe jen signály, které do systému vstupují a z něj vystupují. Tento popis nazýváme *vnější popis*.

Druhý pohled předpokládá, že strukturu systému známe (*white box*), popř. ji dovedeme modelovat, a sledujeme vývoj *stavových* (vnitřních) veličin systému. Tento popis diskrétního systému se nazývá *vnitřní popis*. Vyjadřuje vztah mezi vstupními, stavovými a výstupními veličinami.

Stavové veličiny zpravidla reprezentují energii obsaženou v systému na různých místech. U spojitě pracujících elektrických systémů jde o elektrickou energii nahromaděnou v setrvačných prvcích, jimiž jsou kapacitory a induktory (resp. kondenzátory a cívky). U diskrétních systémů je tento energetický stav systému reprezentován pamětí – paměťové registry [3]

1.1.1 Vnější popis systému

Základním matematickým popisem vlastností diskrétních systémů jsou *diferenční rovnice*. Pokud budeme uvažovat LTI systém, lze jeho vlastnosti definovat pomocí lineární diferenční rovnice s-tého řádu s konstantními koeficienty prvního typu [3]

$$\begin{aligned} A_s\Delta^s y(n) + A_{s-1}\Delta^{s-1} y(n) + \cdots + A_1\Delta y(n) + A_0 y(n) = \\ = B_s\Delta^s x(n) + B_{s-1}\Delta^{s-1} x(n) + \cdots + B_1\Delta x(n) + B_0 x(n). \end{aligned} \tag{1.2}$$

Pomocí diferenčního počtu lze upravit rovnici prvního typu na rovnici druhého typu a vyjádřit ji pomocí posloupností

$$\begin{aligned} a_s y(n+s) + a_{s-1} y(n+s-1) + \cdots + a_1 y(n+1) + a_0 y(n) = \\ b_s x(n+s) + b_{s-1} x(n+s-1) + \cdots + b_1 x(n+1) + b_0 x(n), \end{aligned} \tag{1.3}$$

kde $a_0, a_1, \ldots, a_s$ a $b_0, b_1, \ldots, b_s$ jsou koeficienty diferenční rovnice, $x(n)$ je známá posloupnost a $y(n)$ je hledané řešení diferenční rovnice.

Za předpokladu, že systém popsaný diferenční rovnicí byl pro $n < 0$ v klidu, tedy $x(n) = y(n) = 0$ pro $n < 0$, provedeme $\mathcal{Z}$-transformaci rovnice (1.3) a obdržíme tak obraz partikulárního řešení diferenční rovnice

$$Y(z) = \frac{\sum_{i=0}^{s} b_i z^{-i}}{\sum_{j=0}^{s} a_j z^{-j}} X(z). \tag{1.4}$$

Podíl $Y(z)$ a $X(z)$ se označuje $H(z)$ a nazývá se *systémová přenosová funkce* (nebo jen přenosová funkce) a její normalizovaný tvar ($a_0 = 1$) je definovaný

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_M z^{-M}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_N z^{-N}}. \tag{1.5}$$

Zde je řád čitatele a jmenovatele obecně M, resp. N, které nemusí být stejné (ovšem pro podmínku kauzality musí platit $M \leq N$).

Označíme-li polynom čitatele přenosové funkce $P(z)$ a polynom jmenovatele přenosové funkce $Q(z)$, je možné nalézt kořeny charakteristických rovnic $P(z) = 0, Q(z) = 0$. Potom lze přenosovou funkci vyjádřit jako

$$H(z) = \frac{P(z)}{Q(z)}, = \frac{\sum_{i=0}^{M} b_i z^{-i}}{\sum_{j=0}^{N} a_j z^{-j}} = k \frac{\prod_{i=1}^{M} (z - n_i)}{\prod_{j=1}^{N} (z - p_j)}, \tag{1.6}$$

kde n_i jsou nulové body a p_j póly systému a k je normovací konstanta.

1.1.2 Vnitřní popis systému

Vnitřní (stavový) popis systému je zcela univerzální a důležitý pro jeho analýzu, avšak nehodí se pro realizaci s minimálními výpočetními nároky. Pro tento popis slouží dvě rovnice – stavová rovnice dynamiky a stavová rovnice výstupu systému. Maticový zápis rovnic má tvar [1]

$$\begin{aligned} \mathbf{v}(n+1) &= \mathbf{A}\mathbf{v}(n) + \mathbf{b}x(n) \\ y(n) &= \mathbf{c}\mathbf{v}(n) + dx(n), \end{aligned} \tag{1.7}$$

kde $\mathbf{A}$ je matice systému popisující vztah mezi stavy systému a jejich změnami, $\mathbf{b}$ je sloupcový vektor buzení popisující vztah mezi vstupy a změnami stavů systému, $\mathbf{c}$ je řádkový výstupní vektor, který popisuje vztah mezi stavy systému a jeho výstupem a d je skalár převodu vstupu na výstup, $\mathbf{v}(n)$, resp. $\mathbf{v}(n+1)$ je vektor aktuálního, resp. budoucího stavu stavových proměnných, $x(n)$ a $y(n)$ jsou potom vstupní a výstupní posloupnost.

Elementární zápis systému druhého řádu má potom tvar

$$\begin{aligned} \begin{bmatrix} v_1(n+1) \\ v_2(n+1) \end{bmatrix} &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} x(n) \\ y(n) &= \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \end{bmatrix} + dx(n) \end{aligned} \tag{1.8}$$

Pomocí $\mathcal{Z}$-transformace lze obě stavové rovnice z (1.7) přepsat do tvaru

$$\begin{aligned} z\mathbf{V}(z) - z\mathbf{v}(0) &= \mathbf{A}\mathbf{V}(z) + \mathbf{b}X(z) \\ Y(z) &= \mathbf{c}\mathbf{V}(z) + dX(z). \end{aligned} \tag{1.9}$$

Avšak i při zanedbání odezvy na počáteční podmínky není lehké vyjádřit přenosovou funkci $H(z)$ ve tvaru racionální lomené funkce tak, jak se nejlépe hodí k popisu v rovině $\mathcal{Z}$ pomocí nulových bodů a pólů [1, kap. 5.2.1].

1.1.3 Impulsová odezva diskrétního filtru

Vztah pro obraz výstupního signálu můžeme pomocí přenosové funkce zapsat

$$Y(z) = H(z)X(z). \tag{1.10}$$

Při dosazení $X(z) = 1$, což odpovídá obrazu jednotkového impulsu

$$x(n) = \delta(n) = \begin{cases} 1 & \text{pro } n = 0 \\ 0 & \text{pro } n \neq 0, \end{cases} \tag{1.11}$$

obdržíme rovnost $Y(z) = H(z)$ a po zpětné $\mathcal{Z}$-transformaci $y(n) = h(n)$, kde $h(n)$ je impulsová odezva systému. Za předpokladu, že byl diskrétní systém pro $n < 0$ v klidu, odezva je definovaná operací diskrétní konvoluce [3]

$$y(n) = x(n) * h(n) = \sum_{m=0}^{\infty} h(m)x(n-m) = \sum_{m=0}^{\infty} h(n-m)x(m). \tag{1.12}$$

Kmitočtová charakteristika

Kmitočtové vlastnosti systému popisuje jeho kmitočtová charakteristika $H(\mathrm{e}^{\mathrm{j}\omega})$, kterou lze získat Fourierovou transformací impulsové odezvy $h(n)$ a nebo ji lze také získat z přenosové funkce systému vhodným dosazením za z [3]

$$H(\mathrm{e}^{\mathrm{j}\omega}) = H(z)\Big|_{z=\mathrm{e}^{\mathrm{j}\omega}} = M(\omega)\mathrm{e}^{\mathrm{j}\varphi(\omega)}, \tag{1.13}$$

kde $M(\omega) = |H(\mathrm{e}^{\mathrm{j}\omega})|$ je *modulová kmitočtová charakteristika* a $\varphi(\omega) = \arg H(\mathrm{e}^{\mathrm{j}\omega})$ je *fázová kmitočtová charakteristika* diskrétního LTI systému. Pro některá použití je vhodnější používat místo fázové charakteristiky $\varphi(\omega)$ její derivaci podle ω, kterou nazýváme *skupinovým zpožděním*

$$\tau(\omega) = -\frac{\mathrm{d}\varphi(\omega)}{\mathrm{d}\omega}. \tag{1.14}$$

Protože exponenciální funkce $\mathrm{e}^{\mathrm{j}\omega}$ je periodická s periodou 2π, platí také, že kmitočtová charakteristika je periodická s periodou 2π.

$$H(\mathrm{e}^{\mathrm{j}\omega}) = H(\mathrm{e}^{\mathrm{j}(\omega+2k\pi)}), \qquad k = \pm 1, \pm 2, \ldots \tag{1.15}$$

1.2 Realizace diskrétních systémů

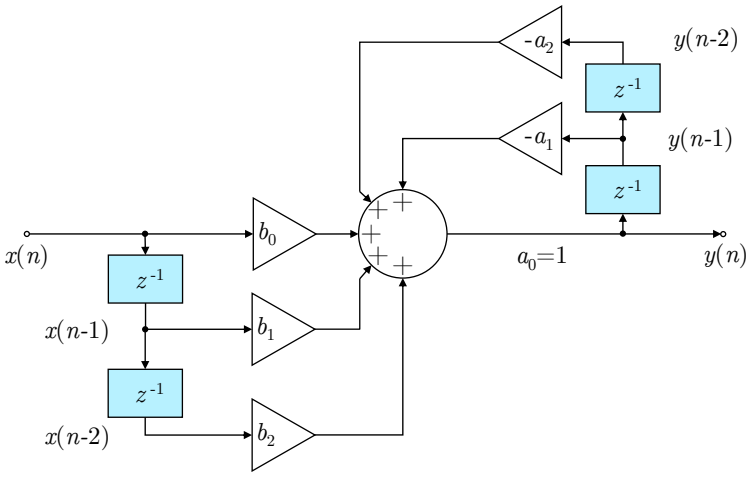
Při realizaci diskrétních systémů je možné použít různé formy a struktury, jejichž volba předurčuje typ realizace. Základní formy se od sebe liší výpočetní náročností, požadavky na paměť, řiditelností, odezvou na počáteční podmínky, přechodovou odezvou atd. Mezi základní formy a struktury patří struktura vycházející z vnitřního popisu, poté přímá forma vycházející z vnějšího popisu, kanonické formy i jako sériová a paralelní realizace nebo zpětnovazební struktura [1].

1.2.1 Přímá forma realizace diskrétního systému

Přímá forma vychází z rovnice (1.3). Úpravou pro vyjádření $y(n)$ dostaneme

$$y(n) = \sum_{i=0}^{M} b_i x(n-i) - \sum_{j=1}^{N} a_j y(n-j). \tag{1.16}$$

K vizualizaci principu diskrétního systému se používají např. bloková schémata nebo grafy signálových toků. Rovnice (1.16) převedená na blokové schéma dá strukturu uvedenou na obr. 1.1.



Obr. 1.1: Blokové schéma přímé formy systému 2. řádu [1, obr 5.9]

Realizace vyžaduje jak zpožďování vstupní, tak výstupní posloupnosti. Velikost zpoždění pro obě posloupnosti je rovna řádu systému. Ve srovnání se strukturou vycházející z vnitřního popisu je však potřeba dvojnásobné množství zpožďovacích bloků, tedy nároky na paměť jsou dvakrát větší. Počet aritmetických operací je však menší. V přímé formě není možné sledovat stav systému, nejsou definovány žádné stavové proměnné.

1.2.2 Kanonické formy

Převod vnějšího popisu systému na vnitřní popis není jednoznačný. Jeden vnější popis může být převeden na více odpovídajících vnitřních popisů. Pokud je normalizovaná přenosová funkce N-tého řádu (stejný řád čitatele i jmenovatele) kauzálního LTI systému definovaná jako

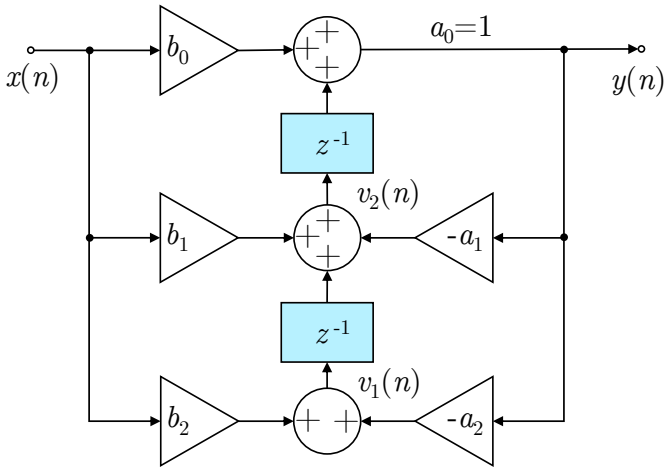
$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_N z^{-N}}, \tag{1.17}$$

je možné ji pomocí Frobeinova kanonického rozkladu převést na tvar, který se označuje **1. kanonická forma**. Pro případ $N = 2$, tedy systém 2. řádu, mají stavové rovnice

1. kanonické formy tvar [1, kap. 5.2.3]

$$\begin{bmatrix} v_1(n+1) \\ v_2(n+1) \end{bmatrix} = \begin{bmatrix} 0 & -a_2 \\ 1 & -a_1 \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \end{bmatrix} + \begin{bmatrix} b_2 - a_2 b_0 \\ b_1 - a_1 b_0 \end{bmatrix} x(n)$$
$$y(n) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} v_1(n) \\ v_2(n) \end{bmatrix} + b_0 x(n) \tag{1.18}$$

Blokové schéma popisující 1. kanonickou formu je na obr. 1.2.



Obr. 1.2: Blokové schéma 1. kanonické formy systému 2. řádu [1, obr. 5.10].

Na schématu vidíme mimo vstupní a výstupní posloupnosti i stavové proměnné a jejich vyjádření odpovídá přepisu rovnice (1.18)

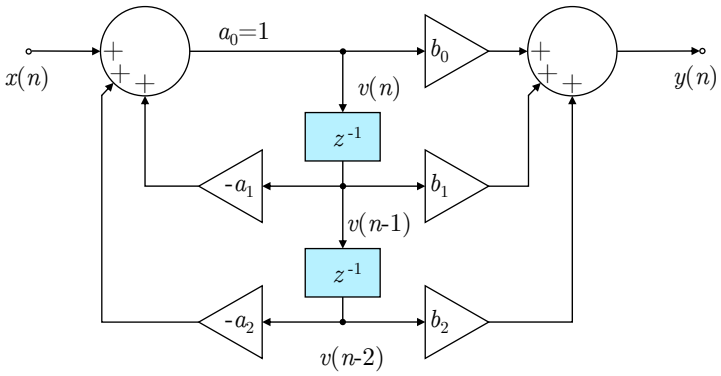
$$\begin{aligned} y(n) &= b_0 x(n) + v_2(n-1) \\ v_2(n) &= b_1 x(n) - a_1 y(n) + v_1(n-1) \\ v_1(n) &= b_2 x(n) - a_2 y(n). \end{aligned} \tag{1.19}$$

Prostřednictvím transponování grafů je možné získat pro stejnou přenosovou funkci další struktury diskrétního systému. Při transponování grafu se zamění uzly vstupní a výstupní veličiny a otočí se směry všech větví. Pomocí této operace přejdeme ze struktury 1. kanonické formy na **2. kanonickou formu**, jejíž schéma vidíme na obr. 1.3.

V grafu se nyní vyskytuje pouze jedna stavová proměnná $v(n)$ a její zpožděné varianty. Rovnice popisující tuto strukturu mají tvar

$$\begin{aligned} v(n) &= x(n) - a_1 v(n-1) - a_2 v(n-2) \\ y(n) &= b_0 v(n) + b_1 v(n-1) + b_2 v(n-2). \end{aligned} \tag{1.20}$$

Tato struktura má oproti 1. kanonické formě o jednu aritmetickou operaci méně a má tak ze všech realizací nejmenší požadavky na výpočetní výkon a zároveň na kapacitu paměti.



Obr. 1.3: Blokové schéma 2. kanonické formy systému 2. řádu [1, obr. 5.11].

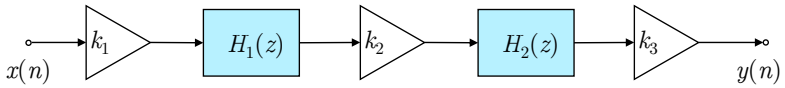
1.2.3 Třetí a čtvrtá kanonická forma

Doposud jsme hovořili obecně o diskrétních systémech, avšak při samotné realizaci systému se bude jednat vždy o systém číslicový. Jak již bylo naznačeno na počátku kapitoly, číslicové systémy pracují se závislostmi, jejichž obor hodnot je konečný, tedy omezený například bitovou architekturou systému nebo volbou datového typu, se kterým systém pracuje.

Kvantování koeficientů přenosové funkce v rovnici (1.17) má za následek konečný počet možných bodů v rovině $\mathcal{Z}$, ve kterých se mohou nulové body a póly přenosové funkce nacházet. Citlivost posunu nulového bodu nebo pólu při kvantování koeficientů je menší, když je číslicový filtr realizován sériovým nebo paralelním spojením dílčích sekcí nižšího řádu [3]. Pokud koeficienty přenosové funkce $H(z)$ jsou reálná čísla, pak se nulové body a póly mohou vyskytovat pouze na reálné ose roviny $\mathcal{Z}$ nebo v komplexně sdružených párech. Z toho plyne skutečnost, že dílčí sekce mohou být prvního nebo druhého řádu.

3. kanonická forma

Sériovým spojením lineárních systémů vznikne 3. kanonická forma realizace systému. Blokové schéma je na obrázku níže.



Obr. 1.4: Blokové schéma 3. kanonické formy systému 2. řádu [1, obr. 5.12].

Schéma obsahuje normovací koeficienty k_i, které jsou voleny tak, aby se minimalizovala navyšující se chyba číselné reprezentace signálů způsobená obecně použitím sériové realizace. Problémy omezeného rozsahu číslicových systémů jsou podrobněji rozebrány např. v [1, kap. 4.6].

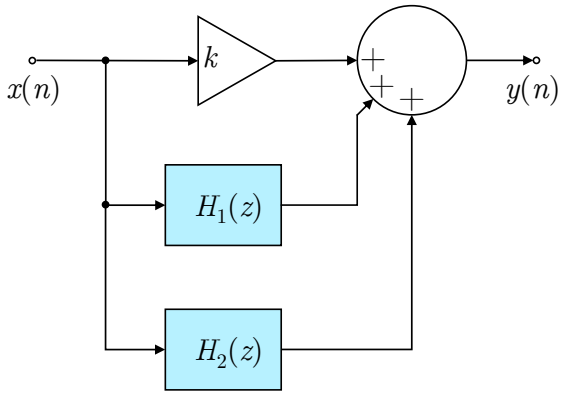
Musí být dodržena podmínka $k = k_1 k_2 k_3 = 1$, aby platila rovnice pro sériovou kombinaci přenosových funkcí

$$H(z) = H_1(z) H_2(z). \tag{1.21}$$

4. kanonická forma

Obdobně paralelní kombinací jednotlivých systémů vzniká struktura nazývající se *4. kanonická forma*. Pokud je konstanta k rovna 0, jde čistě o paralelní spojení jednotlivých přenosových funkcí

$$H(z) = H_1(z) + H_2(z). \tag{1.22}$$



Obr. 1.5: Blokové schéma 4. kanonické formy systému 2. řádu [1, obr. 5.13].

U obou výše zmíněných realizací musí být jednotlivé sekce prvního nebo druhého řádu a jejich realizace jsou zpravidla 1. nebo 2. kanonická forma. Existují i jiné realizace diskrétních systémů, každá se svými klady i zápory. Pro úplnost jsou popsány v [1, 3].

2 ČÍSLICOVÉ FILTRY

Číslicové filtry představují jednu z možných realizací diskrétních systémů číslicovými technickými prostředky. Úkolem číslicového filtru je požadovaným způsobem ovlivnit kmitočtové spektrum vstupního signálu. To znamená buď vybrat část kmitočtového spektra, které bude nezměněno a ostatní části potlačit (kmitočtové filtry typu dolní propusti – DP, horní propusti – HP, pásmové propusti – PP nebo pásmové zádrže – PZ), anebo tvarovat kmitočtovou charakteristiku (např. napodobit kmitočtové vlastnosti lidského ucha), popř. linearizovat fázovou kmitočtovou charakteristiku apod.

Kmitočtové filtry se kategorizují především dle délky jejich impulsové odezvy na filtry typu **FIR** (Finite Impulse Response) – s konečnou impulsovou odezvou a na filtry typu **IIR** (Infinite Impulse Response) – s nekonečnou impulsovou odezvou. Nejen, že jejich vlastnosti se od sebe liší, ale také realizace těchto filtrů se od sebe zásadním způsobem odlišuje. Filtry typu IIR se často realizují pomocí sekcí 2. řádu a jejich řazením do vhodné struktury. Filtry typu FIR mají pro dosažení požadovaných vlastností filtru obvykle podstatně vyšší řád než filtry typu IIR. Přímá realizace FIR se používá spíše výjimečně, filtry FIR s vysokým řádem jsou vždy realizovány v kmitočtové oblasti [1, 3]. Celkové srovnání výhod a nevýhod obou typů filtrů je uvedeno v tabulce 2.1.

2.1 Filtry s konečnou impulsovou odezvou – FIR

Číslicové filtry s konečnou impulsovou odezvou neobsahují žádnou zpětnou vazbu, jde tedy o filtry nerekurzivní. Potom je přenosová funkce takového filtru definovaná pouze polynomem v čitateli přenosové funkce $P(z)$

$$H(z) = P(z) = b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_N z^{-N} = \sum_{i=0}^{N} b_i z^{-i}. \tag{2.1}$$

Vyjádříme-li $H(z)$ v kladných mocninách z, dostaneme

$$H(z) = \frac{\sum_{i=0}^{N} b_i z^{N-i}}{z^N}, \tag{2.2}$$

kde N je řád filtru, délka impulsové odezvy je potom $N + 1$, b_i jsou koeficienty filtru a zároveň vzorky impulsní odezvy. Ve jmenovateli vystupuje operátor z^N, aby systém splnil podmínku kauzality a byl realizovatelný [4]. Současně je splněna i podmínka stability, neboť systém má pouze jeden N-násobný pól v bodě $z = 0$, a tedy všechny póly leží uvnitř jednotkové kružnice. Číslicový filtr typu FIR s přenosovou funkcí podle (2.2) je vždy kauzální a vždy stabilní.

Další předností číslicových filtrů typu FIR je možnost získat lineární fázovou kmitočtovou charakteristiku v celém rozsahu kmitočtů. Skupinové zpoždění τ (1.14) bude konstantní a všechny složky signálu budou průchodem číslicovým filtrem zpožděny o stejný časový interval.

Naopak nevýhodou číslicových filtrů typu FIR je vysoký řád N systému, pokud má realizovat přenosovou funkci se strmými přechody mezi kmitočtovými pásmy – typicky úzkopásmové filtry. S vysokým řádem souvisí i vysoké paměťové nároky při výpočtu, veliké zpoždění při zpracování vstupního signálu a prodlužování přechodných dějů v systému. Z praxe lze říct, že v porovnání s filtry typu IIR vychází pro stejné zadání filtru řád filtru typu FIR asi 10krát vyšší než u filtru typu IIR [3].

2.1.1 Metoda váhové posloupnosti

Jak již bylo uvedeno v (1.15), kmitočtová charakteristika číslicového filtru je periodickou funkcí úhlového kmitočtu s periodou 2π. Protože jde o periodickou funkci, lze ji rozvinout do Fourierovy řady. Koeficienty Fourierovy řady jsou právě koeficienty impulsové odezvy [3]

$$h(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H(\mathrm{e}^{\mathrm{j}\omega}) \mathrm{e}^{\mathrm{j}\omega n} \,\mathrm{d}\omega, \tag{2.3}$$

popř. při vzorkování

$$h(nT) = \frac{1}{2\omega_{\mathrm{vz}}} \int\limits_{-\frac{\omega_{\mathrm{vz}}}{2}}^{\frac{\omega_{\mathrm{vz}}}{2}} H(\mathrm{e}^{\mathrm{j}\omega T}) \mathrm{e}^{\mathrm{j}\omega nT} \,\mathrm{d}\omega, \tag{2.4}$$

kde ω_{vz} je vzorkovací kmitočet. Pro případ ideální dolní propusti popsané na obr. 2.1 a) platí pro kmitočtovou charakteristiku

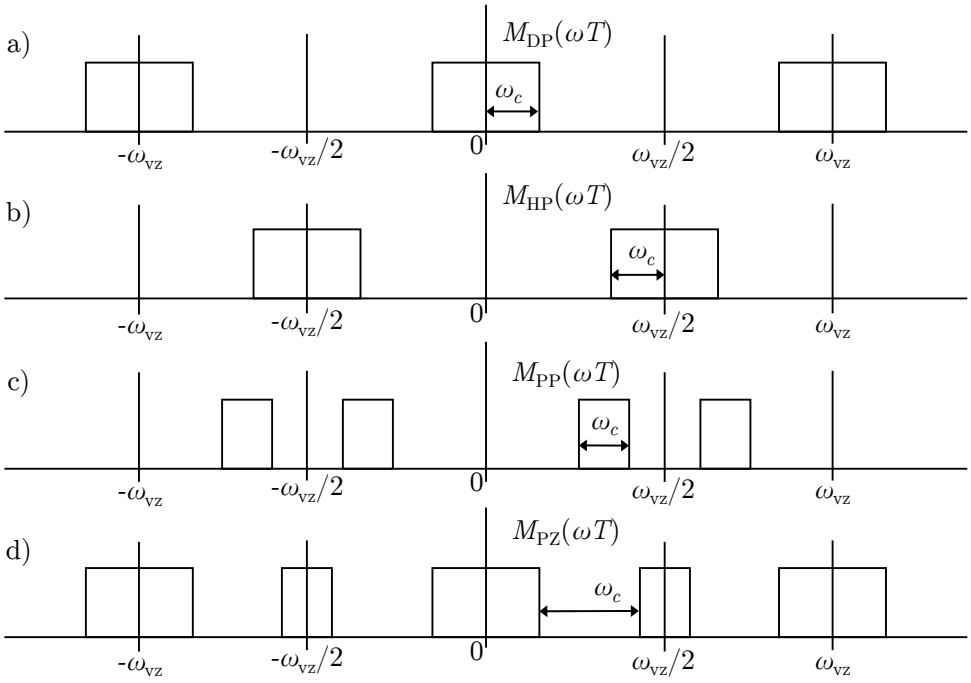
$$\begin{aligned} |H(\omega T)| &= A \quad \text{pro } |\omega| \leq \omega_c \\ |H(\omega T)| &= 0 \quad \text{pro } \omega_c < |\omega| \leq \frac{\omega_{\mathrm{vz}}}{2}, \end{aligned} \tag{2.5}$$

a dosazením této podmínky do rovnice (2.4) obdržíme tvar [3]

$$\begin{gathered} h_{\mathrm{DP}}(nT) = \frac{1}{2\omega_{\mathrm{vz}}} \int\limits_{-\omega_c}^{\omega_c} A \mathrm{e}^{\mathrm{j}\omega nT} \,\mathrm{d}\omega = 2A \frac{f_c}{f_{vz}} \operatorname{sinc}(\omega_c nT) \\ n = 0, \pm 1, \pm 2, \ldots. \end{gathered} \tag{2.6}$$

Vypočtená impulsová odezva však představuje nekauzální systém s nekonečnou impulsovou odezvou. Abychom vytvořili číslicový filtr s konečnou impulsovou odezvou, musíme zachovat pouze konečný počet N členů Fourierovy řady a s ohledem na zachování kauzality musíme impulsovou odezvu posunout (zpozdit) alespoň o poloviční hodnotu počtu členů, v případě lichého počtu N tedy o $(N-1)/2$. Tímto postupem získáme novou impulsovou odezvu kauzálního číslicového filtru typu FIR

$$\begin{aligned} h'(nT) &= h\left[\left(n - \frac{N-1}{2}\right) T\right] \quad \text{pro } n \in \langle 0, N-1 \rangle, \\ h'(nT) &= 0 \qquad\qquad\qquad\qquad\quad \text{pro ostatní } n. \end{aligned} \tag{2.7}$$



Obr. 2.1: Modulové kmitočtové charakteristiky ideálních číslicových filtrů: a) dolní propust, b) horní propust, c) pásmová propust, d) pásmová zádrž [4, obr. 6.1].

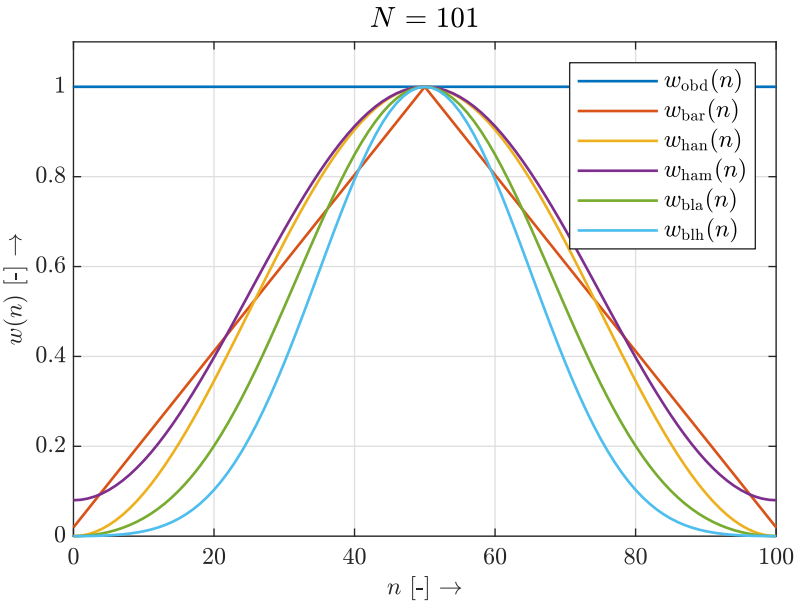
Rovnici (2.7) lze také vyjádřit jako součin původní impulsní charakteristiky $h(nT)$ s váhovou posloupností (nazývanou také oknem), která vybere požadovaný počet členů

$$h'(nT) = h\left[\left(n - \frac{N-1}{2}\right)T\right] w(nT). \tag{2.8}$$

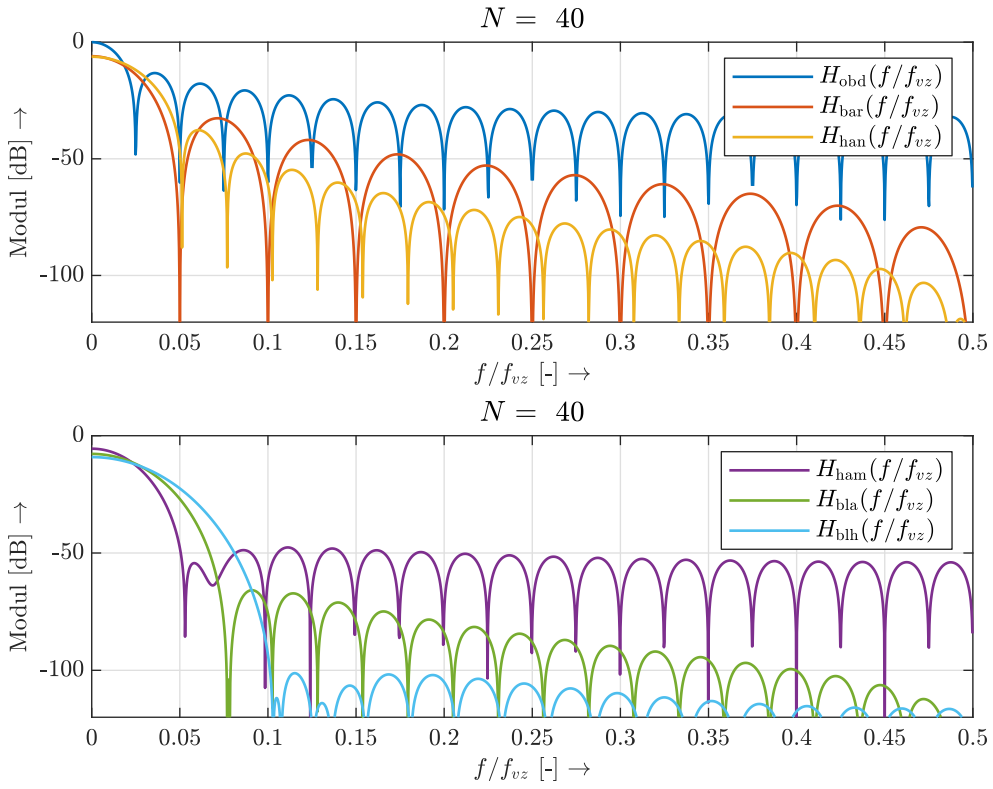
Jak je známo, součinu dvou funkcí v časové oblasti odpovídá komplexní konvoluce v oblasti kmitočtové obrazů jednotlivých funkcí [3].

Nejjednodušší váhovací posloupností je pravoúhlé okno, kdy ponecháme pouze daný počet vzorků impulsové odezvy $h(nT)$ a ostatní vynulujeme. Můžeme však použít libovolnou posloupnost s konečným počtem členů. Typem váhovacího okna výrazně ovlivníme vlastnosti navrženého filtru, jako je strmost v přechodném pásmu nebo útlum v nepropustném pásmu. Mezi základní typy patří okno obdélníkové, Bartlettovo, Hannovo, Hammingovo, Blackmannovo a Blackmann-Harrisovo. Jejich podrobnější popis a vlastnosti jsou k nalezení v [1, 3, 4].

Na obrázku 2.2 jsou srovnány časové průběhy uvedených váhovacích oken. Na dalším obrázku 2.3 jsou potom vyobrazeny modulové kmitočtové charakteristiky výše uvedených oken. N značí délku daného váhovacího okna.



Obr. 2.2: Srovnání časového průběhu posloupnosti váhovacích oken.



Obr. 2.3: Srovnání modulových kmitočtových charakteristika váhovacích oken.

2.2 Filtry s nekonečnou impulsovou odezvou – IIR

Číslicové filtry typu IIR mají nekonečnou impulsovou odezvu a pro jejich realizaci je nutné použití zpětných vazeb. Obecná přenosová funkce má tvar racionálně lomené funkce [4]

$$H(z) = \frac{\sum_{i=0}^{M} b_i z^{-i}}{\sum_{j=0}^{N} a_j z^{-j}}. \tag{2.9}$$

Má-li být systém kauzální, musí být polynom v čitateli přenosové funkce nejvýše stejného řádu jako je polynom ve jmenovateli. V rovnici (2.9) tedy musí platit $M \leq N$. Oproti filtrům typu FIR není vždy zaručena stabilita filtrů typu IIR. Po návrhu se vždy musí provést kontrola, zda všechny póly přenosové funkce leží uvnitř jednotkové kružnice. Také nelze navrhnout filtr s lineární fázovou charakteristikou v celém kmitočtovém rozsahu, lze se jí pouze přiblížit v úzkém rozsahu kmitočtů. Mezi nevýhody patří i velká citlivost na kvantování hodnot koeficientů přenosové funkce.

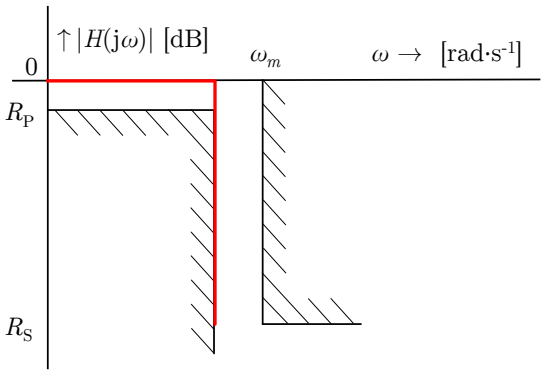
Naopak výhodou IIR filtrů je podstatně menší řád přenosové funkce oproti filtrům typu FIR, tedy i menší nároky na paměť a menší zpoždění při zpracování vstupního vzorku a rychlý průběh přechodových dějů [4].

Metody návrhu lze rozdělit do dvou skupin:

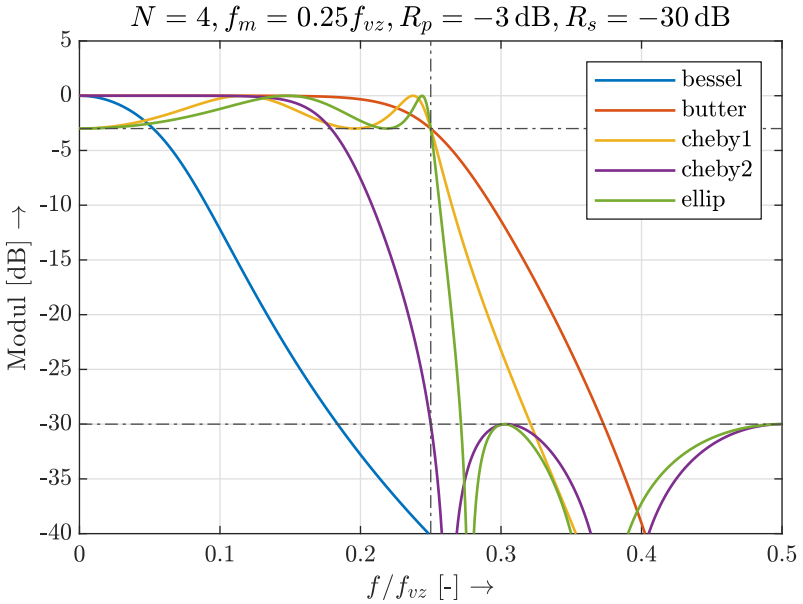
- **Analogově číslicové transformace** – tradiční a nejpopulárnější přístup k návrhu číslicových filtrů typu IIR. Vycházejí ze známých aproximací pro návrh analogových filtrů, které jsou velmi dobře zpracovány a existuje mnoho tabulek, vzorců a doporučení k návrhu. Známé jsou Butterworthovy, Besselovy, Čebyševovy nebo Cauerovy (eliptické) filtry. Číslicový filtr vznikne převodem z roviny p do roviny z pomocí např. bilineární transformace. Jejich průběhy pro filtr typu DP jsou na obr. 2.5.
- **Iterační optimalizační postupy** – přímé metody v časové oblasti nebo v rovině z bez návaznosti na analogové ekvivalenty. Mezi metody patří např. metoda nejmenších čtverců v časové a kmitočtové oblasti, identifikační parametrické metody používající modely typu MA (*Moving Average*), AR (*Auto Regressive*) a ARMA (*Auto Regressive Moving Average*) apod. (viz kap. 3) [3, 5].

2.2.1 Návrh pomocí analogového prototypu

Při návrhu analogového filtru se vychází z ideální normované analogové dolní propusti, která má mezní kmitočet na $1\,\mathrm{rad}\cdot\mathrm{s}^{-1}$, modul kmitočtové charakteristiky v propustném pásmu je 1 a v nepropustném pásmu 0. Takový systém je však nekauzální a jeho impulsová odezva začíná v bodě $-\infty$. Aproximací tohoto ideálního průběhu získáme kauzální systém. Při aproximaci dojde však ke zvlnění modulové kmitočtové charakteristiky a k rozšíření přechodového pásma. Pro popis aproximace modulové kmitočtové charakteristiky slouží toleranční schéma na obr. 2.4.



Obr. 2.4: Toleranční schéma modulové kmitočtové charakteristiky pro normovanou dolní propust (červeně znázorněna ideální dolní propust) [4, obr 7.1].



Obr. 2.5: Modulové kmitočtové charakteristiky základních typů aproximací normované analogové dolní propusti. Normovaný mezní kmitočet byl zvolen $0{,}25 f_{vz}$, řád filtru $N = 4$. Pro Čebyševovu aproximaci 1. typu a Cauerovu aproximaci je zvoleno zvlnění v propustném pásmu $R_\mathrm{p} = -3\,\mathrm{dB}$, pro Čebyševovu aproximaci 2. typu i Cauerovu je zvoleno zvlnění v nepropustném pásmu $R_\mathrm{s} = -30\,\mathrm{dB}$. Na zvoleném mezním kmitočtu f_m dosahují poklesu o 3 dB pouze aproximace Butterworthova, Čebyševova 1. typu a Cauerova. Čebyševova aproximace 2. typu dosahuje na zvoleném kmitočtu f_m hodnoty maximálního zvlnění nepropustného pásma.

Aproximace Besselova, Butterworthova mají modul kmitočtové charakteristiky monotónně klesající funkci, Čebyševova 1. a 2. typu, vykazují zvlnění v propustném, resp. nepropustném pásmu. Cauerova disponuje zvlněním modulové kmitočtové charakteristiky v obou pásmech. Oproti Butterworthově mají však mnohem strmější přechodové pásmo.

Tab. 2.1: Srovnání vlastností číslicových filtrů s konečnou (FIR) a nekonečnou (IIR) impulsovou odezvou [3].

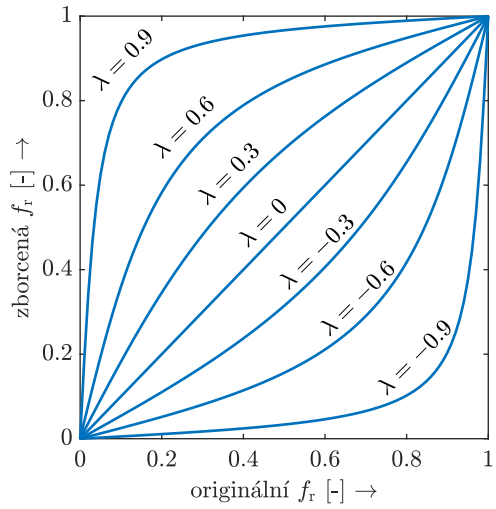
Číslicové filtry typu FIR		Číslicové filtry typu IIR	
Výhody	Nevýhody	Výhody	Nevýhody
Jsou vždy stabilní.	Velký řád přenosové funkce.	Malý řád přenosové funkce.	Nastávají problémy se stabilitou.
Mohou mít lineární fázovou kmitočtovou charakteristiku, neboli konstantní skupinové zpoždění.	Velké zpoždění při zpracování vstupního vzorku.	Malé zpoždění při zpracování vstupního vzorku.	Nemohou mít lineární fázovou kmitočtovou charakteristiku v celém rozsahu.
Mají menší citlivost na kvantování koeficientů a stavových proměnných.	Velké nároky na paměť při výpočtu koeficientů a stavových proměnných.	Malé nároky na paměť při výpočtu koeficientů a stavových proměnných.	Vlivem zpětných vazeb větší náchylnost k saturaci aritmetiky procesoru.
Jsou vhodné pro adaptivní algoritmy.	Optimální iterační metody jsou výpočetně náročné.	Jednoduché metody návrhu využívající vlastností analogových filtrů.	S obtížemi je lze použít pro adaptivní zpracování.
Existuje menší riziko saturace aritmetiky procesoru.	Neexistuje plnohodnotný analogový ekvivalent.	K číslicovému filtru lze najít analogový ekvivalent.	Velká citlivost na kvantování zvláště pro selektivní kmitočtové filtry.

2.3 Filtry navrhované s borcenou kmitočtovou osou

Ačkoliv byly filtry tohoto typu představeny před nemalou dobou, nejsou v povědomí tolik, jako klasické filtry typu FIR a IIR i když jejich použití s sebou nese několik výhod. Tyto filtry (angl. *warped filters*) využívají principu borcení kmitočtové osy (*frequency warping*) a jejich přenosová funkce se od klasických návrhů liší použitím kmitočtově závislého zpožďujícího bloku $\tilde{z}^{-1}$ – fázovacím článkem prvního řádu [7]. Pro zjednodušení bude v této práci použitý termín *warped filtry*.

$$\tilde{z}^{-1} = D_1(z) = \frac{z^{-1} - \lambda}{1 - \lambda z^{-1}} \tag{2.10}$$

Parametrem λ se ovlivňuje zkreslení kmitočtové osy tak, že kladné hodnoty λ zvýší rozlišení na nízkých kmitočtech a záporné hodnoty naopak zvýší rozlišení na kmitočtech vysokých (viz obr. 2.6).



Obr. 2.6: Příklad mapování kmitočtové osy při různých hodnotách λ.

Filtry, které vzniknou substitucí kmitočtově závislého bloku zpoždění $D_1(z)$, se označují jako WFIR – angl. *Warped FIR* a WIIR – angl. *Warped IIR*. O jejich realizaci a implementaci pojednává např. literatura [7, 8].

Dříve popsané FIR a IIR filtry vykazují lineární rozložení kmitočtové osy, avšak vnímání zvuku lidským uchem je logaritmické. Borcením kmitočtové osy je možné dosáhnout podobného rozlišení, jako má barková stupnice kritických pásem ($\lambda \doteq 0{,}7233$ [7]). Výhody těchto filtrů jsou následující:

- Návrh filtru může probíhat při zborceném rozlišení kmitočtové osy a přiblížit se tak vnímání kmitočtu zvukového vlnění člověkem.
- Při vhodném parametru λ jsou nulové body a póly rozmístěny na kmitočtové ose dál od sebe a návrh potom není tak citlivý na kvantování koeficientů přenosové funkce.
- Vhodně navržený *warped* filtr může mít oproti klasickému návrhu znatelně menší řád (až 5–10krát) [8].

Návrh filtru potom probíhá následovně. Cílová kmitočtová charakteristika (nebo impulsová odezva) se nejdříve namapuje na zborcenou kmitočtovou, resp. časovou osu, poté proběhne návrh filtru ve zborcené oblasti (například Pronyho nebo Steiglitz-McBride metodou, viz kap. 3) a poté se výsledný filtr namapuje ze zborcené oblasti zpět [7, 9]. Ukázky návrhu číslicového filtru touto metodou jsou v kapitole 5.

3 POKROČILÉ METODY NÁVRHU FILTRŮ

V předchozí kapitole byly představeny základní metody tradičních návrhů filtrů typu FIR a IIR. Tyto metody aproximují zvolený ideální průběh filtru (viz obr. 2.1) a jejich parametry jsou například maximální povolené zvlnění modulové kmitočtové charakteristiky $|H'(\omega)|$ v různých pásmech filtru oproti ideální modulové kmitočtové charakteristice $|H(\omega)|$.

Další kategorií metod návrhu filtrů jsou metody založené na statistickém modelování spektra; všechny metody uplatňují určité *kritérium optimality* E [6], které vychází z chybové funkce $e(n) = x(n) - x'(n)$, kde $x(n)$ je obecně originální signál a $x'(n)$ je jeho odhad. Nejčastěji se volí metoda nejmenších čtverců, kdy se tento chybový signál umocní na druhou a hodnota součtu tohoto signálu se minimalizuje.

Použití těchto metod je vhodné v případě analýzy stochastických signálů, např. při odhadu výkonového spektra, avšak najdou uplatnění i při použití deterministických signálů, resp. pro návrh invariantních číslicových filtrů [5].

Společným jmenovatelem níže popsaných metod je řešení *přeurčené soustavy lineárních rovnic* (angl. *overdetermined system*) – je zadáno více rovnic, než je neznámých. Toto je klasický problém vyskytující se při aplikaci lineární algebry, a při použití metody nejmenších čtverců existuje elegantní řešení této soustavy, které si popíšeme níže [5, 10].

3.1 Přeurčená soustava rovnic

Nechť je zadaná následující soustava lineárních rovnic. Při použití maticového zápisu ji můžeme zapsat ve formě

$$\mathbf{F}\mathbf{x} = \mathbf{g}, \tag{3.1}$$

kde $\mathbf{F}$ je $L \times N$ matice (L řádku a N sloupců), $\mathbf{x}$ je sloupcový vektor neznámých délky N a $\mathbf{g}$ je sloupcový vektor délky L. Jestliže $L = N$, potom je matice $\mathbf{F}$ čtvercová a pokud je matice regulární, potom existuje právě jedno řešení $\mathbf{x}$ rovnice dáno

$$\mathbf{x} = \mathbf{F}^{-1}\mathbf{g}, \tag{3.2}$$

kde $\mathbf{F}^{-1}$ je inverzní matice k $\mathbf{F}$. Jestliže však $L > N$, potom $\mathbf{F}$ je obdélníková matice s větším počtem řádků než sloupců. V tomto případě nelze najít dokonalé řešení $\mathbf{x}$, aby vyhovovalo pro všechna $\mathbf{g}$. Proto je třeba upravit rovnici (3.1) tím způsobem, že se přidá chybový vektor $\mathbf{e}$

$$\mathbf{F}\mathbf{x} = \mathbf{g} + \mathbf{e}. \tag{3.3}$$

Poté se zvolí vhodné kritérium optimality pro minimalizaci, nejčastěji se volí kvadratická odchylka (LSE – *Least Squares Error*) [10]

$$E = \mathbf{e}^\top \mathbf{e} = \sum_{i=0}^{L-1} e_i^2, \tag{3.4}$$

kde $\top$ značí transpozici vektoru.

Pokud chybový vektor $\mathbf{e}$ zapíšeme jako $(\mathbf{Fx} - \mathbf{g})$ a dosadíme do předchozí rovnice, dostaneme [5]

$$\begin{aligned} E = (\mathbf{Fx} - \mathbf{g})^\top (\mathbf{Fx} - \mathbf{g}) = (\mathbf{x}^\top \mathbf{F}^\top - \mathbf{g}^\top)(\mathbf{Fx} - \mathbf{g}) = \\ = \mathbf{x}^\top \mathbf{F}^\top \mathbf{Fx} - 2\mathbf{x}^\top \mathbf{F}^\top \mathbf{g} + \mathbf{g}^\top \mathbf{g}. \end{aligned} \tag{3.5}$$

Jestliže E má být minimalizováno vzhledem k vhodně zvolenému řešení $\mathbf{x}$, potom musí platit, že jeho parciální derivace dle vektoru neznámých musí být rovna 0 [5, 10]

$$\frac{\partial E}{\partial \mathbf{x}} = \mathbf{0}. \tag{3.6}$$

Parciální derivací výsledku rovnice (3.5) dle (3.6) dostaneme [5]

$$2\mathbf{F}^\top \mathbf{Fx} - 2\mathbf{F}^\top \mathbf{g} = 0 \quad \Rightarrow \quad \mathbf{F}^\top \mathbf{Fx} = \mathbf{F}^\top \mathbf{g} \tag{3.7}$$

Rovnice (3.7) je soustava *normálních* rovnic pro řešení metodou nejmenších čtverců. Povšimněme si, že když vynásobíme obě strany rovnice z dřívějška (3.3) $\mathbf{F}^\top$, potom obdržíme

$$\mathbf{F}^\top \mathbf{Fx} = \mathbf{F}^\top \mathbf{g} + \mathbf{F}^\top \mathbf{e}. \tag{3.8}$$

Porovnáním (3.7) a (3.8) vyjde

$$\mathbf{F}^\top \mathbf{e} = \mathbf{0}. \tag{3.9}$$

To je důkaz principu ortogonality, který říká, že všechny složky vektoru $\mathbf{e}$ musí být ortogonální vzhledem ke každému sloupci matice $\mathbf{F}$ [5, 11].

Jestliže matice $\mathbf{F}^\top \mathbf{F}$ o rozměru $N \times N$ je regulární, potom je řešení normálních rovnic (3.7) jednoznačné a vypočítá se vztahem

$$\mathbf{x} = (\mathbf{F}^\top \mathbf{F})^{-1} \mathbf{F}^\top \mathbf{g} \quad \text{nebo} \quad \mathbf{x} = \mathbf{F}^{\#} \mathbf{g}, \tag{3.10}$$

kde $\mathbf{F}^{\#} = (\mathbf{F}^\top \mathbf{F})^{-1} \mathbf{F}^\top$ značí **pseudoinverzní** matici k $\mathbf{F}$ (někdy také zobecněná inverze nebo *Moore-Penroseovská*).

Zpět ke struktuře číslicového filtru. V mnohých literaturách nalezneme další terminologii rozdělení filtrů podle zastoupení nulových bodů a pólu v přenosové funkci filtru. Toto rozdělení je:

- **AR** (*autoregressive*) – autoregresní, přenosová funkce obsahuje pouze póly.
- **MA** (*moving average*) – klouzavý průměr, přenosová funkce obsahuje pouze nulové body.
- **ARMA** (*autoregressive moving average*) – přenosová funkce obsahuje i póly i nulové body.

Blokové schéma ARMA číslicového filtru představuje například obr. 1.1, kde vidíme dopřednou – nerekurzivní část, která vytváří vážený klouzavý průměr z daného počtu přicházejících hodnot, a zpětnovazební – rekurzivní část, která vytváří vážený součet ze zpožděných hodnot výstupu. Obecně lze říci, že metody MA a ARMA se uplatňují méně

často, kvůli větší výpočetní náročnosti – identifikace koeficientů filtru vede na nelineární normální rovnice. Filtry založené na AR modelech lépe vystihují úzkopásmové složky v signálu (tzv. *emisní spektra*), časté uplatnění naleznou při řečové analýze. Naproti tomu MA modely jsou vhodnější v případech širokopásmových signálů s potlačenými úzkými pásmy (*absorpční spektra*) [6].

3.2 Návrh AR filtrů

Uvažujme autoregresní model číslicového filtru řádu N. Jeho přenosovou funkci potom můžeme zapsat ve tvaru

$$H(z) = \frac{1}{A(z)}, \tag{3.11}$$

kde $A(z) = a_0 + a_1 z^{-1} + \cdots + a_N z^{-N}$.

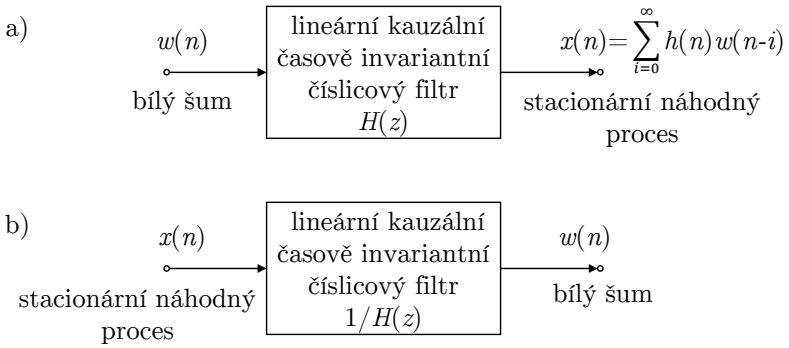
Potom platí

$$H(z)A(z) = 1, \tag{3.12}$$

nebo, po zpětné $\mathcal{Z}$-transformaci, v časové oblasti

$$h(n) * a(n) = \delta(n) \tag{3.13}$$

FIR filtr $A(z)$ je *inverzním* filtrem k $H(z)$ a nese také označení **bělicí** filtr. Pojem bělicího filtru nalezneme i ve spojitosti se spektrální analýzou stochastických signálů při odhadu výkonového spektra [2, 6]. Jeho účel je stejný – transformace analyzovaného signálu na signál, jehož spektrum je „bílé“, tedy jeho výkonová spektrální hustota je konstantní. V případě náhodných procesů je možné ilustrovat funkci bělicího filtru na obr. 3.1, kde výsledný signál je bílý šum.



Obr. 3.1: Koncept bělicího filtru [2, obr. 12.1].

V případě rovnice (3.13) je výsledným signálem se stejnými spektrálními vlastnostmi jednotkový impuls $\delta(n)$. Při zadané $h(n)$ pouze v konečném intervalu 0 až N a za předpokladu, že $h(n) = 0$ pro $n < 0$, lze přepsat konvoluci na maticovou operaci, která repre-

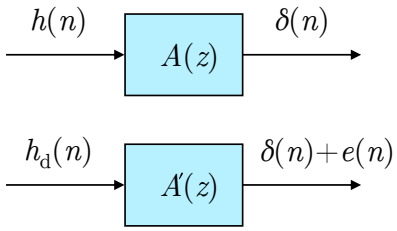
zentuje soustavu $N+1$ rovnic pro vektor neznámých $a(n)$

$$\begin{bmatrix} h_0 & 0 & \cdots & 0 \\ h_1 & h_0 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ h_N & & \cdots & h_0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{3.14}$$

Avšak tím, že zadaná cílová, nebo změřená impulsová odezva $h_\mathrm{d}(n)$[1] nemá úplně zcela přesně AR charakter, nebo je vyššího řádu než použitá aproximace v (3.11), můžeme tak pouze určit odhad inverzního filtru. Poté rovnice (3.13) změní tvar na [5]

$$h_\mathrm{d}(n) * a(n) = \delta(n) + e(n), \tag{3.15}$$

kde $e(n)$ je chyba aproximace a $A(z)$ je nyní pouze aproximací inverzního filtru, jako je znázorněno na obr. 3.2.



Obr. 3.2: Funkce inverzního filtru při skutečné impulsové odezvě číslicového filtru a při použití zadané aproximaci impulsové odezvy [5, obr. 10.1].

Maticový zápis rovnice (3.15) má tvar [5]

$$\mathbf{H}_\mathrm{d}\mathbf{a} = \boldsymbol{\delta} + \mathbf{e}, \tag{3.16}$$

kde složky matice $\mathbf{H}_\mathrm{d}$ jsou

$$\mathbf{H}_\mathrm{d} = \begin{bmatrix} h_{\mathrm{d}0} & 0 & \cdots & 0 \\ h_{\mathrm{d}1} & h_{\mathrm{d}0} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ h_{\mathrm{d}N} & h_{\mathrm{d},N-1} & \cdots & h_{\mathrm{d}0} \\ \vdots & & & \vdots \\ h_{\mathrm{d}L} & & \cdots & h_{\mathrm{d},L-N} \end{bmatrix} \tag{3.17}$$

a

$$\begin{aligned} \mathbf{a} &= [a_0, a_1, \ldots, a_N]^\top, \\ \boldsymbol{\delta} &= [1, 0, 0, \ldots, 0]^\top, \\ \mathbf{e} &= [e_0, e_1, \ldots, e_L]^\top. \end{aligned}$$

[1] V literatuře se používá index d, angl. *desired* – požadovaná, zadaná, někdy také index t – *target*.

Oba vektory pravé strany rovnice $\boldsymbol{\delta}, \mathbf{e}$ mají délku $L+1$, stejně jako délka zadané impulsové odezvy $h_\mathrm{d}(n)$, samozřejmě za splnění podmínky $L > N$. Naším cílem je minimalizovat střední kvadratickou odchylku

$$E_N = \mathbf{e}^\top \mathbf{e} = \sum_{n=0}^{L} e^2(n). \tag{3.18}$$

Toto je opět klasický příklad řešení přeurčené soustavy rovnic metodou nejmenších čtverců, jak již bylo popsáno v kapitole 3.1. Řešení pro vektor neznámých $\mathbf{a}$ dostaneme vynásobením obou stran maticí $\mathbf{H}_\mathrm{d}^\top$, čímž obdržíme

$$\mathbf{H}_\mathrm{d}^\top \mathbf{H}_\mathrm{d} \mathbf{a} = \mathbf{H}_\mathrm{d}^\top \boldsymbol{\delta} + \mathbf{H}_\mathrm{d}^\top \mathbf{e}. \tag{3.19}$$

Již bylo dokázáno, že $\mathbf{H}_\mathrm{d}^\top \mathbf{e} = 0$, takže řešení opět využívá pseudoinverzní transformaci matice $\mathbf{H}_\mathrm{d}$

$$\mathbf{a} = \mathbf{H}_\mathrm{d}^{\#} \boldsymbol{\delta}. \tag{3.20}$$

Je dobré si povšimnout, že $\mathbf{H}_\mathrm{d}^\top \boldsymbol{\delta} = h_{\mathrm{d}0} \boldsymbol{\delta}$, tedy pouze vážený jednotkový impuls, kde délka vektoru $\boldsymbol{\delta}$ je $L + 1$.

3.2.1 Kovarianční metoda

Kompaktnějším zápisem rovnice (3.19) vyjádříme soustavu normálních rovnic

$$\boldsymbol{\Phi}_N \mathbf{a} = h_{\mathrm{d}0} \boldsymbol{\delta}, \tag{3.21}$$

kde $\boldsymbol{\Phi}_N$ je symetrická kovarianční matice o rozměrech $(N+1) \times (N+1)$

$$\boldsymbol{\Phi}_N = \mathbf{H}_\mathrm{d}^\top \mathbf{H}_\mathrm{d}, \tag{3.22}$$

a potom je řešením

$$\mathbf{a} = h_{\mathrm{d}0} \boldsymbol{\Phi}_N^{-1} \boldsymbol{\delta}. \tag{3.23}$$

Vektor $\mathbf{a}$ je tedy úměrný prvnímu sloupci inverzní kovarianční matice. Jelikož všechny sloupce $\mathbf{H}_\mathrm{d}$ jsou lineárně nezávislé, platí, že matice $\boldsymbol{\Phi}_N$ je invertovatelná. Tento přístup se obecně nazývá *kovarianční metoda lineární predikce* [5, 6].

3.2.2 Korelační metoda

Předpokládejme nyní, že bychom minimalizovali chybu z rovnice (3.18) na nekonečném intervalu $-\infty < n < \infty$. Potom přepíšeme rovnici (3.21) na tvar

$$\mathbf{R}_N \mathbf{a} = h_{\mathrm{d}0} \boldsymbol{\delta}, \tag{3.24}$$

kde $\mathbf{R}_N$ reprezentuje symetrickou Toeplitzovu **autokorelační** matici, jejíž složky popisuje signál $r(m)$

$$r(m) = \lim_{N \to \infty} \sum_{n=-N}^{N} h(n) h(n+m), \tag{3.25}$$

kde $h(n)$ je nyní ideální impulsová odezva filtru definovaná na nekonečném intervalu.

$$\mathbf{R}_N = \begin{bmatrix} r_0 & r_1 & \cdots & r_N \\ r_1 & r_0 & & \vdots \\ \vdots & & \ddots & \\ h_N & & \cdots & r_0 \end{bmatrix} \tag{3.26}$$

Reálně však máme impulsovou odezvu definovanou pouze v omezeném časovém intervalu $n = 0, 1, \ldots, L$, a tedy nemůžeme hovořit o čistě autokorelační matici, pouze o její aproximaci. Prodloužením dané impulsové odezvy $h_\mathrm{d}(n)$ nulami sestrojíme odhad autokorelační matice $\widehat{\mathbf{R}}_N$ [5]

$$\widehat{\mathbf{R}}_N = \widehat{\mathbf{H}}_\mathrm{d}^\top \widehat{\mathbf{H}}_\mathrm{d} = \begin{bmatrix} \hat{r}_0 & \hat{r}_1 & \cdots & \hat{r}_N \\ \hat{r}_1 & \hat{r}_0 & & \vdots \\ \vdots & & \ddots & \\ \hat{r}_N & & \cdots & \hat{r}_0 \end{bmatrix}, \tag{3.27}$$

kde

$$\widehat{\mathbf{H}}_d = \begin{bmatrix} h_{\mathrm{d}0} & 0 & \cdots & 0 \\ h_{\mathrm{d}1} & h_{\mathrm{d}0} & \cdots & 0 \\ & & \ddots & \vdots \\ \vdots & & & h_{\mathrm{d}0} \\ h_{\mathrm{d}L} & & & \\ 0 & h_{\mathrm{d}L} & & \vdots \\ \vdots & & \ddots & \\ 0 & & \cdots & h_{\mathrm{d},L} \end{bmatrix} \tag{3.28}$$

a rovnice pro autokorelační funkci (3.25) potom změní své meze na

$$\hat{r}(m) = \sum_{n=0}^{L-m} h_\mathrm{d}(n) h_\mathrm{d}(n+m). \tag{3.29}$$

Tato metoda nese v literatuře také název *autokorelační metoda lineární predikce* [5]. Řešení pro vektor **a** má potom podobný tvar jako (3.23).

Vedlejším efektem autokorelační metody je, že při vyjmutí konečně dlouhé $h_\mathrm{d}(n)$ z $h(n)$ vlastně provedeme operaci vynásobení dané posloupnosti obdélníkovým oknem $w(n)$ délky $L+1$, a to se poté projeví v kmitočtové charakteristice výsledného číslicového filtru $H(\omega)$, jenž vznikne konvolucí původní požadované kmitočtové charakteristiky $H_\mathrm{d}(\omega)$ s kmitočtovou charakteristikou zvoleného okna, čímž se do návrhu vnese další nepřesnost. Na druhou stranu výhodou této metody je, že výsledný inverzní filtr $A(z)$ bude vždy s minimální fází, což znamená, že $H(z)$ bude vždy stabilní. Stabilita návrhu naopak není zaručena u kovarianční metody, avšak pro stabilní $h_\mathrm{d}(n)$ metoda obvykle vytvoří stabilní model [5].

3.3 Návrh ARMA filtrů

Návrh modelu ARMA filtru je dle očekávání složitější, než návrh oddělených sekcí AR nebo MA. V představených metodách návrhu AR filtru byla minimalizována chybová funkce $e(n)$ popsána v rovnici(3.15)

$$e(n) = h_\mathrm{d}(n) * a(n) - \delta(n). \tag{3.30}$$

Takto zadaná chybová funkce bývá referována jako *chyba inverzního filtru*[2]. Různé přístupy uvádějí, že tato chybová funkce není nejvhodnější k nalezení nejlepšího řešení $\mathbf{a}$. Zavádí se jiná definice chybové funkce, která nese označení *chyba návrhu*[3] a je definovaná [5]

$$e(n) = h_\mathrm{d}(n) - h(n). \tag{3.31}$$

Kritériem pro minimalizaci se poté stává

$$E_N = \sum_{n=0}^{L} [h_\mathrm{d}(n) - h(n)]^2. \tag{3.32}$$

Takto zadané kritérium však vede na soustavu nelineárních rovnic pro řešení $\mathbf{a}$, která musí být vyřešena iterativně. Obecně však může být pro výpočet chybové funkce použito mnoho rovnic, které mohou, nebo nemusí vést k optimálnímu řešení. Každá metoda produkuje trochu jiné výsledky, tak je v praxi třeba zkusit, která se pro danou aplikaci hodí nejvíce [5].

3.3.1 Pronyho metoda

Úkolem je najít model přenosové funkce ARMA filtru

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + \cdots + b_M z^{-M}}{1 + a_1 z^{-1} + \cdots + a_N z^{-N}}. \tag{3.33}$$

Zavedeme pouze podmínku $a_0 = 1$, poté $H_d(z) \approx H(z)$, nebo v jiném tvaru

$$H_d(z)A(z) \approx B(z). \tag{3.34}$$

Přepisem do časové oblasti obdržíme soustavu lineárních rovnic pro $a(n)$ a $b(n)$ [5]

$$h_\mathrm{d}(n) * a(n) = b(n) + e(n), \tag{3.35}$$

kde $e(n)$ nyní nemá reálný fyzikální význam, ale je to *chyba stanovená rovnicí*[4]. Při stanovení $e(n) = 0$ pro $n \leq M$, kde M je řád čitatele přenosové funkce filtru, lze přepsat rovnici (3.35) do maticového tvaru

$$\mathbf{H}_\mathrm{d}\mathbf{a} = \mathbf{b}_L \quad \Leftrightarrow \quad \begin{bmatrix} \mathbf{H}_\mathrm{d1} \\ \hdashline \mathbf{H}_\mathrm{d2} \end{bmatrix} \mathbf{a} = \begin{bmatrix} \mathbf{b} \\ \hdashline \mathbf{e}_2 \end{bmatrix}. \tag{3.36}$$

[2] angl. *inverse-filter error*
[3] angl. *fitting error*
[4] angl. *equation error*

Matice $\mathbf{H}_\mathrm{d}$ vystupuje v rovnici (3.17) a zde má stejný tvar. Pro pochopení rovnice (3.36) znázorníme její jednotlivé členy [5]

$$\begin{bmatrix} h_{\mathrm{d}0} & 0 & \cdots & 0 \\ h_{\mathrm{d}1} & h_{\mathrm{d}0} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ h_{\mathrm{d}N} & h_{\mathrm{d},N-1} & \cdots & h_{\mathrm{d}0} \\ \vdots & & & \vdots \\ h_{\mathrm{d}L} & & \cdots & h_{\mathrm{d},L-N} \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} b_0 \\ \vdots \\ b_M \\ e_{N+1} \\ \vdots \\ e_L \end{bmatrix} \tag{3.37}$$

a označíme její jednotlivé části

$$\begin{bmatrix} h_{\mathrm{d}0} & 0 & \cdots & 0 \\ \vdots & & & \vdots \\ h_{\mathrm{d}M} & & \cdots & h_{\mathrm{d},M-N} \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} b_0 \\ \vdots \\ b_M \end{bmatrix}, \tag{3.38}$$
$$\begin{bmatrix} h_{\mathrm{d},M+1} & \cdots & h_{\mathrm{d},M-N+1} \\ \vdots & & \vdots \\ h_{\mathrm{d}L} & \cdots & h_{\mathrm{d},L-N} \end{bmatrix} \qquad \begin{bmatrix} e_{N+1} \\ \vdots \\ e_L \end{bmatrix}$$

kde vrchní část matice $\mathbf{H}_\mathrm{d}$ označíme jako $\mathbf{H}_{\mathrm{d}1}$ a spodní část jako $\mathbf{H}_{\mathrm{d}2}$. Vrchní část vektoru $\mathbf{b}_L$ obsahuje koeficienty čitatele přenosové funkce $\mathbf{b}$ a spodní část, označena jako $\mathbf{e}_2$ je nenulová část chyby aproximace $e(n)$. Minimalizací $\mathbf{e}_2^\top \mathbf{e}_2$ a použitím kovarianční metody získáme odhad řešení $\mathbf{a}$. Při dalším rozdělení $\mathbf{H}_{\mathrm{d}2}$ způsobem $\left[\begin{array}{cc} \mathbf{h}_{\mathrm{d}2} & \mathbf{H}'_{\mathrm{d}2} \end{array}\right]$ kde

$$\mathbf{h}_{\mathrm{d}2} = \begin{bmatrix} h_{\mathrm{d},M+1} \\ \vdots \\ h_{\mathrm{d}L} \end{bmatrix}, \quad \mathbf{H}'_{\mathrm{d}2} = \begin{bmatrix} h_{\mathrm{d},M} & \cdots & h_{\mathrm{d},M-N+1} \\ \vdots & & \vdots \\ h_{\mathrm{d},L-1} & \cdots & h_{\mathrm{d},L-N} \end{bmatrix}, \tag{3.39}$$

tedy první sloupec matice $\mathbf{H}_{\mathrm{d}2}$ a zbylé sloupce, a také definicí

$$\mathbf{a} = \left[\begin{array}{c} 1 \\ \hdashline \mathbf{a}' \end{array}\right], \tag{3.40}$$

můžeme zapsat spodní část rovnice (3.36) jako

$$\mathbf{H}'_{\mathrm{d}2}\mathbf{a}' = -\mathbf{h}_{\mathrm{d}2} + \mathbf{e}_2 \tag{3.41}$$

a tedy

$$\mathbf{a}' = -(\mathbf{H}'_{\mathrm{d}2})^{\#}\mathbf{h}_{\mathrm{d}2}. \tag{3.42}$$

Tímto obdržíme řešení pro koeficienty $\mathbf{a}$ jmenovatele přenosové funkce filtru. Nyní je třeba určit koeficienty čitatele $\mathbf{b}$. Dle vrchní části rovnice (3.36) je lze určit jako

$$\mathbf{b} = \mathbf{H}_{\mathrm{d}1}\mathbf{a}. \tag{3.43}$$

3.3.2 Steiglitz-McBride metoda

Steiglitz-McBride metoda je mocná iterativní metoda aproximace cílové impulsové odezvy IIR filtru s ARMA strukturou. Zvolené kritérium, které se má minimalizovat, vychází z rovnice chyby návrhu filtru

$$e(n) = h_{\mathrm{d}}(n) - h(n). \tag{3.44}$$

Kritérium se potom stává

$$E_N = \sum_{n=0}^{L} [h_{\mathrm{d}}(n) - h(n)]^2, \tag{3.45}$$

Avšak výsledné normální rovnice vedou na nelineární řešení pro vektor $\mathbf{a}$. Ve [12] byl však představen algoritmus, který nelineární řešení převede na iterativní způsob řešení problému lineárního.

AR verze Steiglitz-McBride

Uveďme prvně pouze verzi Steiglitz-McBride algoritmu pro filtr se strukturou AR. S pomocí rovnice (3.13) můžeme napsat [5]

$$h_{\mathrm{d}}(n) = h_{\mathrm{d}}(n) * \delta(n) = h_{\mathrm{d}}(n) * [h(n) * a(n)] = h(n) * h_{\mathrm{d}}(n) * a(n). \tag{3.46}$$

Přepisem do maticové formy získáme

$$\mathbf{h}_{\mathrm{d}} = \mathbf{H}\mathbf{H}_{\mathrm{d}}\mathbf{a} = \mathbf{h} + \mathbf{e}, \tag{3.47}$$

kde

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & \cdots & h_L \\ h_1 & h_0 & & \vdots \\ \vdots & & \ddots & \\ h_L & & \cdots & h_0 \end{bmatrix}, \mathbf{h}_d = \begin{bmatrix} h_{\mathrm{d}0} \\ h_{\mathrm{d}1} \\ \vdots \\ h_{\mathrm{d}L} \end{bmatrix}, \mathbf{h} = \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_L \end{bmatrix}. \tag{3.48}$$

Aproximací řešení je tedy

$$\mathbf{a} = (\mathbf{H}\mathbf{H}_{\mathrm{d}})^{\#}\mathbf{h} = (\mathbf{H}_{\mathrm{d}}^{\top}\mathbf{H}^{\top}\mathbf{H}\mathbf{H}_{\mathrm{d}})^{-1}\mathbf{h}. \tag{3.49}$$

Řešení je pouze odhadem, protože byl zamítnut fakt, že samotný model $h(n)$ a tedy i $\mathbf{H}$ a $\mathbf{h}$ jsou funkcí $\mathbf{a}$ [5]. Produkt násobení $\mathbf{H}\mathbf{H}_{\mathrm{d}}$ vlastně odpovídá konvoluci $h(n) * h_{\mathrm{d}}(n)$, čímž „**předfiltrujeme**“ zadanou impulsovou odezvu $h_{\mathrm{d}}(n)$ aktuálně navrženým modelem $h(n)$ a teprve poté vypočítáme $\mathbf{a}$. Takto můžeme postupovat iterativním způsobem, že i-tý odhad $\mathbf{a}_i$ můžeme vypočítat s použitím odpovídajících matic $\mathbf{H}_i$ a $\mathbf{h}_i$

$$\mathbf{a}_{i+1} = (\mathbf{H}_i\mathbf{H}_{\mathrm{d}})^{\#}\mathbf{h}_i. \tag{3.50}$$

Konvergence této metody je rapidní (postačí mezi 5 až 10 iteracemi). Pro počáteční odhad $\mathbf{a}_0$ může být použita kovarianční nebo autokorelační metoda zmíněné v kapitole 3.2. Potenciální nestabilita konvergence této metody může nastat, když zvolíme řád jmenovatele přenosové funkce N větší, než bude řád zadání $h_{\mathrm{d}}(n)$, je tedy důležité nezvolit N příliš vysoké [5].

ARMA verze Steiglitz-McBride

Zavedeme nyní matici popisující pouze jmenovatel přenosové funkce filtru (inverzní filtr) $H_a(z) = 1/A(z)$

$$\mathbf{H}_a = \begin{bmatrix} h_{a0} & 0 & \cdots & 0 \\ h_{a1} & h_{a0} & & 0 \\ \vdots & & \ddots & \vdots \\ h_{aL} & & \cdots & h_{a0} \end{bmatrix}, \tag{3.51}$$

a její část z pouze prvních $M+1$ sloupců

$$\mathbf{H}_a^{'} = \begin{bmatrix} h_{a0} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ h_{aM} & \cdots & h_{a0} \\ \vdots & & \vdots \\ h_{aL} & \cdots & h_{aL-M} \end{bmatrix}. \tag{3.52}$$

Můžeme potom zapsat rovnici podobným způsobem, jako v (3.47)

$$\mathbf{h}_d = \mathbf{H}_a\mathbf{H}_\mathrm{d}\mathbf{a} = \mathbf{h} + \mathbf{e} = \mathbf{H}_a^{'}\mathbf{b} + \mathbf{e}, \tag{3.53}$$

ze které potom vyjádříme řešení pro $\mathbf{a}$

$$\mathbf{a} = (\mathbf{H}_a\mathbf{H}_\mathrm{d})^{\#}\mathbf{H}_a^{'}\mathbf{b}. \tag{3.54}$$

Řešení pro vektor $\mathbf{b}$ je potom

$$\mathbf{h} = \mathbf{H}_a^{'}\mathbf{b} = \mathbf{h}_\mathrm{d} - \mathbf{e} \quad \Rightarrow \quad \mathbf{b} = (\mathbf{H}_a^{'})^{\#}\mathbf{h}_\mathrm{d}. \tag{3.55}$$

Abychom odstranili explicitní závislost výpočtu $\mathbf{a}$ na $\mathbf{b}$, dosadíme (3.55) do (3.54) a obdržíme řešení pro jednotlivé iterace $\mathbf{a}_i$

$$\mathbf{a}_{i+1} = (\mathbf{H}_{ai}\mathbf{H}_\mathrm{d})^{\#}\mathbf{H}_{ai}^{'}(\mathbf{H}_{ai}^{'})^{\#}\mathbf{h}_\mathrm{d}. \tag{3.56}$$

Opět je třeba udělat prvotní odhad $\mathbf{a}_0$, kde kvalitní výsledky poskytne např. Pronyho metoda. Po dostatečném počtu konvergujících iterací se vypočtou koeficienty $\mathbf{b}$ dle (3.55).

Velmi důležitý a speciální případ nastane při podmínce $M = N-1$, tedy řád polynomu jmenovatele je o jeden větší, než řád čitatele. Impulsová odezva odpovídá sumě exponenciálně utlumených harmonických funkcí [5]. Pro takto zadaný filtr existuje řešení popsané v [12], které využívá metodu pro počáteční odhad $\mathbf{a}_0$ (i $\mathbf{b}_0$) kterou uvedl R. E. Kallman. Pro výpočet parametrů se používá *pseudokorelační* matice vstupního a výstupního signálu, zde vstupním signálem cílová impulsová odezva filtru $h_\mathrm{d}(n)$ a výstupním opravdová impulsová odezva navrženého filtru $h(n)$.

Kallmanova metoda spočívá v minimalizaci $e(n)$ stanoveným rovnicí [12]

$$e(n) = b(n) * h_\mathrm{d}(n) - a(n) * h(n) - h(n). \tag{3.57}$$

Stanovíme vektor koeficientů $\mathbf{p}$ a vstupně výstupní vektor modelovou matici $\mathbf{Q}$

$$\mathbf{p} = \begin{bmatrix} b_0 & \cdots & b_M & -a_1 & \cdots - a_N \end{bmatrix}^\top$$
$$\mathbf{Q} = \begin{bmatrix} h_{\mathrm{d}0} & \cdots & h_{\mathrm{d}M} & h_{\mathrm{d}M+1} & \cdots & h_{\mathrm{d}L} \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \cdots & h_{\mathrm{d}0} & h_{\mathrm{d}1} & \cdots & h_{\mathrm{d}L-M} \\ 0 & h_0 & \cdots & h_N & \cdots & h_{L-1} \\ \vdots & \ddots & \ddots & \vdots & & \vdots \\ 0 & \cdots & 0 & h_0 & \cdots & h_{L-N-1} \end{bmatrix} \tag{3.58}$$

a rovnice (3.57) se s pomocí těchto matic dá zapsat

$$\mathbf{e} = \mathbf{Q}^\top \mathbf{p} - \mathbf{h}. \tag{3.59}$$

Dle podmínky ortogonality musí parciální derivace kritéria optimality $E = \mathbf{e}^\top \mathbf{e}$ dle vektoru neznámých $\mathbf{p}$ být 0 [12]

$$\frac{\partial \mathbf{e}^\top \mathbf{e}}{\partial \mathbf{p}} = 2\mathbf{Q}\mathbf{e} = 0. \tag{3.60}$$

Dosazením (3.59) do (3.60) obdržíme tvar

$$(\mathbf{Q}\mathbf{Q}^\top)\mathbf{p} = \mathbf{h}\mathbf{Q}. \tag{3.61}$$

Zavedením $2N \times 2N$ pseudokorelační matice $\mathbf{\Phi}$ a vektoru $\mathbf{c}$ o délce $2N$

$$\mathbf{\Phi} = \mathbf{Q}\mathbf{Q}^\top, \quad \mathbf{c} = \mathbf{h}\mathbf{Q} \tag{3.62}$$

dostaneme řešení pro parametry filtru

$$\mathbf{p} = \mathbf{\Phi}^{-1}\mathbf{c}. \tag{3.63}$$

Koeficienty čitatele a jmenovatele potom obdržíme jako

$$\mathbf{b} = [p_0, \ldots, p_{M-1}]$$
$$\mathbf{a} = [1, -p_M, \ldots, -p_{2M-1}].$$

4 ÚPRAVA A MODELOVÁNÍ KMITOČTOVÉ CHARAKTERISTIKY ZVUKOVÝCH ZAŘÍZENÍ

Snaha o úpravu kmitočtové charakteristiky zvukových zařízení je předmětem výzkumu už několik desetiletí. Nejčastěji se s úpravou audio řetězce setkáme při snaze zdokonalit systém reproduktor–místnost (nebo pouze samotné zvukové zařízení, pokud vycházíme z měření v bezodrazové komoře), aby výsledná kmitočtová charakteristika byla co nejvyrovnanější. Obecně se jedná o úpravu modulové kmitočtové charakteristiky, méně potom i fázové.

Jelikož samotný systém bývá obvykle vyššího řádu, než je praktické při modelování číslicového filtru, upravuje se pouze zevrubná kmitočtová charakteristika, kteréžto můžeme dosáhnout například použitím zlomkooktávového vyhlazení. Pokud upravujeme kmitočtovou charakteristiku nějakého zvukového systému, který není ve volném poli, spolu s okolním prostředím vytváří systém zvukové pole, které rozhodně není homogenní. Na různých poslechových pozicích v místnosti nalezneme podobnost na nízkých kmitočtech, avšak na vyšších se charakteristika může výrazně lišit i s malou změnou pozice posluchače. Nadměrná úprava kmitočtové charakteristiky pro jeden bod v prostoru obvykle zhorší poslechové podmínky ve všech ostatních místech, proto je vhodné vysoké kmitočty upravovat pouze zevrubně [13].

Vjem výšky zvuku lidským sluchem má logaritmickou závislost, avšak klasické filtry typu FIR a IIR mají kmitočtové rozlišení lineární. U filtrů typu FIR tato vlastnost vychází z jejich podstaty, filtry typu IIR by sice mohly mít větší hustotu pólů na nízkých kmitočtech, ale zkreslení lineární kmitočtové osy oproti logaritmické je v tomto pásmu tak výrazné, že většinou neprodukují uspokojivé výsledky.

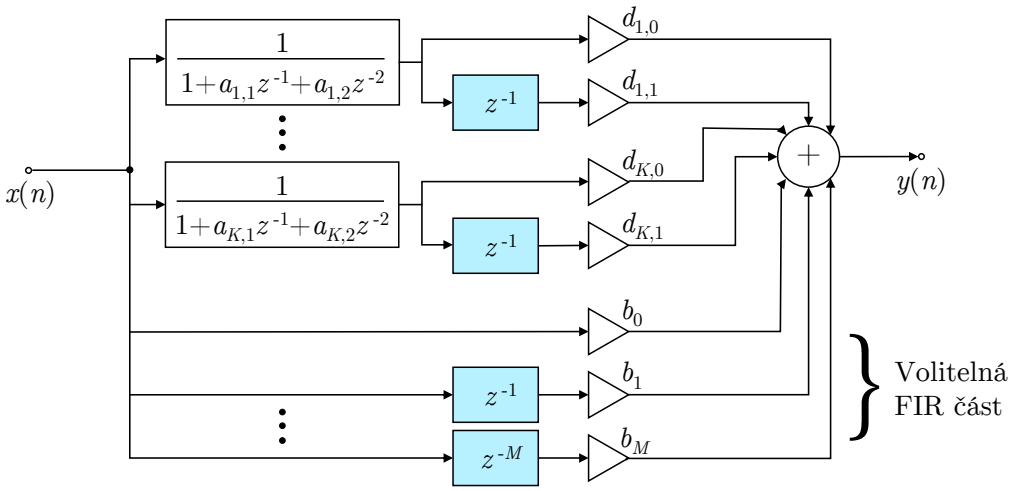
Nejrozšířenější metoda, která uplatňuje vlastnosti lidského sluchu, uplatňuje borcení kmitočtové osy – *warped* filtry (viz kap. 2.3). Ve srovnání s klasickými realizacemi filtrů vykazují kvalitnější výsledky i při nižším řádu navrženého číslicového filtru.

V [13] je popsána další metoda návrhu číslicového filtru, která vychází ze struktury filtru s *paralelními sekcemi druhého řádu.*

4.1 Paralelní filtry

Jedna ze základních realizací číslicových systémů je jeho vnitřní uspořádání do sekcí nízkého řádu. V kapitole 1.2.2 nalezneme popis těchto základních forem. Paralelní kombinací sekcí druhého řádu získáme při implementaci několik výhod, jako je menší citlivost na kvantování koeficientů přenosových funkcí jednotlivých sekcí nebo případná možnost pro paralelizaci kódu. Na obr. 4.1 vidíme strukturu paralelního filtru se sekcemi druhého řádu.

Póly filtru $a_{k,0}, a_{k,1}$ jsou určeny předem (například logaritmickým rozložením, nebo jiným způsobem), čímž zůstávají jako volné parametry pouze koeficienty čitatele přenosové funkce $d_{k,0}, d_{k,1}$ a popřípadě koeficienty simulující FIR část b_m. Tím, že máme volné parametry v čitateli, je možné je nalézt pomocí metody nejmenších čtverců.



Obr. 4.1: Struktura paralelních filtrů druhého řádu [15].

Číslicový filtr může být přímo implementován i v podobě paralelní kombinace komplexních filtrů prvního řádu, avšak praktičtější je ale spojit komplexní póly na společný jmenovatel, a vytvořit tak paralelní kombinace druhého řádu, avšak nyní s reálnými koeficienty. Výpočetní algoritmus je potom efektivnější, když nepočítá s komplexními čísly. Takto přepsaná přenosová funkce je ve tvaru [13]

$$H(z) = \sum_{k=1}^{K} \frac{d_{k,0} + d_{k,1}z^{-1}}{1 + a_{k,1}z^{-1} + a_{k,2}z^{-2}} + \sum_{m=0}^{M} b_m z^{-m}, \tag{4.1}$$

kde K je počet sekcí 2. řádu. Tato přenosová funkce přesně odpovídá struktuře na obr. 4.1.

Víme, že pro podobné vlastnosti kmitočtové charakteristiky filtru je při realizaci filtrem typu FIR řád přenosové funkce i několikanásobně vyšší, než při realizaci filtrem typu IIR. Obzvláště pak v oblasti nízkých kmitočtů, které u zvukových zařízení nejčastěji upravujeme, kde řád filtru FIR může dosahovat stovek až tisíců [9]. Každá elementární sekce paralelního filtru je ovšem jen 2. řádu a realizace poté vyžaduje pouze 2 zpožďovací bloky, při jakémkoliv zvoleném kmitočtu. Oproti použití filtru typu FIR se tak několikanásobně zmenší procesní zpoždění při zpracování vstupního signálu.

4.2 Řešení v časové oblasti

Jak již bylo řečeno, póly přenosových funkcí paralelního filtru musí být určeny předem, aby výsledné normální rovnice při řešení metodou nejmenších čtverců byly lineární. Rozmístění pólů pro zpracování audiosignálu je vhodné logaritmické a dá se odvodit z rovnic

$$\begin{aligned} \vartheta_k &= \frac{2\pi f_k}{f_s} \\ p_k &= R^{\vartheta_k/\pi} \mathrm{e}^{\pm \mathrm{j}\vartheta_k}, \end{aligned} \tag{4.2}$$

kde ϑ_k jsou kmitočty pólů v radiánech vypočítané ze setu logaritmicky rozmístěných kmitočtů f_k. Moduly pólů jsou exponenciálně utlumeny, čímž se mají přiblížit rozlišení s konstantní šířkou pásma [14]. Logaritmické rozmístění pólů není podmínkou, pomocí *warped* filtrů lze docílit většího zaměření na určitou část spektra nebo lze navrhnout ideální rozmístění pólů pro danou specifikaci některou z metod popsaných v kapitole 3.

4.2.1 Návrh filtru

Impulsovou odezvu paralelních filtrů lze zapsat jako [13]

$$h(n) = \sum_{k=1}^{K} d_{k,0}u_k(n) + d_{k,1}u_k(n-1) + \sum_{m=0}^{M} b_m\delta(n-m), \tag{4.3}$$

kde $u_k(n)$ je impulsová odezva funkce jednotlivé sekce s přenosovou funkcí

$$U_k(z) = \frac{1}{1 + a_{k,1}z^{-1} + a_{k,2}z^{-2}}, \tag{4.4}$$

což je exponenciálně utlumená harmonická funkce, a $\delta(n)$ je jednotkový impuls.

Maticovým přepisem rovnice (4.3) obdržíme

$$\mathbf{h} = \mathbf{M}\mathbf{p}, \tag{4.5}$$

kde $\mathbf{p} = [d_{1,0}, d_{1,1}, \dots d_{K,0}, d_{K,1}, b_0, \dots b_M]^\top$ je sloupcový vektor volných parametrů. Řádky modelové matice $\mathbf{M}$ obsahují jednotlivé modelové signály $u_k(n)$ a jejich zpožděné varianty $u_k(n-1)$. Poslední řádky matice $\mathbf{M}$ obsahují jednotkový impuls $\delta(n)$ a jeho zpožděné varianty až do M-tého řádu FIR části paralelního modelu a $\mathbf{h} = [h(0) \dots h(N)]^\top$ je sloupcový vektor výsledné impulsové odezvy.

Úkolem je vypočítat takové ideální parametry $\mathbf{p}_{\text{opt}}$, aby výsledek $\mathbf{M}\mathbf{p}_{\text{opt}}$ byl co nejblíže cílové impulsové odezvě $\mathbf{h}_{\text{d}}$. Po zvolení vhodného kritéria optimality, např. *chyba návrhu filtru* $E_N = \sum_{n=0}^{N} [h_{\text{d}}(n) - h(n)]^2$, lze optimální parametry najít metodou nejmenších čtverců [13]

$$\mathbf{p}_{\text{opt}} = (\mathbf{M}^{\text{H}}\mathbf{M})^{-1}\mathbf{M}^{\text{H}}\mathbf{h}_{\text{d}}, \tag{4.6}$$

kde $\mathbf{M}^{\text{H}}$ je hermitovská transpozice matice $\mathbf{M}$.

4.2.2 Návrh kompenzačního filtru

Pro korekci kmitočtové charakteristiky (nejčastěji na vyrovnané spektrum) lze využít číslicový filtr, jehož kmitočtová charakteristika je inverzní ke kmitočtové charakteristice zařízení. S pomocí paralelního filtru je však možné tento problém vyřešit bez inverze stávající kmitočtové charakteristiky a přímo v časové oblasti [13].

Požadujeme, aby výsledná impulsová odezva kompenzačního filtru $h(n)$, která vznikne jako konvoluce odezvy filtru $h_{\text{eq}}(n)$ a reproduktoru (systému) $h_{\text{s}}(n)$, byla co nejvíce podobná požadované impulsové odezvě $h_{\text{d}}(n)$ (např. jednotkový impuls). V tomto případě je

vstupem do paralelních filtrů místo jednotkového impulsu (jako je tomu v rovnici (4.3)) impulsová odezva systému $h_\mathrm{s}(n)$. Výpočet $h(n)$ je popsán rovnicí [13]

$$h(n) = h_\mathrm{eq}(n) * h_\mathrm{s}(n) = \sum_{k=1}^{K} d_{k,0}u_k(n) * h_\mathrm{s}(n) + d_{k,1}u_k(n-1) * h_\mathrm{s}(n)$$
$$+ \sum_{m=0}^{M} b_m\delta(n-m) * h_\mathrm{s}(n) = \sum_{k=1}^{K} d_{k,0}s_k(n) + d_{k,1}s_k(n-1) + \sum_{m=0}^{M} b_m h_\mathrm{s}(n), \tag{4.7}$$

kde signál $s_k(n) = u_k(n) * h_\mathrm{s}(n)$ je odezva systému $h_\mathrm{s}(n)$ filtrovaná jednou sekcí pólů. S použitím $\mathcal{Z}$-transformace můžeme signál $s_k(n)$ zapsat jako

$$S_k(z) = \frac{H_\mathrm{s}(z)}{1 + a_{k,1}z^{-1} + a_{k,2}z^{-2}}. \tag{4.8}$$

Je vidět, že rovnice (4.7) má stejný tvar jako rovnice (4.3), takže parametry $d_{k,0}, d_{k,1}$ a b_m se vypočítají stejným způsobem. Maticový zápis rovnice (4.7) je

$$\mathbf{h} = \mathbf{M}_\mathrm{eq}\mathbf{p}, \tag{4.9}$$

kde řádky nové modelové matice $\mathbf{M}_\mathrm{eq}$ obsahují signály $s_k(n), s_k(n-1)$ a poslední řádky odezvu systému $h_\mathrm{s}(n)$ a její zpožděné formy do M-tého řádu. Poté optimální parametry nalezneme pomocí obdobné rovnice

$$\mathbf{p}_\mathrm{opt} = (\mathbf{M}_\mathrm{eq}^\mathrm{H}\mathbf{M}_\mathrm{eq})^{-1}\mathbf{M}_\mathrm{eq}^\mathrm{H}\mathbf{h}_\mathrm{t}. \tag{4.10}$$

4.3 Řešení v kmitočtové oblasti

Opět zvolíme logaritmické rozmístění pólů

$$\theta_k = \frac{2\pi f_k}{f_s} \tag{4.11}$$
$$p_k = \mathrm{e}^{-\frac{\Delta\theta_k}{2}}\mathrm{e}^{\pm\mathrm{j}\theta_k}, \tag{4.12}$$

kde θ_k jsou kmitočty pólů v radiánech vypočítané z logaritmicky rozmístěných absolutních kmitočtů f_k a vzorkovacího kmitočtu f_s. Šířka pásma k-tého filtru je $\Delta\theta_k$ a vypočítá se ze sousedních kmitočtů jednotlivých pólů [15]:

$$\begin{aligned} \Delta\theta_k &= \frac{\theta_{k+1} - \theta_{k-1}}{2} && \text{pro } k = [2, \ldots, K-1] \\ \Delta\theta_1 &= \theta_2 - \theta_1 \\ \Delta\theta_K &= \theta_K - \theta_{K-1}. \end{aligned} \tag{4.13}$$

Tímto způsobem se budou přenosové funkce jednotlivých filtrů překrývat zhruba na úrovni $-3\,\mathrm{dB}$.

4.3.1 Návrh filtru

S předurčenými póly je problém opět lineární a řeší se metodou nejmenších čtverců. Přepisem rovnice (4.1) na maticový tvar pro konečnou délku úhlových kmitočtů ϑ_n dostaneme [15]

$$\mathbf{h} = \mathbf{M}\mathbf{p}, \tag{4.14}$$

kde $\mathbf{p} = [d_{1,0}, d_{1,1}, \dots d_{K,0}, d_{K,1}, b_0, \dots b_M]^\top$ je ten samý sloupcový vektor volných parametrů. Řádky modelové matice $\mathbf{M}$ nyní obsahují jednotlivé vyčíslené přenosové funkce sekcí druhého řádu pro kmitočty ϑ_n

$$\frac{1}{1 + a_{k,1}\mathrm{e}^{-\mathrm{j}\vartheta_n} + a_{k,2}\mathrm{e}^{-\mathrm{j}2\vartheta_n}}$$

a jejich zpožděné varianty

$$\frac{\mathrm{e}^{-\mathrm{j}\vartheta_n}}{1 + a_{k,1}\mathrm{e}^{-\mathrm{j}\vartheta_n} + a_{k,2}\mathrm{e}^{-\mathrm{j}2\vartheta_n}}.$$

Poslední řádky matice $\mathbf{M}$ jsou přenosové funkce FIR části modelu, $\mathrm{e}^{-\mathrm{j}m\vartheta_n}$ pro $m = [0 \dots M]$.

Matice $\mathbf{h}$ je sloupcový vektor obsahující vypočítanou kmitočtovou odezvu paralelní kombinace filtrů $[H(\vartheta_0) \dots H(\vartheta_n)]^\top$.

Nyní je cílem najít optimální parametry $\mathbf{p}_{\mathrm{opt}}$, aby $\mathbf{h} = \mathbf{M}\mathbf{p}_{\mathrm{opt}}$ byla co nejblíže cílové kmitočtové odezvě $\mathbf{h}_{\mathrm{d}} = [H_{\mathrm{d}}(\vartheta_0) \dots H_{\mathrm{d}}(\vartheta_n)]^\top$ [15]

$$\mathbf{p}_{\mathrm{opt}} = (\mathbf{M}^{\mathrm{H}}\mathbf{M})^{-1}\mathbf{M}^{\mathrm{H}}\mathbf{h}_{\mathrm{d}}, \tag{4.15}$$

kde $\mathbf{M}^{\mathrm{H}}$ je Hermitovská transpozice matice $\mathbf{M}$.

Rovnice (4.15) uvažuje specifikaci cílové kmitočtové odezvy $H_{\mathrm{d}}(\vartheta_n)$ na oboustranném spektru, tedy v rozsahu $\vartheta_n \in \langle -\pi, \pi \rangle$. Nalezené optimální parametry mohou tedy být komplexní, protože zadání cílové frekvenční odezvy není omezené na symetrické spektrum.

Avšak pro většinu případů budeme chtít pouze reálnou odezvu a tím pádem i reálné koeficienty. V případě jednostranné specifikace na intervalu $\vartheta_n \in \langle 0, \pi \rangle$ je třeba použít místo rovnice (4.15) [15]

$$\mathbf{p}_{\mathrm{opt}} = \left(\mathrm{Re}\left\{\mathbf{M}^{\mathrm{H}}\mathbf{M}\right\}\right)^{-1} \mathrm{Re}\left\{\mathbf{M}^{\mathrm{H}}\mathbf{h}_{\mathrm{d}}\right\}. \tag{4.16}$$

4.3.2 Návrh kompenzačního filtru

Podobným způsobem jako v časové oblasti je možnost kompenzovat kmitočtovou charakteristiku systému, aby vyhovovala zadání (vyrovnané spektrum). Při konstrukci inverzní kmitočtové charakteristiky podílem $H_{\mathrm{d}}(\vartheta_n)/H_{\mathrm{s}}(\vartheta_n)$ mohou vzniknout problémy, když změřená kmitočtová charakteristika systému, bude mít poměrně výrazná úzká minima(obr. 4.2), která po vydělení vytvoří ostré výběžky na cílové charakteristice. To může ovlivnit přesnost návrhu filtru a vůbec kvalitu následné reprodukce.

Vhodnější řešení je analogické k návrhu kompenzačního filtru v časové oblasti návrhem filtru [15]

$$H(z) = \sum_{k=1}^{K} \frac{d_{k,0} + d_{k,1} z^{-1}}{1 + a_{k,1} z^{-1} + a_{k,2} z^{-2}} H_{\mathrm{s}}(z) + \sum_{m=1}^{M} b_m z^{-m} H_{\mathrm{s}}(z). \tag{4.17}$$

Maticovým zápisem této rovnice dostaneme

$$\mathbf{h} = \mathbf{M}_{\mathrm{eq}} \mathbf{p}_{\mathrm{eq}}, \tag{4.18}$$

kde $\mathbf{p}_{\mathrm{eq}} = [d_{1,0}, d_{1,1}, \dots d_{K,0}, d_{K,1}, b_0, \dots b_M]^{\top}$ je sloupcový vektor obsahující volné parametry systému paralelních filtrů. Řádky modelové matice $\mathbf{M}_{\mathrm{eq}}$ vzniknou vynásobením modelové matice $\mathbf{M}$ z rovnice (4.14) s odezvou systému $H_{\mathrm{s}}(\vartheta_n)$, tedy místo

$$\frac{1}{1 + a_{k,1} \mathrm{e}^{-\mathrm{j}\vartheta_n} + a_{k,2} \mathrm{e}^{-\mathrm{j}2\vartheta_n}}$$

bude použita přenosová funkce

$$\frac{H_{\mathrm{s}}(\vartheta_n)}{1 + a_{k,1} \mathrm{e}^{-\mathrm{j}\vartheta_n} + a_{k,2} \mathrm{e}^{-\mathrm{j}2\vartheta_n}}.$$

Výpočet parametrů proběhne stejným způsobem podle rovnice (4.15) nebo (4.16).

4.3.3 Kmitočtově závislé váhování

Další výhodou návrhu filtru v kmitočtové oblasti je možnost zvolit různé váhovací koeficienty na různých kmitočtech. Potom se výpočet parametrů změní na [15]

$$\mathbf{p}_{\mathrm{opt}} = (\mathbf{M}^{\mathrm{H}} \mathbf{W} \mathbf{M})^{-1} \mathbf{M}^{\mathrm{H}} \mathbf{W} \mathbf{h}_{\mathrm{d}}, \tag{4.19}$$

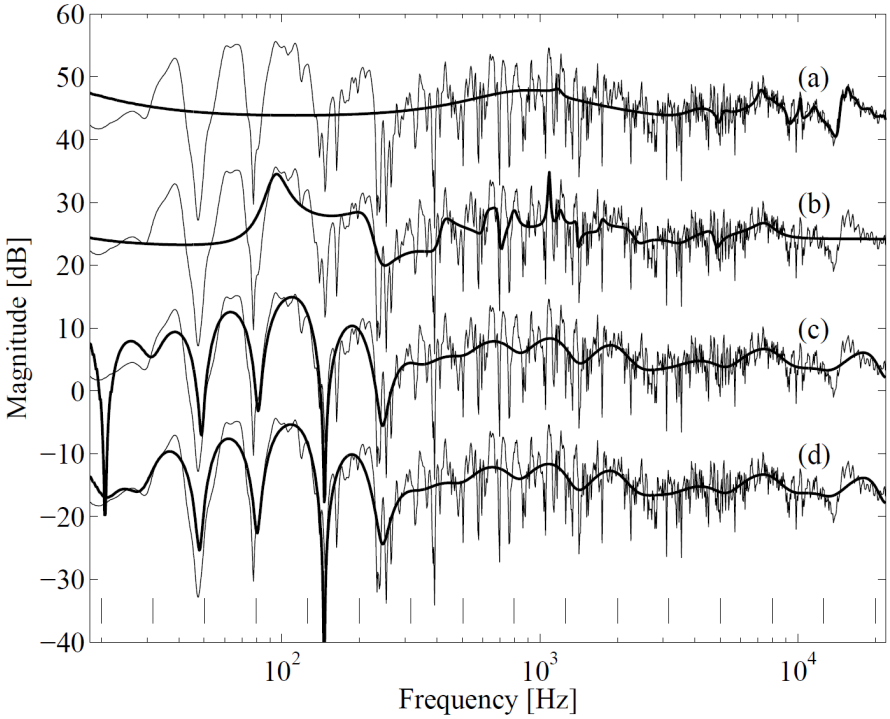
kde $\mathbf{W}$ je diagonální matice vytvořená z váhovacích koeficientů $W(\vartheta_n)$, a nebo v případě jednostranné specifikace $\vartheta_n \in \langle 0, \pi \rangle$

$$\mathbf{p}_{\mathrm{opt}} = \left(\mathrm{Re}\left\{ \mathbf{M}^{\mathrm{H}} \mathbf{W} \mathbf{M} \right\} \right)^{-1} \mathrm{Re}\left\{ \mathbf{M}^{\mathrm{H}} \mathbf{W} \mathbf{h}_{\mathrm{d}} \right\}, \tag{4.20}$$

Možný příklad použití: pokud cílová kmitočtová charakteristika vznikla průměrem několika změřených charakteristik (např. ve více poslechových bodech v místnosti), z průběhů se vypočítá rozptylová funkce σ_n^2 a dle ní se může přidělit méně spolehlivým bodům specifikace menší váha; například stanovením $W(\vartheta_n) = 1/\sigma_n^2$.

4.4 Srovnání návrhu filtru v časové a kmitočtové oblasti

V kapitolách 4.2 a 4.3 byl popsán návrh filtru v časové a kmitočtové oblasti. Pro dosažení stejných výsledků při návrhu v kmitočtové oblasti jako při návrhu v časové oblasti, musí být úhlové kmitočty ϑ_n, rozmístěny lineárně a dostatečně hustě [15]. Avšak pokud jsou kmitočty ϑ_n rozmístěny logaritmicky, je dosaženo lehce odlišných výsledků, viz obr. 4.2 (c) a (d).



Obr. 4.2: Porovnání různých návrhů filtrů. (a) FIR filtr 32. řádu, (b) *warped* IIR filtr 32. řádu, (c) sekce paralelních filtrů 32. řádu navržená v časové oblasti a (d) v kmitočtové oblasti [15].

Návrh vychází z impulsové odezvy, kmitočtová charakteristika je potom po Fourierově transformaci definována v lineárním rozložení. Byl proveden přepočet na 128 logaritmicky rozložených bodů, ze kterých vycházel návrh v kmitočtové oblasti. Je vidět zlepšení kmitočtové charakteristiky návrhu na nízkých kmitočtech, protože se chybová funkce počítá vzhledem k logaritmickému rozložení samotného zobrazení [15].

Navíc, pokud je požadovaná specifikace filtru $H_\mathrm{d}(\vartheta_n)$ popsaná v kmitočtové oblasti, jak tomu často bývá, není impulsová odezva k návrhu vůbec potřeba. To odstraní výpočetní nepřesnosti při použití Fourierovy transformace a případnou interpolaci kmitočtové charakteristiky.

V případě definice kmitočtové charakteristiky na logaritmicky rozmístěných kmitočtech vyžaduje návrh filtru méně definovaných bodů – kratší vektor $\mathbf{h}_\mathrm{d}$ a matice $\mathbf{M}$ v porovnání s návrhem v časové oblasti, čímž se sníží výpočetní nároky. Abychom dosáhli v časové oblasti na dostatečnou přesnost na nízkých kmitočtech, je třeba použít dlouhou modelovou impulsovou odezvu. Pro případ vyobrazený na obr. 4.2 bylo pro návrh (d) v kmitočtové oblasti potřeba 80krát méně zadaných bodů než v časové (c), což je už velmi výrazný rozdíl. Takto snížená výpočetní náročnost může být výhodou při zpracování signálů v reálném čase [15].

5 NÁVRH PARALELNÍHO FILTRU

Modelováním kmitočtové charakteristiky nebo návrhem kompenzačního filtru zvukových zařízení se v několika jeho publikacích zabývá Balázs Bank [13, 15, 16, 17]. Na svých stránkách [16] poskytuje vzorové skripty pro prostředí MATLAB, které se zabývají návrhem paralelních filtrů především pro modelování nebo kompenzace kmitočtové charakteristiky reproduktoru.

Zaměříme se na návrh tohoto filtru v kmitočtové oblasti. Výchozím zadáním bude změřená modulová kmitočtová charakteristika daného reproduktoru (záleží na postupu měření, jestli bude k dispozici modulová i fázová charakteristika, předpokládejme však, že fázová kmitočtová charakteristika není vždy k dispozici).

5.1 Postup při návrhu filtru

V kapitole 4.1 bylo řečeno, že návrh paralelního filtru vyžaduje předem určit pozice pólů. Existuje více způsobů, jak zvolit ideální rozmístění pólů vzhledem k vlastnostem lidského sluchu nebo na míru dané požadované kmitočtové odezvě.

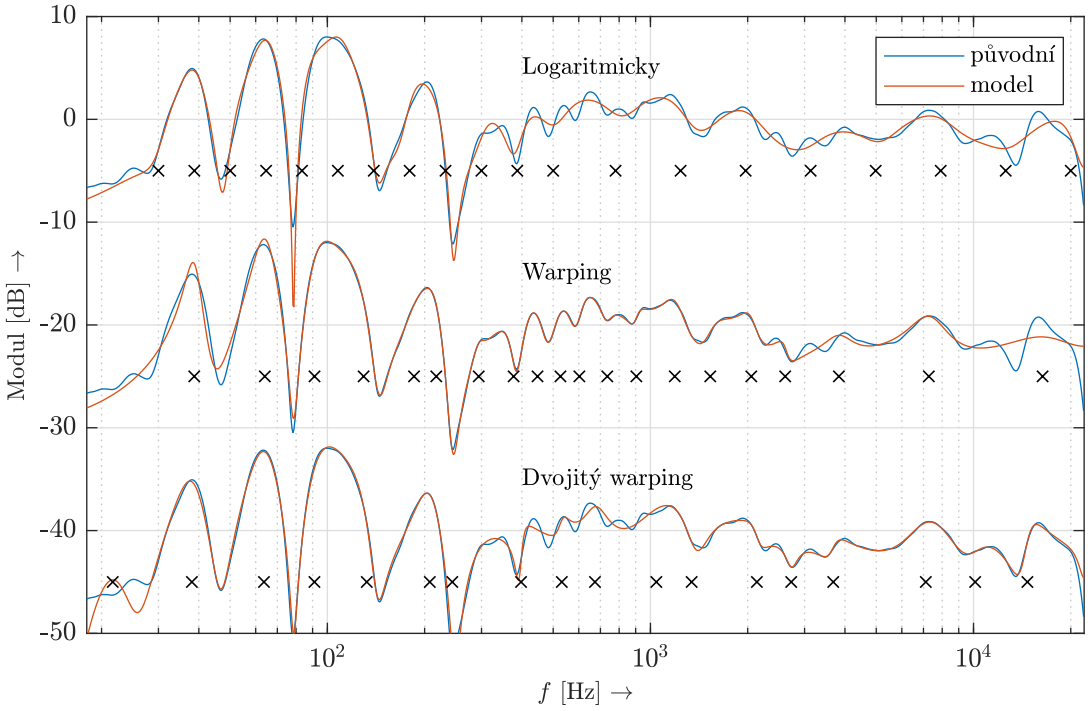
Lineární rozmístění pólů na kmitočtové ose je nevhodné, docílilo by se nadměrné korekce vysokých kmitočtů, která je většinou nežádoucí. Logaritmické rozmístění pólů má větší smysl, v logaritmickém zobrazení kmitočtové odezvy potom vidíme póly ve stejné vzdálenosti od sebe.

Nabízí se také použít některou z metod návrhu AR kmitočtového filtru popsaných v kapitole 3. Pronyho nebo Steiglitz-McBride metoda a jejich modifikace v kmitočtové rovině [18] jsou schopné určit vhodné kmitočty pólů přenosové funkce, avšak při klasickém použití vykazují stejný nedostatek – lineární rozložení na kmitočtové ose. Je však možné filtr navrhnout ve zborceném kmitočtovém rozlišení, podobně jako *warped* filtry (kap. 2.3). Úhlové kmitočty ω_k, na kterých je definovaná zadaná kmitočtová charakteristika, se transformují podle vzorce [19]

$$\omega_{\mathrm{wp}k} = \arctan\left(\frac{(1-\lambda^2)\sin(\omega_k)}{(1+\lambda^2)\cos(\omega_k) - 2\lambda}\right), \tag{5.1}$$

a návrh filtru proběhne např. zmíněnou Steiglitz-McBride metodou.

Pro kvalitnější poslechové podmínky je vhodné kmitočtovou charakteristiku reproduktorů upravit spíše v pásmu nižších a středních kmitočtů a vyšší kmitočty korigovat méně. Pro oddělení přístupu k těmto dvěma sekcím kmitočtové charakteristiky se osvědčil přístup rozdělit kmitočtovou charakteristiku na pásma, ke kterým se bude přistupovat jinak. Nejjednodušším způsobem je při logaritmickém rozložení pólů zvolit jinou hustotu pólů v těchto pásmech. Je možné i použít vícenásobně zborcenou kmitočtovou osu a k návrhu filtru přistupovat v různým oblastem odděleně (postup není tak přímočarý, jak se zdá, podrobný popis lze nalézt v kap. 6 nebo v [17]). Výsledky dosažené použitím tzv. *dvojitého warpingu* jsou velmi uspokojivé. Porovnání jednotlivých metod lze nalézt v [9] nebo na obr. 5.1 a v kap. 6.5.



Obr. 5.1: Modelování kmitočtové charakteristiky reproduktoru v místnosti paralelním filtrem 40. řádu (tedy 20 sekcí) různými způsoby. Je zobrazena kmitočtová charakteristika dvoupásmového reproduktoru postaveného na zemi v obyčejném obývacím pokoji změřená ve vzdálenosti 2 m, která byla vyhlazena šestinooktávovým filtrem. Příklad byl zvolen kvůli výraznému zvlnění modulové kmitočtové charakteristiky na nízkých kmitočtech způsobeným vlivem místnosti [13].

Návrh filtru

Algoritmus návrhu paralelního filtru pro zadanou kmitočtovou charakteristiku vyžaduje specifikaci několika parametrů. Všechny přístupy mají jeden parametr společný a to je **počet pólů**. Při rozdělení kmitočtové charakteristiky do dvou částí a zpracování zvlášť jsou poté parametry:

- **Počet pólů** ve spodní části spektra.
- **Počet pólů** ve vrchní části spektra.
- **Dělicí kmitočet**.
- Případně **délka překryvu** dvou pásem.

Při použití *dvojitého warpingu* poté ještě:

- Parametr $\boldsymbol{\lambda}$ pro spodní část spektra.
- Parametr $\boldsymbol{\lambda}$ pro vrchní část spektra.

Postup návrhu je popsán v následujících krocích:

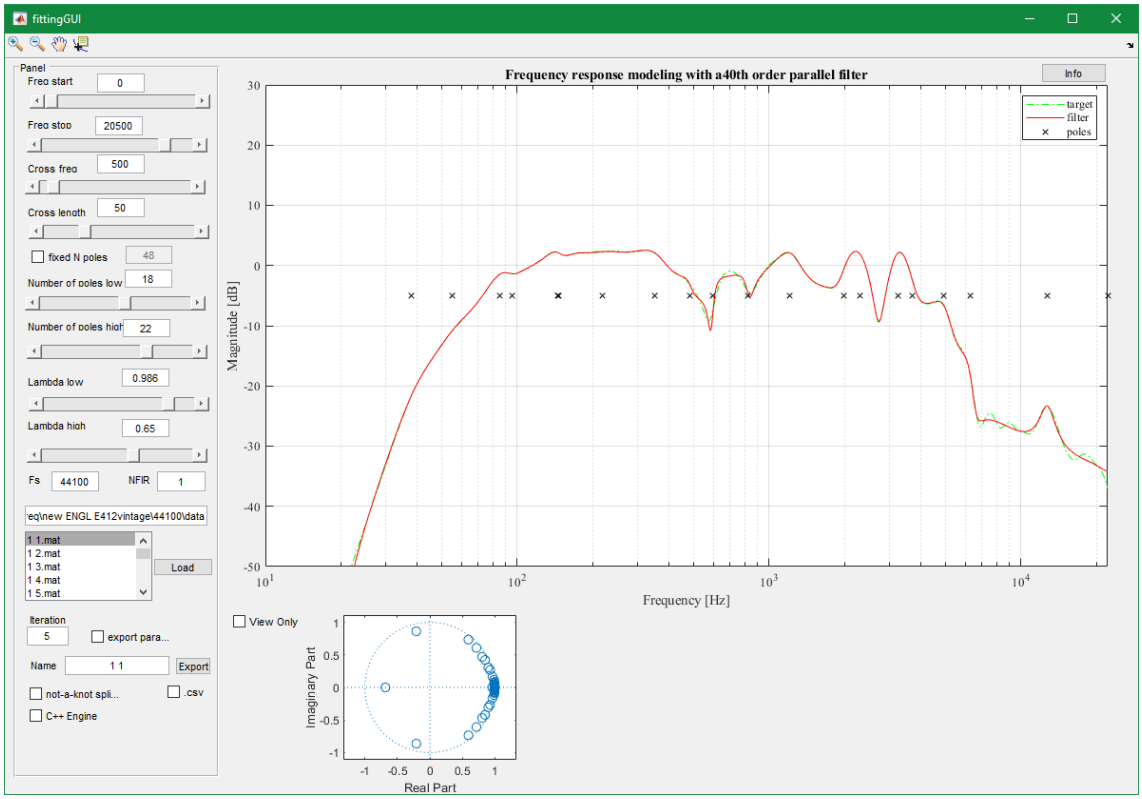
1. Vstupem je cílová modulová kmitočtová charakteristika $H_\mathrm{d}(\vartheta_n)$ definovaná na úhlových kmitočtech ϑ_n. Uměle se vytvoří její fázová kmitočtová charakteristika tak, aby

odpovídala systému s minimální fází – Hilbertovou transformací [15, 24]. Je možné, a i vhodné, změřenou kmitočtovou charakteristiku zlomkooktávově vyhladit (např. šestinooktávově).

2. Určí se póly paralelních filtrů některou z popsaných metod.
3. Proběhne výpočet nulových bodů paralelních sekcí filtru způsobem popsaným v 4.3.1.

Metoda disponující největší flexibilitou a, jak bude v kap. 6.5 ukázáno, dosahující nejpřesnějších výsledků při návrhu filtru je metoda využívající *dvojitý warping* kmitočtové osy. Vstupem algoritmu je mimo kmitočtovou charakteristiku i všech šest dříve zmíněných parametrů.

Pro testovací účely jsem navrhl v prostředí MATLAB jednoduchou aplikaci *Fitting GUI* za použití prostředí GUIDE.



Obr. 5.2: Grafické uživatelské rozhraní aplikace *Fitting GUI* (MATLAB)

Mimo již popsané parametry je také možnost nastavit mezní spodní a vrchní kmitočet optimalizace – omezí se rozsah zvolené kmitočtové charakteristiky, pro který má být návrh co nejpřesnější.

Aplikace je dobrým pomocníkem pro prozkoumání vlivu jednotlivých parametrů na výsledný návrh filtru. Cílem je navrhnout algoritmus, který by automaticky optimalizoval tyto parametry tak, aby byl návrh co nejpřesnější.

Problém sestává z optimalizace několika parametrů, jejichž počet můžeme ovlivňovat případnými zjednodušeními – zvolíme pevně celkový počet pólů filtru a optimalizujeme

pouze poměr počtu ve spodním a vrchním pásmu, zvolíme pevně počet pólů v obou pásmech, zvolíme pevně dělicí kmitočet apod.

Vliv jednotlivých parametrů na výsledný návrh lze jen těžko analyticky odhadnout, tak je třeba pro optimalizaci zvolit vhodnou numerickou metodu.

5.2 Optimalizace hejnem částic

Pro řešení tohoto problému byl zvolen algoritmus *optimalizace hejnem částic – Particle Swarm Optimization* (dále jen PSO). Jde o jeden z metaheuristických algoritmů, který byl inspirován přírodními systémy, konkrétně hejny ptáků či ryb a spadá do skupiny algoritmů souhrnně nazývaných *inteligence hejna*. Poprvé byl popsán v roce 1995 Eberhartem a Kennedym [20]. Tento algoritmus je založen na prohledávání N-dimenzionálního prostoru hejny částic, které hledají globální optimální řešení určitého problému.

Ke každé částici se přistupuje jako k nehmotnému bodu v N-rozměrném prostoru, který má svoji polohu a rychlost, které se v průběhu výpočtu dynamicky mění. Rychlost částice je přepočítávána na základě znalosti její doposud nejlepší dosažené pozice nebo na základě zkušenosti ostatních členů hejna. Algoritmus lze uplatnit na problémy se spojitými i nespojitými proměnnými, problémy s omezujícími podmínkami, multimodální problémy a další [20, 21].

Kvůli velké variabilitě a zároveň malému množství vlastních parametrů PSO byl zvolen tento algoritmus pro optimalizaci parametrů pro návrh filtru.

5.2.1 Popis algoritmu

Od původní verze algoritmu z roku 1995 vzniklo několik standardů PSO (SPSO 2006, SPSO 2011, ...) jejichž vnitřní principy se v některých krocích liší [22]. Všechny ale sdílejí model částice, který je tvořen:

- Momentální pozicí v prohledávaném prostoru.
- Hodnotou účelové funkce na této pozici.
- Rychlostí.
- Znalostí poslední nejlepší dosažené pozice touto částicí (*pbest*).
- Znalostí poslední nejlepší dosažené pozice jejím okolím (*gbest*).

Dohromady částice tvoří tzv. *hejno*, které prohledává určený prostor. Uvnitř hejna existují vztahy mezi částicemi. Aby mohl algoritmus správně fungovat, musí si částice vyměňovat informace o svých doposud nejlepších pozicích a hodnotách účelové funkce[1] na těchto pozicích. Podle zvolené *topologie* se volí např. komunikace se svým nejbližším okolím, nebo i se všemi částicemi zároveň.

[1] Pro testování algoritmů se používají tzv. účelové funkce, např. *Ackley, Rastrigin, Rosenbrock, Sphere* aj. Při konkrétní aplikaci bude použita funkce specifická k danému problému. Souhrnně se však používá označení *účelová funkce*, *fitness* nebo *cost* funkce [21, 23].

Základní rovnice algoritmu PSO pro výpočet nové polohy částice jsou [20]

$$v_{ij}(t+1) = wv_{ij}(t) + c_1 \cdot \text{rand}()(pbest_{ij}(t) - x_{ij}(t)) + c_2 \cdot \text{rand}()(gbest_j(t) - x_{ij}(t)), \tag{5.2}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1). \tag{5.3}$$

První rovnice aktualizuje rychlost v nové iteraci, druhá rovnice novou rychlostí aktualizuje polohu částice. Index i symbolizuje pořadí částice a index j pořadí dimenze. Potom x_i je poloha částice, v_i je rychlost částice, $pbest_i$ je nejlepší výsledek i-té částice a *gbest* je celkový nejlepší výsledek. Člen rand() představuje vektor náhodných čísel s rovnoměrným rozdělením v intervalu $\langle 0, 1\rangle$ se stejnou délkou, jako je počet dimenzí. V rovnici také vystupují koeficienty w, c_1 a c_2, pro jejichž hodnoty existují různá doporučení [23]. Ovlivňují rychlost konvergence ke globálním a lokálním minimům.

Algoritmus použitý v této práci odpovídá standardu SPSO 2006 [22] a skládá se z těchto kroků:

Inicializace:

1. Každé částici se přiřadí náhodná pozice uvnitř prohledávaného prostoru.
2. Každé částici se přiřadí náhodná rychlost.
3. Spočítá se hodnota fitness.
4. Každá částice aktualizuje svoji hodnotu *pbest* dosavadní pozicí.
5. Do proměnné *gbest* se uloží pozice částice s nejlepší fitness.

Iterace:

1. Pro každou částici se vypočítá nová rychlost.
2. Pro každou částici se vypočítá nová pozice na základě nové rychlosti.
3. **Pokud** částice vyletěla z prohledávaného prostoru je třeba zvolit vhodné ošetření.
4. Spočítá se hodnota fitness.
5. **Pokud** je nová fitness lepší než *pbest*:
   (a) Aktualizuje se *pbest* dané částice.
   (b) **Pokud** je *pbest* dané částice lepší než *gbest*, aktualizuje se *gbest* pozicí této částice.
6. Kroky 1. až 5. se opakují dokud není splněno kritérium pro ukončení výpočtu.

Je třeba říci, že proměnné *pbest* a *gbest* obsahují jak informace o pozici částice, tak i hodnotu fitness. Fitness funkcí se určuje „kvalita“, nebo také „cena“ částice na dané pozici. Pro náš problém při návrhu paralelního filtru je touto kvalitou částice jednočíselná hodnota přesnosti návrhu filtru. Jako fitness funkci lze zvolit jakoukoliv z možností popsaných v následující kapitole.

5.3 Vyhodnocení přesnosti návrhu filtru

Pro výpočet jednočíselného kritéria hodnotící přesnost návrhu budou porovnány kmitočtové charakteristiky zadání a výsledného filtru. Je možné charakteristiky porovnávat pro každý bod jejich specifikace (za předpokladu, že úhlové kmitočty ϑ_n budou shodné), nebo počet bodů snížit např. vypočítáním průměrné hodnoty charakteristiky v různých kmitočtových pásmech.

5.3.1 Výpočet bod po bodu

MSE celé kmitočtové charakteristiky

Ze dvou průběhů kmitočtových charakteristik v dB se vypočítá střední kvadratická odchylka – MSE (MATLAB funkce `immse()`) dle rovnice

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2. \tag{5.4}$$

Pearsonův korelační koeficient

Korelační koeficient ρ vyjadřuje podobnost jednotlivých průběhů. Jeho definice je

$$\rho_{x,y} = \frac{\mathrm{cov}(x,y)}{\sigma_x \sigma_y}, \tag{5.5}$$

kde $\mathrm{cov}(x,y)$ je kovariance a σ_x, σ_y jsou směrodatné odchylky jednotlivých průběhů. V prostředí MATLAB funkce `corrcoef()`.

Spectrum deviation

Dokument [27] pojednává o metodách určování vad transformátorů z jejich kmitočtových charakteristik a definuje estimátor „spectrum deviation“.

$$\sigma = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\left(\frac{x_i - \frac{x_i+y_i}{2}}{\frac{x_i+y_i}{2}}\right)^2 + \left(\frac{y_i - \frac{x_i+y_i}{2}}{\frac{x_i+y_i}{2}}\right)^2} \tag{5.6}$$

Popsaný nedostatek tohoto výpočtu je, že při hodnotách charakteristiky blízkým nule podává falešné negativní výsledky – i malé fluktuace v charakteristikách je klasifikována jako větší chyba. Pokud jsou absolutní hodnoty charakteristik větší, tak jsou výsledky relevantnější. Přesto při testech níže poskytuje poměrně dobrou informaci o přesnosti návrhu filtru (viz tab. 6.2, 6.3 a 6.4).

5.3.2 Výpočet v kmitočtových pásmech

MSE ve zlomkooktávových pásmech

Z kmitočtových charakteristik se vypočítá průměrná hodnota ve zlomkooktávových pásmech (oktávové a třetinooktávové) a z těchto hodnot se spočítá MSE. Pro výpočet zlomkooktávových pásem slouží rovnice [25]

$$f_{\mathrm{c}} = 2^{\frac{x}{n}} f_{\mathrm{ref}} \tag{5.7}$$

$$f_{\mathrm{l}} = 2^{-\frac{1}{2n}} f_c \tag{5.8}$$

$$f_{\mathrm{u}} = 2^{\frac{1}{2n}} f_c, \tag{5.9}$$

kde f_{ref} je referenční kmitočet 1 kHz, f_c je střední kmitočet pásma, $x \in \mathbb{Z}$ je relativní pořadí pásma k pásmu s referenčním kmitočtem a n je počet pásem analýzy v jedné oktávě. Potom f_{l} a f_{u} je spodní, resp. horní mezní kmitočet pásma.

MSE v kritických pásmech

Ve své publikaci [26] Zwicker uvádí rovnici pro přepočet kmitočtu v Hz na barkové číslo.

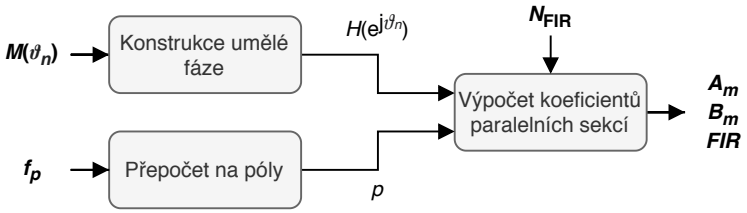
$$z(f) = 13 \cdot \arctan\left(\frac{0{,}76 \cdot f}{1000}\right) + 3{,}5 \cdot \arctan\left(\left(\frac{f}{7500}\right)^2\right) \tag{5.10}$$

Poskytnutý vektor kmitočtů se přepočítá do barkové škály a v rámci jednotlivých pásem bude vypočítána střední hodnota kmitočtové charakteristiky. Z těchto hodnot se potom vypočítá hodnota MSE.

6 IMPLEMENTACE ALGORITMŮ PRO NÁVRH PARALELNÍHO FILTRU

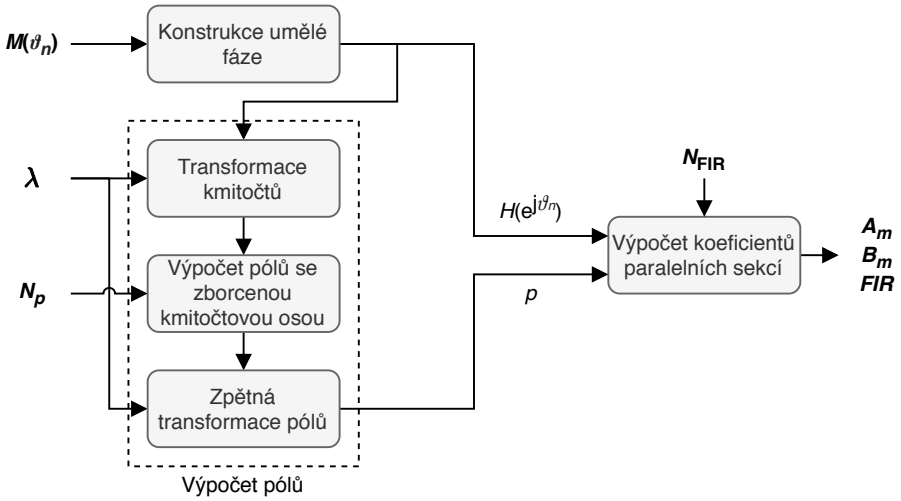
Na obr. 6.1, 6.2 a 6.3 jsou bloková schémata algoritmů pro výpočet koeficientů paralelních sekcí filtru různými metodami.

Vstupem do algoritmu dvojitě logaritmicky rozmístěných pólů je modulová kmitočtová charakteristika $M(\vartheta_n)$, kmitočty pólů f_p a počet FIR koeficientů. Výpočet koeficientů paralelních sekcí probíhá podle postupu popsaného v kap. 4.3.1.



Obr. 6.1: Blokové schéma výpočtu koeficientů paralelního filtru – dvojitě logaritmicky rozmístěné póly.

Při použití borcení kmitočtové osy musí být specifikovaný parametr λ a také počet pólů N_p. Výpočet pólů je potom složitější než při první metodě, protože nyní nemáme polohy pólů explicitně zadané. V bloku výpočtu pólů je využita Kallmanova metoda pro prvotní návrh pozic pólů, na kterou navazuje Steiglitz-McBride iterační metoda (viz kap. 3.3.2).

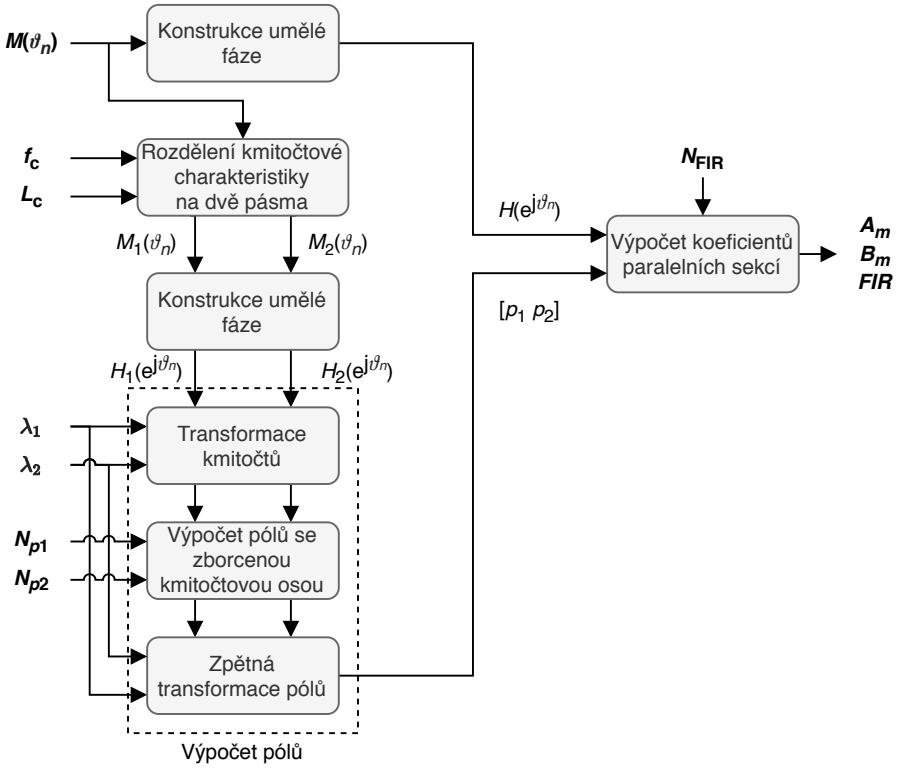


Obr. 6.2: Blokové schéma výpočtu koeficientů paralelního filtru – *jednoduchý warping.*

Při dvojitém borcení kmitočtové osy přibydou parametry pro dělicí kmitočet f_c a délku překryvu L_c. Modulová kmitočtová charakteristika se potom rozdělí na dvě části podle dělicího kmitočtu tak, že v oblasti pod, resp. nad dělicím kmitočtem jsou vzorky původní kmitočtové charakteristiky a v druhé části charakteristiky je konstantní hodnota

kmitočtové charakteristiky na dělicím kmitočtu. Tímto způsobem budou při návrhu póly umístěny pouze v pásmu, které je pro ně vyhrazeno [17].

Výpočet pólů proběhne potom se zborcenou kmitočtovou osou pro každé pásmo zvlášť a výsledné póly se potom spojí do jednoho vektoru.



Obr. 6.3: Blokové schéma výpočtu koeficientů paralelního filtru – *dvojitý warping.*

Umělé vytvoření fáze k zadané modulové kmitočtové charakteristice se v algoritmu nachází na více místech a je jeho slabou stránkou. Kmitočtová charakteristika systému s minimální fází má tvar [24]

$$H(\mathrm{e}^{\mathrm{j}\vartheta_n}) = M(\vartheta_n) \cdot \mathrm{e}^{\mathrm{j}\phi(\vartheta_n)}, \tag{6.1}$$

kde $M(\vartheta_n)$ je modulová kmitočtová charakteristika a $\phi(\vartheta_n)$ je fázová. Fázovou kmitočtovou charakteristiku lze vypočítat na základě modulové vztahem Hilbertovy transformace [24]

$$\phi(\vartheta_n) = -\mathcal{H}\{\ln(M(\vartheta_n))\}, \tag{6.2}$$

kde $\mathcal{H}$ značí Hilbertovu transformaci.

Jádrem Hilbertovy transformace je Fourierova transformace, která vykazuje lineární kmitočtové rozložení. Poskytnutá modulová kmitočtová charakteristika, většinou v logaritmickém rozložení, se musí interpolovat na lineární rozložení a pro dosažení dostatečného kmitočtového rozlišení na nízkých kmitočtech je nutné zvolit vysoký počet bodů (pro použití rychlé Fourierovy transformace je vhodné volit mocniny 2). V algoritmu je v základu

použitý počet bodů $N = 2^{14}$ (16384). Dostatečných výsledků je dosaženo i při použití $N = 2^{12}$, s nižší mocninou je však rozlišení nedostatečné a návrh je nepřesný na nízkých kmitočtech.

Protože délka takto interpolované odezvy je většinou několikrát větší, než délka zadaní, přináší výpočty s takto dlouhými vektory značné procesní zpoždění. Navrhl jsem tedy metodu, jak k tomuto problému přistoupit nelineárně a dosáhnout tak vyššího rozlišení na nízkých kmitočtech, za cenu nižšího rozlišení na vysokých.

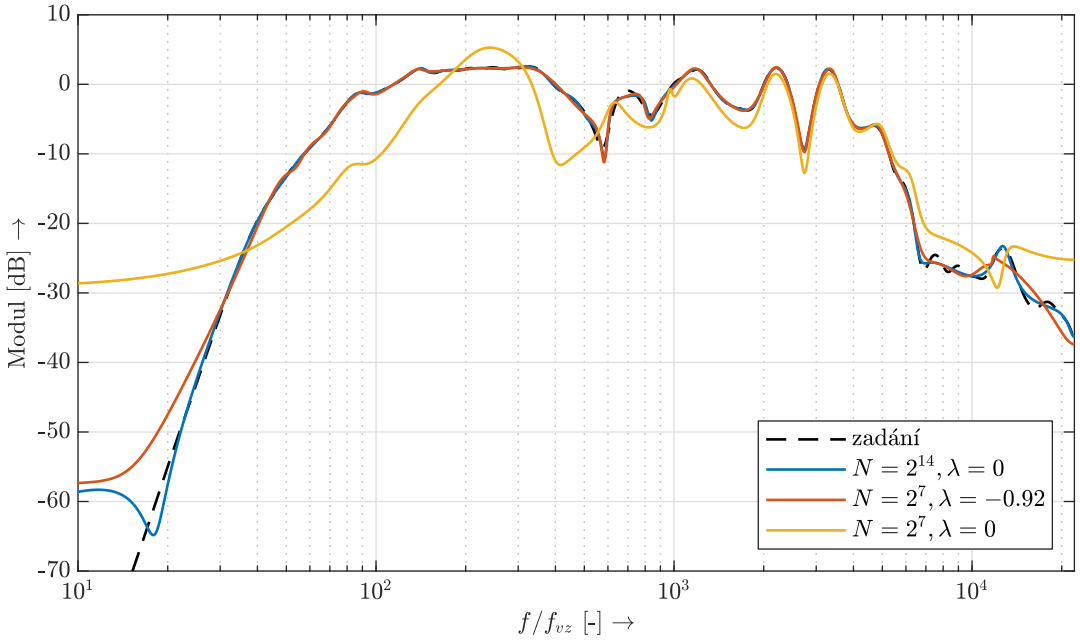
Kroky při výpočtu uměle vytvořené fáze jsou následující (převzato ze zdrojových souborů dostupných z [16]):

1. Vytvoření vektoru lineárně rozmístěných kmitočtů o délce $N + 1$, $\omega_n \in \langle 0, \pi \rangle$.
2. Interpolace modulové kmitočtové charakteristiky. Operátor $\mathcal{S}$ značí kubickou interpolaci.
$$M_{\text{lin}}(\omega_n) = \mathcal{S}(M(\vartheta_n))$$
3. Symetrizace kmitočtové charakteristiky – zamezení problémům na okrajích charakteristiky. Navíc při použití Fourierovy transformace dostaneme reálné spektrum.
$$M_{\text{perlin}}(\omega_m) = \begin{cases} M_{\text{lin}}(\omega_n) & \text{pro } m, n \in \langle 0, N \rangle \\ M_{\text{lin}}(\omega_{N-n}) & \text{pro } m \in \langle N+1, 2N \rangle,\ n \in \langle 1, N \rangle \end{cases}$$
4. Výpočet fáze v lineárním rozložení.
$$\phi_{\text{perlin}}(\omega_m) = -\mathcal{H}\{\ln(M_{\text{perlin}}(\omega_m))\}$$
5. Interpolace fáze zpět do logaritmického rozložení.
$$\phi(\vartheta_n) = \mathcal{S}\{\phi_{\text{perlin}}(\omega_m)\}, \quad \text{pro } m \in \langle 0, N \rangle$$
6. Výpočet komplexní kmitočtové charakteristiky podle rovnice (6.1).

Kroku 1. předchází volba N tak, aby bylo dostatečné rozlišení na nízkých kmitočtech (při $f_s = 44\,100\,\text{Hz}$ a $N = 2^{14}$ dostáváme rovnoměrné kmitočtové rozlišení zhruba 2,7 Hz, při $f_s = 192\,000\,\text{Hz}$ pouze 11,7 Hz). Pro výpočet fáze v nelineárním rozlišení je potřeba změnit první krok. Vektor lineárně rozmístěných kmitočtů se transformuje podle rovnice (5.1) použité v metodě návrhu filtru se zborcenou kmitočtovou osou. Tentokrát ovšem s použitím záporného koeficientu λ. Hodnoty, které lze použít, jsou libovolné; nejlepších výsledků bylo dosaženo při použití $\lambda \in \langle -0{,}95, -0{,}8 \rangle$.

Na obr. 6.4 jsou vidět návrhy filtru při použití různých parametrů. Při použití lineárního rozložení byla použita délka 2^{14}. Při zvolení $\lambda = -0{,}92$ bylo možné snížit délku interpolované odezvy až na $N = 2^7 = 128$ a přitom dosáhnout velmi dobrých výsledků, jako při použití lineárního rozložení (samotná délka vstupní modulové kmitočtové charakteristiky je asi 1000 vzorků, takže při výpočtu fáze došlo vlastně k podvzorkování). Pro porovnání je při té samé délce vynesen návrh při použití lineárního rozložení ($\lambda = 0$). Je

vidět, že zkreslením kmitočtové osy lze dosáhnout výrazného snížení výpočetní náročnosti při zachování přesnosti návrhu.



Obr. 6.4: Ukázky návrhů filtru při lineárním a nelineárním výpočtu fáze.

Pro představu, při použití $N = 2^7$ je při $f_s = 44\,100$ Hz konstantní kmitočtové rozlišení asi 172 Hz, při nelineárním rozlišení s $\lambda = -0{,}92$ je rozlišení na začátku spektra zhruba 7 Hz a exponenciálně roste až po 4000 Hz na konci.

6.1 Problémy při návrhu paralelního filtru

Výsledky návrhu filtru s paralelními sekcemi 2. řádu jsou parametry čitatele a jmenovatele elementárních přenosových funkcí. Kmitočtovou charakteristiku filtru lze potom zobrazit v jakémkoliv rozsahu kmitočtů od 0 do $f_{\mathrm{vz}}/2$, avšak pro jednoduché zobrazení se jako na obr. 5.1 volí rozsah kmitočtů, na kterých je definovaná cílová kmitočtová charakteristika.

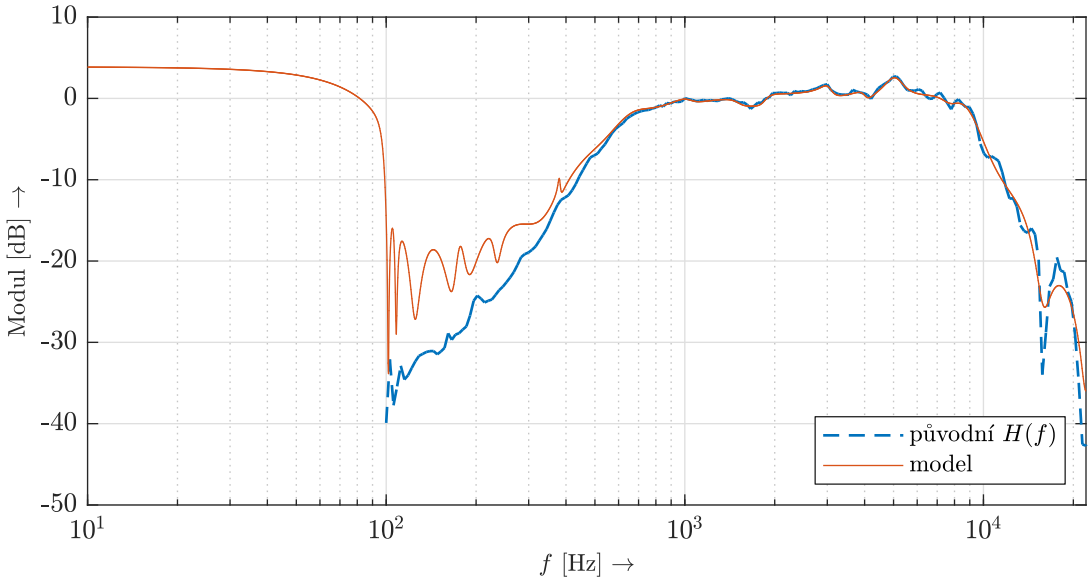
Pokud však cílová kmitočtová charakteristika bude definovaná na velmi omezeném rozsahu kmitočtů – např. kvůli ochraně zvukového zařízení před jeho zničením na příliš nízkých kmitočtech při měření – návrh filtru sice lze provést, ale jeho výsledek bude optimalizovaný pouze pro rozsah kmitočtů, na kterých bude definováno zadání.

Na obr. 6.5 vidíme návrh paralelního filtru pro změřenou kmitočtovou charakteristiku tabletu iPad. Je vidět, že v rozsahu kmitočtů, kde je zadání definované, je návrh filtru poměrně přesný a kopíruje zadanou kmitočtovou charakteristiku. Hůř sleduje kmitočtovou charakteristiku na jejím začátku a konci a v oblastech, kde žádná definice zadání neexistuje, je průběh filtru nedeterminovatelný.

Vezměme v úvahu případ, kdy bychom pomocí návrhu filtru chtěli simulovat vlastnosti kmitočtové charakteristiky změřeného tabletu na jiném zařízení (např. studiových monitorech). Pod 100 Hz by v tomto případě byl vstupní signál ještě zesílen, což v žádném případě neodpovídá realitě.

Je třeba proto navrhnout úpravu cílové kmitočtové charakteristiky, a provést extrapolaci jejího průběhu v oblastech, které jsou pro návrh modelu důležité. Za tímto účelem byla navržena metoda extrapolace, která postupuje v těchto krocích:

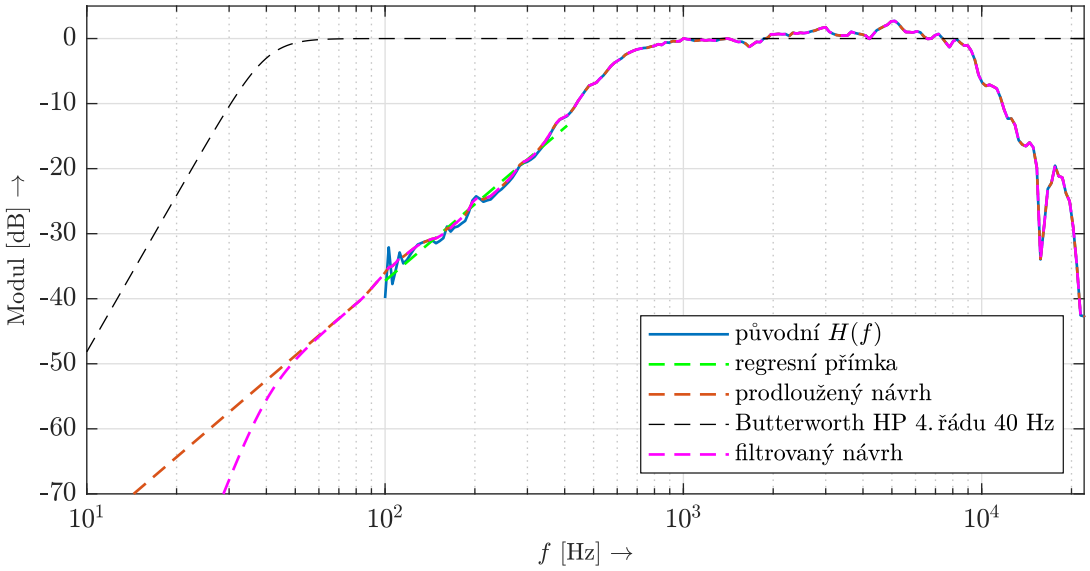
1. Z okrajového průběhu kmitočtové charakteristiky (např. první dvě oktávy) se vypočítá regresní přímka, která odpovídá průběhu kmitočtové charakteristiky v tomto pásmu.
2. Definuje se druhý průběh kmitočtové charakteristiky, jehož hodnoty odpovídají od **nového** počátečního kmitočtu (např. 10 Hz) do počátečního kmitočtu zadání regresní přímce, a poté prvním dvěma oktávám změřené kmitočtové charakteristiky.
3. Provede se třetinooktávové vyhlazení tohoto průběhu.
4. Vyhlazený průběh se sečte s původním zadání, přičemž v překrývajícím se pásmu se provede jejich váhovaný součet (*crossfade*).
5. Kvůli ochraně přehrávacího zařízení se výsledný průběh může sečíst s kmitočtovou charakteristikou filtru typu HP Butterworthovou aproximací 4. řádu na zvoleném nízkém kmitočtu (40 Hz).



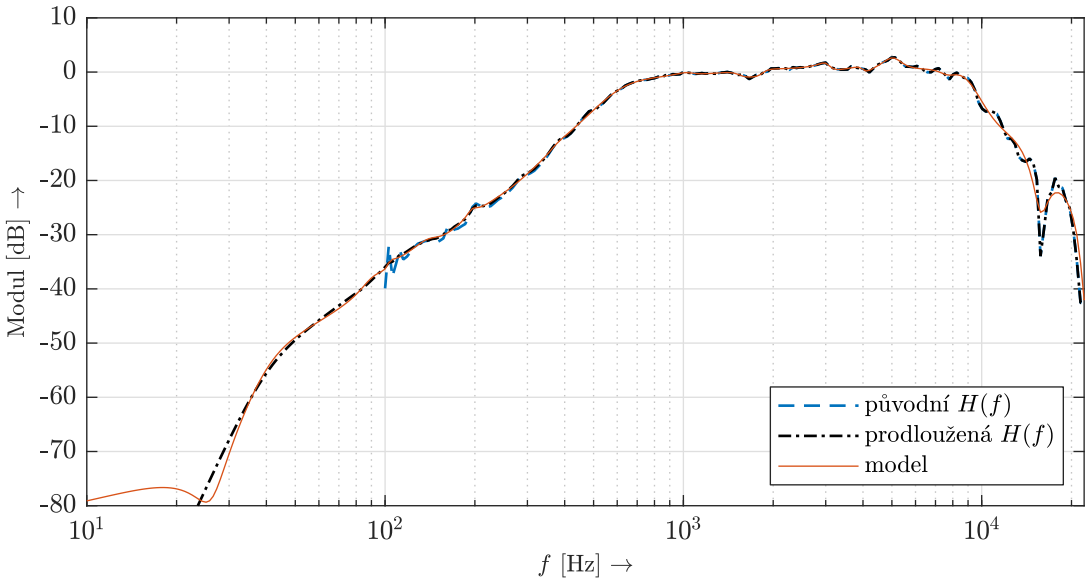
Obr. 6.5: Návrh paralelního filtru pro omezenou délku kmitočtové charakteristiky.

Takto navržená extrapolovaná kmitočtová charakteristika a jednotlivé kroky postupu jsou zobrazeny na obr. 6.6. Výsledný filtr navržený po této úpravě je potom na obr. 6.7. Je vidět, že i v kmitočtovém pásmu, kde nebylo zadání definované nehrozí nechtěné nebo nebezpečné zvukové artefakty. V adresáři `MATLAB/response extrapolation` je příklad využívající funkce k prodloužení a vyhlazení kmitočtové charakteristiky. Obsažena je i funkce

pro extrapolaci do vyšších kmitočtů, která má analogický postup.



Obr. 6.6: Prodloužení zadané kmitočtové charakteristiky.



Obr. 6.7: Návrh filtru s prodloužením zadané kmitočtové odezvy.

6.2 Problémy při implementaci algoritmu v programovacím jazyce

Klíčovými kroky při návrhu filtru metodou dvojitého borcení kmitočtové osy je vyřešení soustavy rovnic metodou nejmenších čtverců. S touto operací se setkáme při výpočtu pólů paralelních filtrů a poté při výpočtu nulových bodů jednotlivých paralelních sekcí. Soustava rovnic se řeší maticovým počtem, pomocí tzv. pseudoinverzní matice. Je potřeba vyřešit rovnice (3.63) a (4.15), resp. (4.16)

$$\mathbf{p} = \mathbf{\Phi}^{-1}\mathbf{c},$$
$$\mathbf{p}_{\mathrm{opt}} = \left(\mathrm{Re}\left\{\mathbf{M}^{\mathrm{H}}\mathbf{M}\right\}\right)^{-1} \mathrm{Re}\left\{\mathbf{M}^{\mathrm{H}}\mathbf{h}_{\mathrm{d}}\right\}.$$

Pokud první z operandů má být inverzní maticí sebe sama, odpovídá operace tzv. dělení matic zleva. V prostředí MATLAB k této operaci slouží operátor „\“, jehož popisem je řešení soustavy lineárních rovnic (angl. *solver*). MATLAB při řešení soustavy rovnic zvolí optimální algoritmus, aby se minimalizoval výpočetní čas. Zvolení vhodného algoritmu závisí na tvaru matice, hodnotách prvků a jestli je matice řídká, nebo plná (hustá).

V obou případech je matice plná, poté se algoritmus volí podle diagramu na obr. B.1 [28].

V případě rovnic (3.63) a (4.16) jsou matice levého operandu čtvercové, Hermitovské a na diagonále mají vždy kladná čísla. Podle diagramu se volí buď algoritmus *Cholesky* – LL nebo LDL.

Na zvolení vhodného algoritmu pro vyřešení soustavy rovnic bude třeba dbát při implementaci algoritmu v jiném programovacím jazyce než MATLAB.

6.3 Realizace v jazyce C++

Funkce, které zajistí výpočet koeficientů přenosové funkce paralelních filtrů způsoby, které jsou vyobrazeny na obr. 5.1, byly implementovány v jazyce C++. Protože MATLAB disponuje obrovskou řadou interních funkcí, byly kromě zásadních funkcí uplatňujících teorii popsanou touto prací vytvořeny také funkce nahrazující vnitřní funkce MATLABu (`linspace`, `logspace`, `roots`, `poly`, `polyval`, `hilbert` apod.).

Vyjádření číselných posloupností je zpravidla ve struktuře `std::vector<double>` nebo `std::vector<complex<double>>`. Pro zjednodušenou operaci s vektory byly implementovány různé přetížené funkce a operátory (`abs`, `log`, `log10`, `exp` apod.).

V příloze A je podrobný rozpis struktury funkcí a informace o použitých knihovnách a algoritmech.

6.4 Realizace v jazyce Python

Pro otestování výsledků při implementaci v jiném jazyce než C++ jsem potřebné funkce přepsal do jazyka Python (zdrojový kód v adresáři `Python`). Operace s maticemi a vektory

dostatečně zajistí knihovna *NumPy*[1] a pro řešení soustavy lineárních rovnic jsou k dispozici funkce `solve()` v knihovně *NumPy* nebo *SciPy*[2]. Obě knihovny využívají ve svém jádru algoritmy knihovny LAPACK[3], ale detailnější informace o postupu zvolení optimálního algoritmu pro výpočet v dokumentaci nejsou.

6.5 Porovnání výsledků realizací v jiných prostředích

V adresáři `C++/ParallelFilters/ParallelFilters` se nachází projekt *ParallelFilters*. Spuštěním funkce `main` budou pro zadaná data vypočítány koeficienty přenosové funkce paralelních filtrů metodou dvojitě logaritmicky rozmístěných pólů, jednoduchého a dvojitého *warpingu* a uloženy do souboru CSV.

Porovnáme nyní výsledky implementace v prostředí MATLAB, kterou navrhl Balázs Bank [16],v C++ a v jazyce Python. V tab. 6.1 jsou vypsány parametry použité při návrzích. Výsledky vidíme na obr. 6.9 a obr. 6.11 a v tabulkách 6.2 6.3 a 6.4. Pro vyhodnocení byly použity metody popsané v kap. 5.3.

Protože návrhy kopírují zadanou kmitočtovou charakteristiku poměrně přesně a návrhy v různých programovacích jazycích se téměř překrývají, na obr. 6.10 a 6.12 jsou vykresleny rozdíly zadané kmitočtové charakteristiky a návrhů. Zde je lépe patrné, ve kterých částech spektra je který algoritmus přesnější. I takto se výsledky vypočítané v prostředí MATLAB a v jazyce Python téměř stoprocentně překrývají.

K porovnání výsledků návrhů při použití jiných implementací poslouží dvě testovací kmitočtové charakteristiky. V prvním případě jde o kytarový reprobox Engl 4x12. Měření proběhlo v bezodrazové komoře v laboratoři VUT FEKT SC5.50 s použitím měřicího mikrofonu Brüel & Kjær 4189-A-021 ve vzdálenosti 2,5 cm od středu reproduktoru. Druhá charakteristika vznikla změřením sluchátek Numark RedWave na simulátoru hlavy a torsa (HATS) Brüel & Kjær Type 4128-C. Zpracování měření proběhlo audio analyzátorem APx525 s rozhraním APx1701 Transducer Test Interface.

Je vidět, že návrhy ve všech prostředích se mimo okrajové hodnoty kmitočtové charakteristiky téměř shodují. Po důkladném zkoumání jednotlivých kroků algoritmů bylo zjištěno, že rozdíly jsou způsobeny při řešení soustavy lineárních rovnic – dělení matic zleva. Každá implementace využívá svoje optimální prostředky a knihovny pro maticové operace (viz. kap. 6.3 a 6.4).

Další rozdíly při zpracování vznikají při použití kubické interpolace – funkce `spline()`, která se v průběhu výpočtu využívá při konstrukci umělé fáze vstupní kmitočtové charakteristiky. MATLAB při použití kubické interpolace pomocí funkce `spline()` používá okrajové podmínky typu tzv. *not-a-knot* [29, 30]. Na obr. 6.8 je vidět detail návrhů v okrajích kmitočtových charakteristik navržených filtrů – MATLAB a Python pracuje s *not-a-knot* okrajovými podmínkami, proto jsou průběhy podobné. Implementace C++ pro okrajové

[1] NumPy – `https://numpy.org/doc/stable/index.html`
[2] SciPy – `https://docs.scipy.org/doc/scipy/reference/index.html`
[3] LAPACK – Linear Algebra PACKage – `http://www.netlib.org/lapack/`

podmínky používá tzv. *přirozený spline*[4] a extrapolace je potom lineární. Pro úplnost jsem implementoval funkci v C++, která využívá okrajové podmínky *not-a-knot* podle [30] a souborů dostupných z [31].

Je třeba říci, že všechny zobrazené „původní“ průběhy kmitočtových charakteristiky jsou šestinooktávově vyhlazeny.

Zvýrazněná data v tabulkách jsou ta, která odpovídají nejvyšší shodě zadání s navrženým filtrem. Nejúspěšnější se zdá implementace v C++, která používá pro okrajové podmínky *přirozený spline*. Pro dané příklady se tedy zdá, že je použití těchto okrajových podmínek v algoritmu vhodnější, ale obecně pro každé zadání mohou být výsledky jiné.

V případě, že lepší výsledky poskytuje MATLAB nebo Python, jejich hodnoty se v tabulkách 6.2 a 6.3 liší v řádu asi 10^{-7}, zvýrazněna je vždy ta lepší hodnota.

Tab. 6.1: Použité parametry návrhů.

	logaritmicky	warping	dvojitý warping
počet pólů 1. pásmo	18	40	18
počet pólů 2. pásmo	22		22
λ_1		0,92	0,986
λ_2			0,65
dělicí kmitočet [Hz]	500		500
délka překryvu [vzorky]			50

Tab. 6.2: Porovnání jednotlivých implementací – dvojitě logaritmicky rozmístěné póly.

	Engl			Numark		
	MATLAB	C++	Python	MATLAB	C++	Python
MSE	124,0387	**120,0658**	124,0387	2,2725	**2,1880**	2,2725
ρ	0,9469	**0,9478**	0,9469	0,9854	**0,9859**	0,9854
σ	**407,0530**	408,2040	407,0530	186,9223	**182,1615**	186,9223
oct MSE	70,3308	**67,6672**	70,3308	0,0292	0,0300	**0,0292**
1/3 oct MSE	32,1144	**30,1702**	32,1144	0,0428	0,0582	**0,0428**
Zwicker MSE	5,6080	**5,2763**	5,6080	0,0290	**0,0286**	0,0290

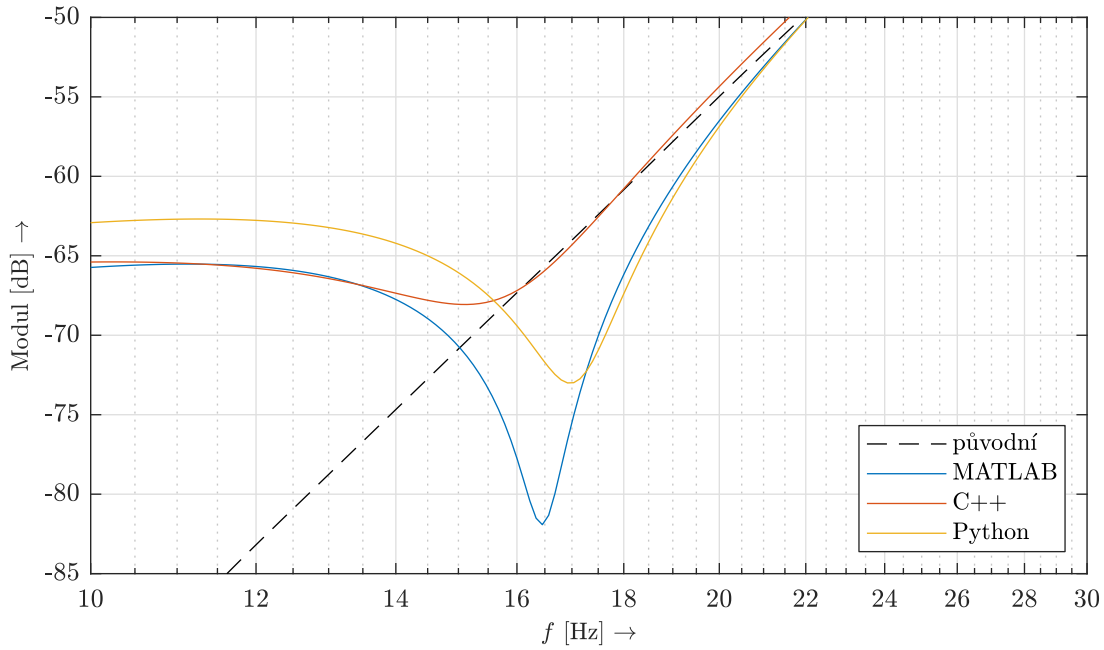
[4]Druhá derivace spline polynomu na okrajových hodnotách je rovna 0.

Tab. 6.3: Porovnání jednotlivých implementací – *jednoduchý warping*.

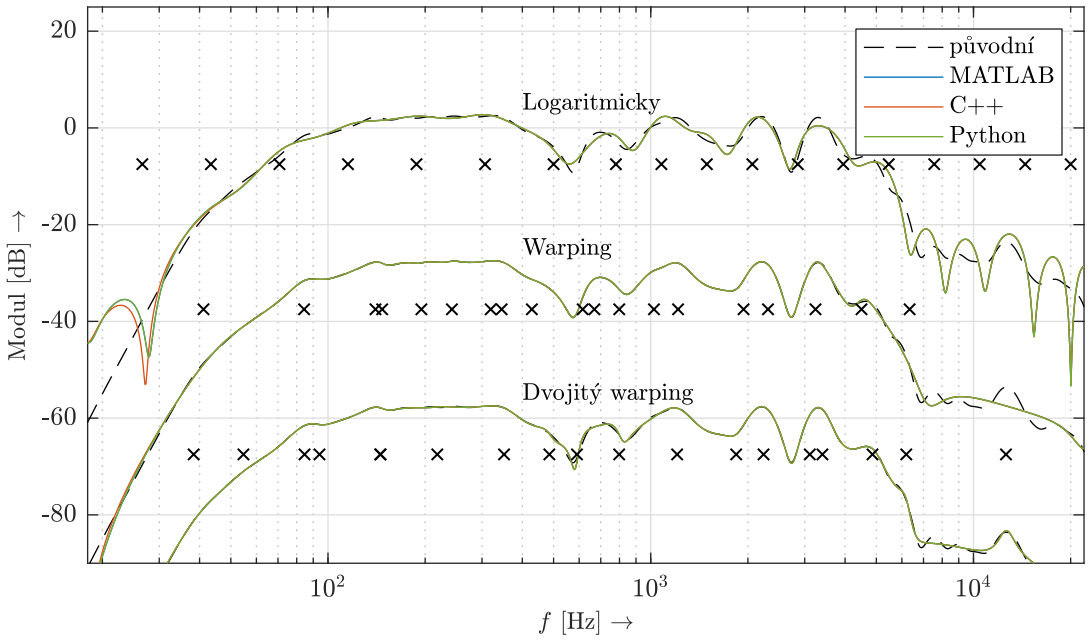
	Engl			Numark		
	MATLAB	C++	Python	MATLAB	C++	Python
MSE	**23,1036**	35,1250	23,1036	0,2187	**0,1869**	0,2187
ρ	**0,9889**	0,9813	0,9889	0,9986	**0,9988**	0,9986
σ	**35,6830**	39,8998	35,6830	24,4120	25,4762	**24,4119**
oct MSE	3,8595	4,4498	**3,8595**	0,0044	**0,0014**	0,0044
1/3 oct MSE	**1,1869**	1,2544	1,1869	0,0459	**0,0242**	0,0459
Zwicker MSE	0,5891	0,7088	**0,5891**	0,0295	**0,0195**	0,0295

Tab. 6.4: Porovnání jednotlivých implementací – *dvojitý warping*.

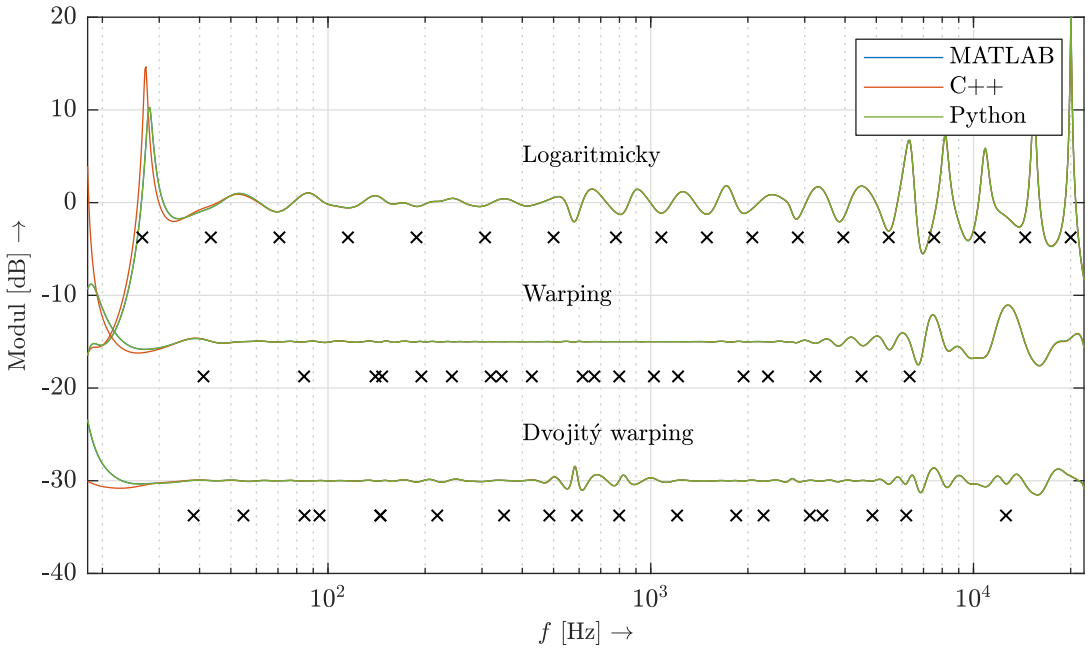
	Engl			Numark		
	MATLAB	C++	Python	MATLAB	C++	Python
MSE	22,4472	**17,1914**	23,6156	0,0254	**0,0196**	0,0255
ρ	0,9881	**0,9925**	0,9878	0,9998	**0,9999**	0,9998
σ	54,8471	**51,8537**	55,5236	20,6662	**12,9586**	20,7418
oct MSE	**2,1173**	3,4035	2,6948	3,5177e−4	**1,1617e−4**	3,4661e−4
1/3 oct MSE	0,3406	**0,1325**	0,2542	4,6148e−3	**3,6476e−3**	4,6371e−3
Zwicker MSE	**0,3059**	0,3901	0,3567	2,2610e−4	**2,0695e−4**	2,3116e−4



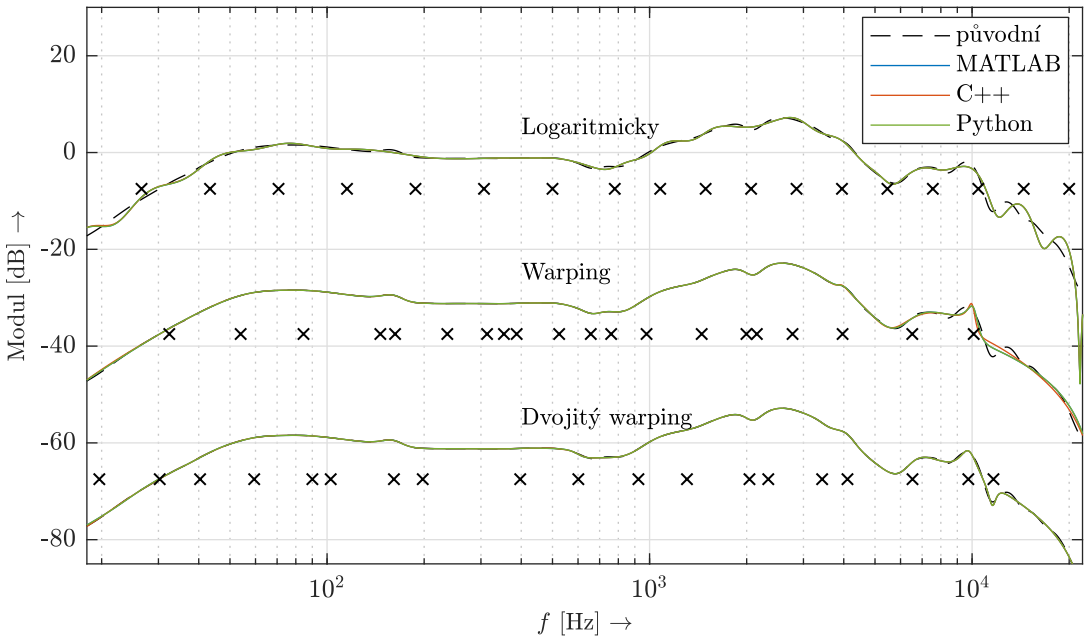
Obr. 6.8: Detail výsledků návrhu paralelních filtrů pro kytarový box Engl 4x12 při použití *dvojitého warpingu* a okrajových podmínek v C++ *přirozený spline*.



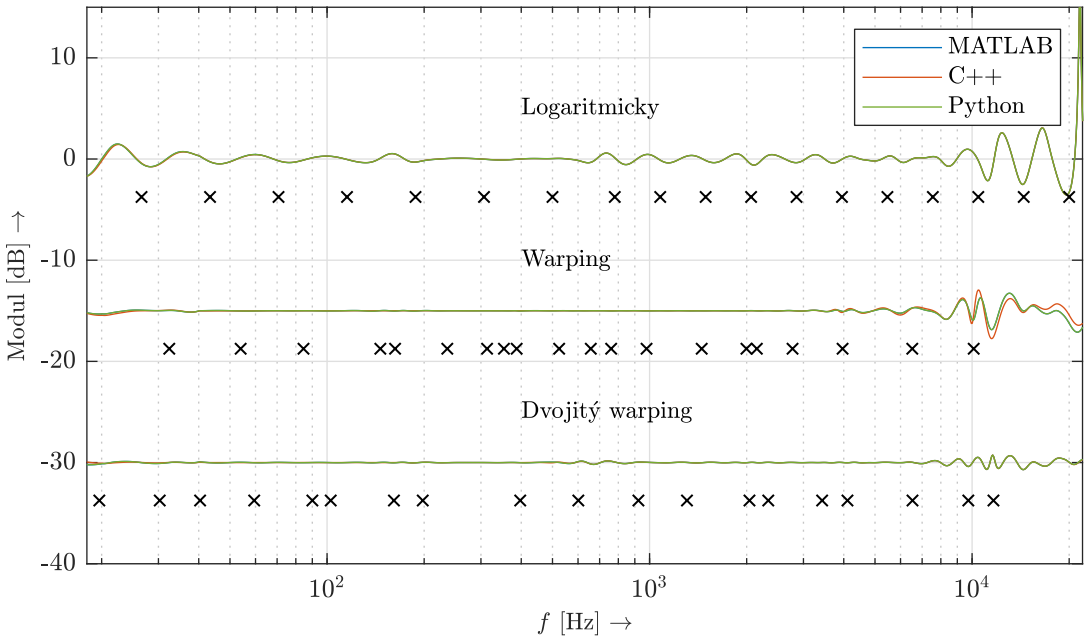
Obr. 6.9: Porovnání výsledků návrhu paralelních filtrů pro kytarový reprobox Engl 4x12.



Obr. 6.10: Rozdíl návrhů paralelních filtrů a zadání – kytarový reprobox Engl 4x12.



Obr. 6.11: Porovnání výsledků návrhu paralelních filtrů pro sluchátka Numark RedWave.



Obr. 6.12: Rozdíl návrhů paralelních filtrů a zadání – sluchátka Numark RedWave.

6.6 Výsledky PSO

Pro čtyři zvolené kmitočtové charakteristiky byl proveden výpočet optimálních parametrů pomocí PSO algoritmu. První dvě jsou již zmíněné charakteristiky kytarového reproboxu Engl a sluchátek Numark. Dále potom charakteristika kytarového reproboxu Marshall a charakteristika interiéru auta Škoda Fabia na místě řidiče.

Bude použitý algoritmus SPSO 2006. Počet optimalizovaných parametrů je 5:

- Počet pólů pro nízké kmitočty.
- Parametr λ_1.
- Parametr λ_2.
- Dělicí kmitočet.
- Délka překryvu dvou pásem.

Celkový počet pólů je pevně nastavený na 48, pro vysoké kmitočty se tedy počet pólů dopočítá.

Parametry PSO:

- Počet částic: norma SPSO 2006 doporučuje velikost hejna $S = 10 + \left\lfloor 2\sqrt{D} \right\rfloor$, kde D je počet dimenzí problému [22]. V tomto případě $D = 5$, tedy podle vzorce $S = 14$. Pro testování bylo použito 15 částic.
- Počet iterací: 35
- Parametry konvergence w, c_1, c_2. Dle verze algoritmu představené Clercem a Kennedym – *Constriction* PSO (CPSO) [23] se parametry vypočítají na základě následujících rovnic

$$\phi_1 = 2{,}05, \phi_2 = 2{,}05, \phi = \phi_1 + \phi_2, \chi = \frac{2\kappa}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|},$$

kde $\kappa \in \langle 0, 1 \rangle$ a $\phi_1 + \phi_2 > 4$.

Potom

$$w = \chi,$$
$$c_1 = \chi \cdot \phi_1,$$
$$c_2 = \chi \cdot \phi_2.$$

Pro každou charakteristiku byly vypočítány 4 návrhy filtru, protože nastavení počátečních parametrů probíhá náhodně, tak se výsledky liší. Výsledky návrhů filtrů vidíme na obr. 6.14 až 6.17.

Hodnoty MSE nebo jiných spočítaných koeficientů nelze mezi sebou porovnávat, protože se pokaždé počítají z jiného počtu definovaných bodů. Nyní lze spíše subjektivně posoudit, která fitness funkce se nejlépe hodí pro vyhodnocení. V tab. 6.5 jsou zapsány průměrné výsledky přesnosti návrhů filtrů vyhodnocené popsanými metodami.

Na obr. 6.16 vidíme výsledky návrhu filtru při použití fitness funkce výpočtu MSE v oktávových pásmech. Je vidět, že ve středním pásmu kmitočtů žádný návrh nebyl schopný dostatečně přesně sledovat cílovou charakteristiku. Naproti tomu při použití výpočtu MSE

Tab. 6.5: Průměrné výsledky ze čtyř návrhů pomocí PSO.

	Engl	Numark	Marshall	Fabia
FullRespMSE	1,1242	5,8541e−3	1,0304	2,5481e−2
FracOctMSE	8,8879e−4	1,9717e−6	1,7422e−3	8,1985e−5
FracThirdMSE	1,8729e−2	1,7247e−4	2,6555e−2	6,6356e−4
CriticalMSE	7,4176e−4	1,7688e−6	5,8940e−3	7,5618e−5
Pearson	0,99902	0,99996	0,99868	0,99961
SpectDev	13,647	5,8885	13,871	9,6435

v kritických pásmech na obr. 6.17 jsou výsledky excelentní. V tomto případě není oktávové rozložení dostatečné.

V tabulce 6.5 a i dále v textu budou použity následující zkratky pro dané fitness funkce:

- **FullRespMSE** – MSE z celého průběhu kmitočtové charakteristiky.
- **FracOctMSE** – MSE v oktávových pásmech.
- **FracThirdMSE** – MSE v třetinooktávových pásmech.
- **CriticalMSE** – MSE v kritických pásmech.
- **Pearson** – Pearsonův korelační koeficient.
- **SpectDev** – *spectrum deviation.*

6.7 Váhování

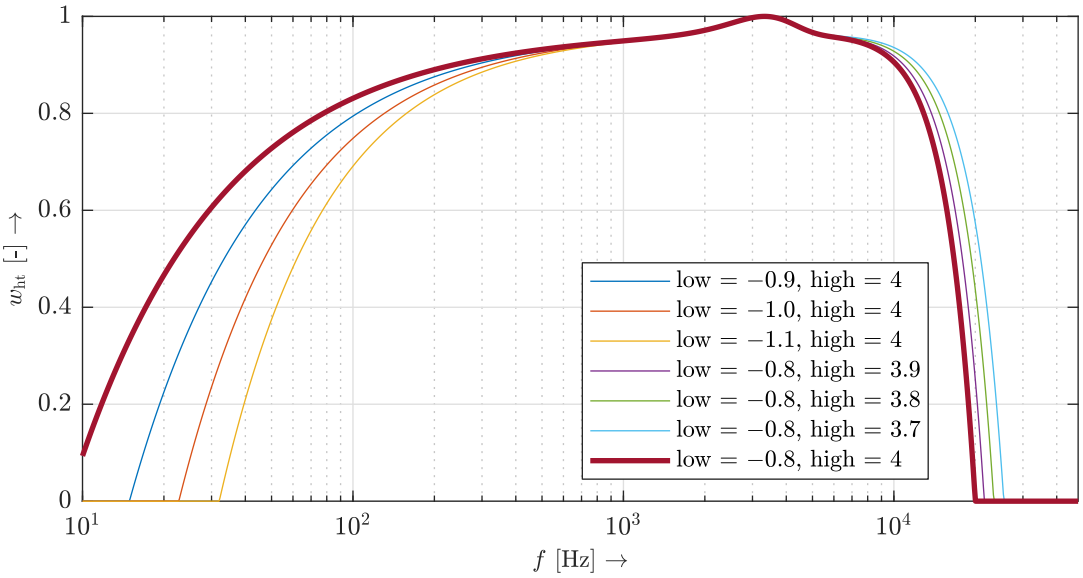
Nyní jsou výsledky fitness funkcí počítány bez jakéhokoliv váhování. Každý rozdíl při výpočtu MSE, ať už bod po bodu nebo v kmitočtových pásmech, má stejný podíl na výsledné odchylce. Pokud však bude mít návrh velkou odchylku na velice nízkých kmitočtech nebo naopak blízko Nyquistova kmitočtu a zároveň může filtr kopírovat velice dobře zadanou charakteristiku na středních kmitočtech, bude návrh vyhodnocený jako méně kvalitní, i když rozdíly nebudou prakticky slyšitelné.

6.7.1 Kmitočtové váhování

Jako vektor vah lze zvolit upravenou funkci prahu slyšení – pro kmitočty, kde je ucho nejcitlivější bude vypočtená chyba mít nejvyšší váhu, naopak pro nejnižší citlivost bude mít váhu nejnižší. V [32] je navržena aproximace prahu slyšení funkcí

$$L_{\mathrm{ht}}(f) = 3{,}64 \cdot \left(\frac{f}{1000}\right)^{-0{,}8} - 6{,}5 \cdot \mathrm{e}^{-0{,}6\left(\frac{f}{1000}-3{,}3\right)^2} + 10^{-3} \cdot \left(\frac{f}{1000}\right)^{4} \,[\mathrm{dB}] \tag{6.3}$$

jako vhodná regrese k datům změřených Zwickerem. Pomocí této funkce lze vypočítat hodnotu prahu slyšení na jakémkoliv kmitočtu. Normalizací a obrácením znaménka hodnot funkce obdržíme vektor vah v rozmezí 0 až 1. Pro danou aplikaci je však někdy vhodné tento průběh lehce upravit – např. pro návrhy filtrů s vyšším vzorkovacím kmitočtem. Dva zvýrazněné exponenty v rovnici funkce ovlivňují průběh funkce na nízkých, resp. vysokých kmitočtech. Na obr. 6.13 vidíme průběhy vektoru vah s různými hodnotami exponentů.



Obr. 6.13: Průběhy vektoru vah prahu slyšení s různými hodnotami exponentů.

Pro váhované MSE nebo Pearsonův korelační koeficient vzorce existují. Při výpočtu v kmitočtových pásmech se vektor vah vypočte jako průměrná hodnota v daném kmitočtovém intervalu. Pro *spectrum deviation* je logické upravit rovnici (5.6) na

$$\sigma = \frac{1}{N}\sum_{i=1}^{N}\sqrt{w_i\left(\frac{x_i - \frac{x_i+y_i}{2}}{\frac{x_i+y_i}{2}}\right)^2 + w_i\left(\frac{y_i - \frac{x_i+y_i}{2}}{\frac{x_i+y_i}{2}}\right)^2}, \tag{6.4}$$

kde w je vektor vah.

Na obr. 6.18 jsou vidět výsledky návrhu s použitím kmitočtového váhování s vektorem vah vypočítaným podle (6.3) a fitness funkcí pro výpočet MSE z oktávových pásem. Přidané váhování ukazuje zlepšení oproti výsledkům na obr. 6.16 bez váhování.

6.7.2 Váhování podle hodnoty

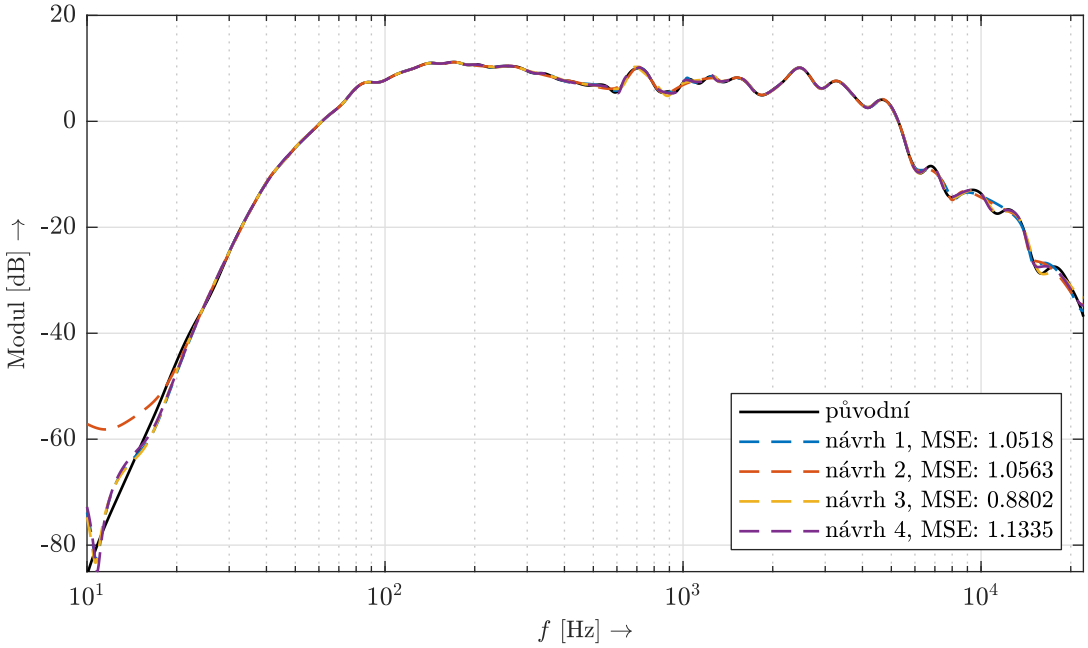
Váhování podle hodnoty lze provést tak, že nižším hodnotám kmitočtové charakteristiky bude přikládána nižší váha při výpočtu chyby. Princip je podobný jako průchod expanderem.

Výpočet vektoru vah probíhá v těchto krocích:

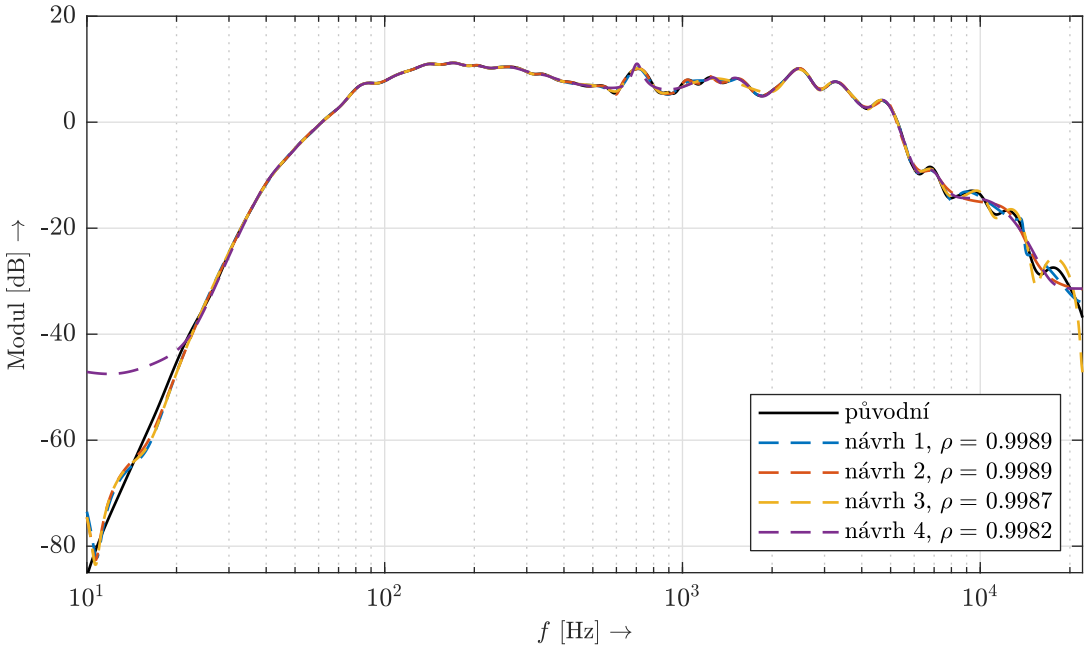
1. Vstupem jsou parametry expanderu *threshold*, *ratio* a *knee width*.
2. Vypočte se mediánová hodnota kmitočtové charakteristiky.
3. Threshold se upraví $T = \text{median}(H) - T$;
4. Na kmitočtovou charakteristiku se aplikuje expander.
5. Váha se vypočítá jako poměr charakteristik před a po expanzi.

Parametr *threshold* je rozhodovací úroveň ovlivnění charakteristiky v dB, *ratio* je míra ovlivnění pod rozhodovací úrovní a *knee width* šířka přechodového pásma. Původní a ex-

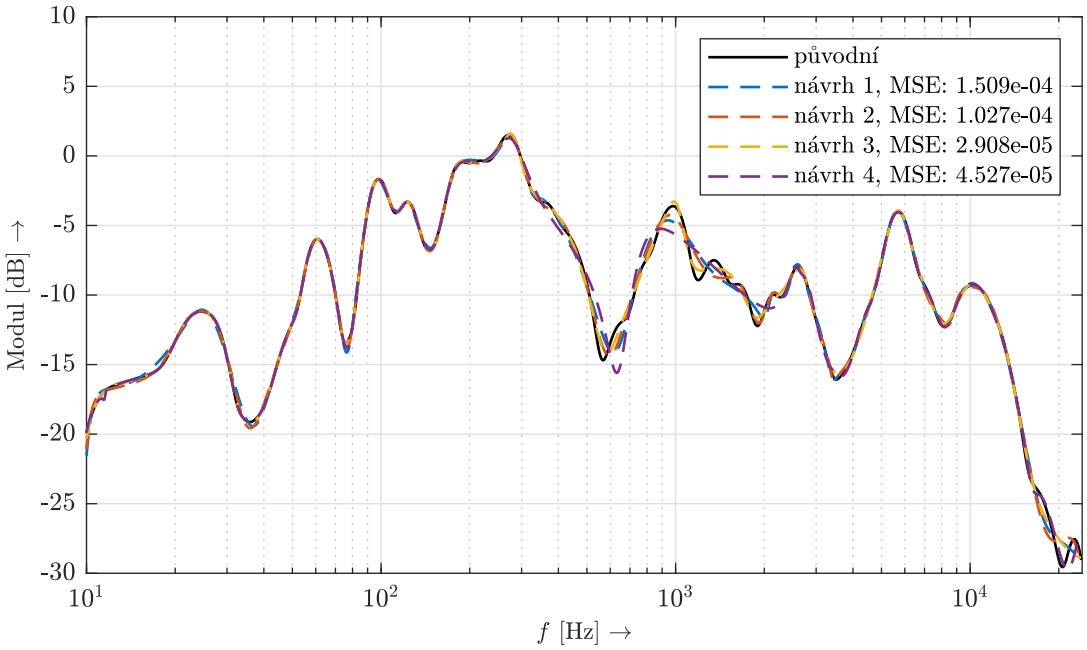
pandovanou kmitočtovou charakteristiku a vektor vah potom vidíme na obr. 6.19. Je možné použít také kombinaci váhování podle kmitočtu a hodnoty jejich vynásobením.



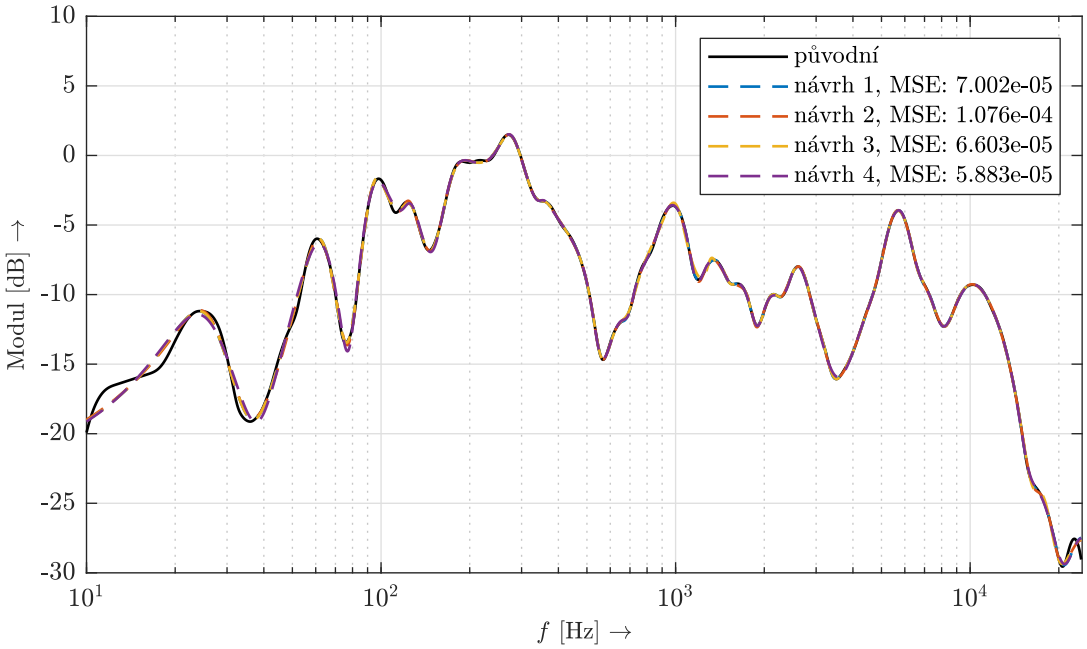
Obr. 6.14: Výsledky návrhů pomocí PSO FullRespMSE fitness – Marshall.



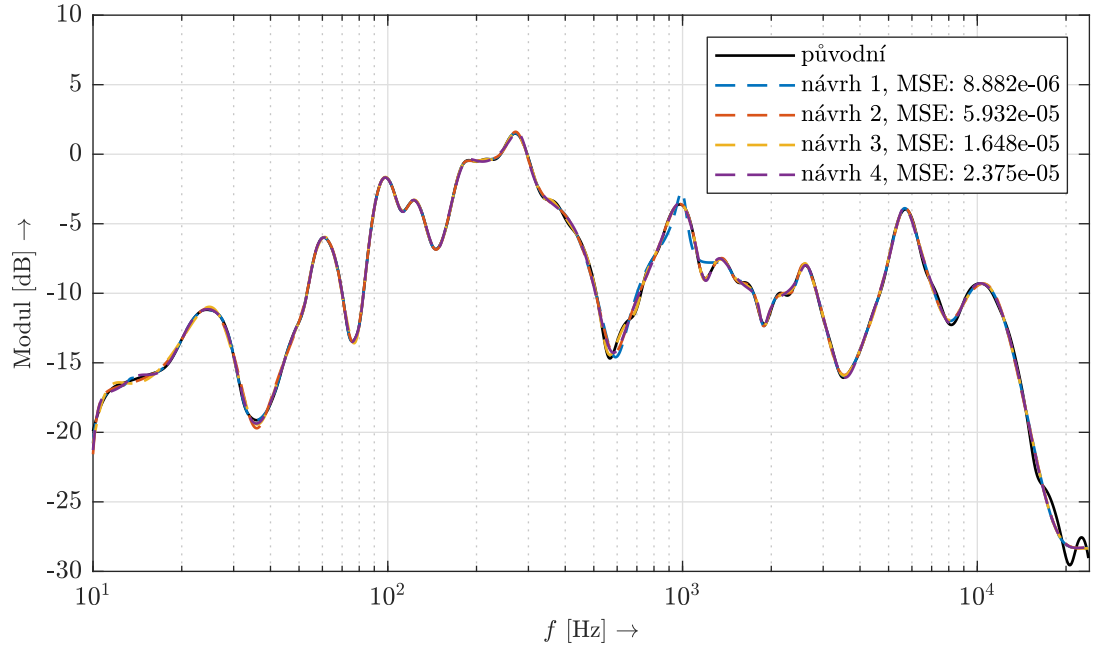
Obr. 6.15: Výsledky návrhů pomocí PSO Pearson fitness – Marshall.



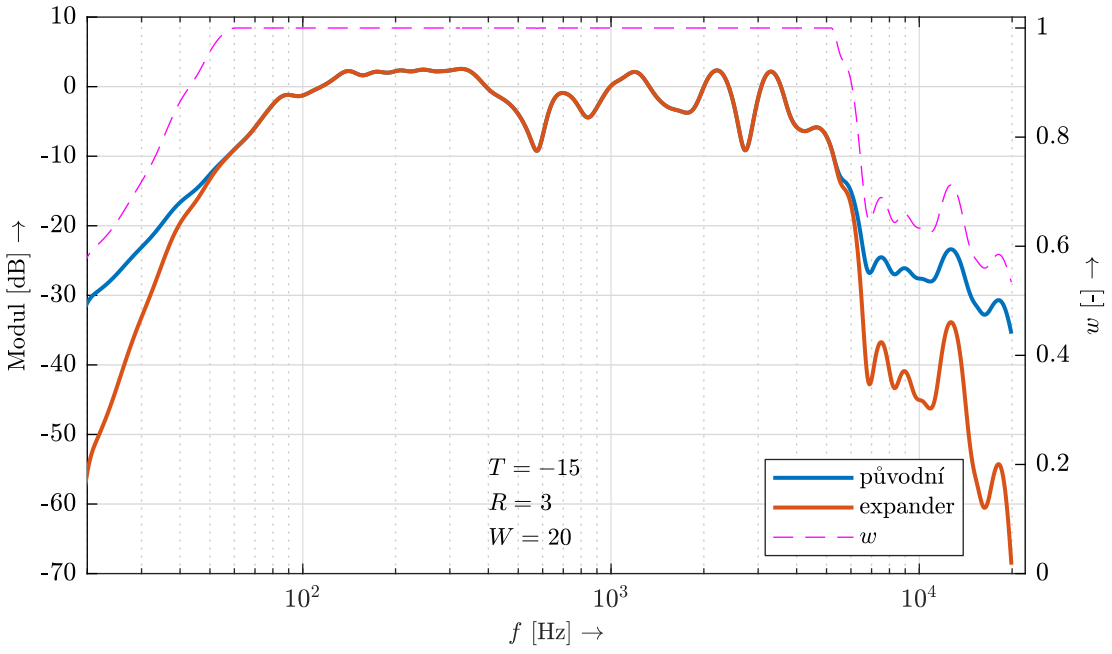
Obr. 6.16: Výsledky návrhů pomocí PSO FracOctMSE fitness – Fabia.



Obr. 6.17: Výsledky návrhů pomocí PSO CriticalMSE fitness – Fabia.



Obr. 6.18: Výsledky návrhů pomocí PSO FracOctMSE fitness a s použitým váhováním – Fabia.



Obr. 6.19: Ukázka výpočtu vektoru vah z hodnot charakteristiky.

7 VÝSLEDKY PRÁCE

Výsledkem této práce je algoritmus, který slouží k návrhu paralelního filtru. První část se věnuje různým realizacím algoritmů pro samotný návrh filtru, v druhé části je potom popsán algoritmus automatické optimalizace vstupních parametrů návrhu.

7.1 Návrh filtru

7.1.1 Algoritmus návrhu filtru v programovacích jazycích

Soubor funkcí, které zajistí výpočet koeficientů paralelních sekcí filtru v C++, se nachází v adresáři `C++/ParallelFilters/ParallelFilters` a je podrobně popsaný v příloze A.

Analogické funkce pro výpočet koeficientů v prostředí MATLAB původně navrhl Balázs Bank [16] a nacházejí se ve složce `MATLAB/common_code` v souboru **bankCore.m**.

Stejné funkce implementované v jazyce Python nalezneme ve složce `Python`.

7.1.2 MATLAB

Pro prozkoumání vlivu jednotlivých parametrů návrhu při použití *dvojitého warpingu* byla vytvořena již zmíněná aplikace *Fitting GUI* (obr. 5.2). Zdrojové soubory a skripty se nachází v adresáři `MATLAB/fittingGUI` a spustí se skriptem **fittingGUI.m**. Po načtení souboru ve vhodném formátu se provede návrh se zvolenými parametry a průběhy kmitočtových charakteristik zadání a navrženého filtru jsou zobrazeny v grafu spolu s pozicemi pólů.

Je možné vybrat mezi algoritmy implementovanými v prostředí MATLAB nebo v C++ a mezi různými možnostmi extrapolace při použití funkce `spline()`.

Ve složce `MATLAB/mex` se nachází několik souborů, které zajišťují kompilaci C++ kódu pro návrh filtru do obalovací (*wrapper*) `mex` funkce, která se volá pokud je pro výpočet koeficientů filtru zvolen algoritmus v C++.

Je také možné pouze číst soubory s výslednými koeficienty filtru a zobrazit tak průběh kmitočtové charakteristiky. Přesnější informace o chodu programu poskytne tlačítko s nápisem „Info“.

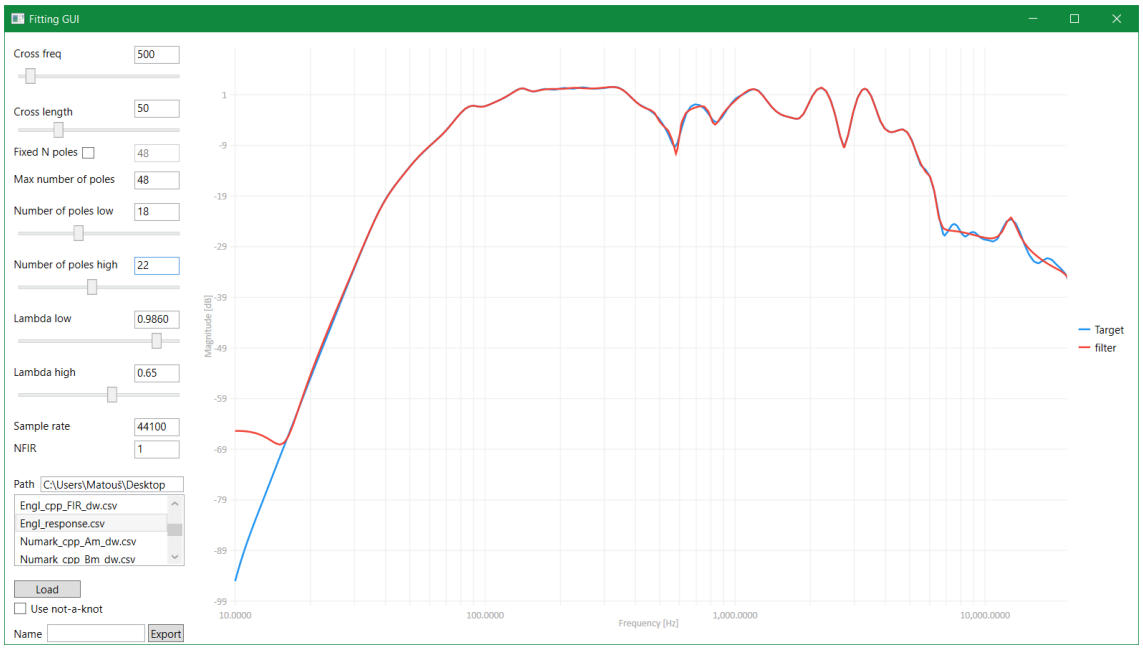
Zkompilovaná spustitelná aplikace je v adresáři `Output/MATLAB`.

7.1.3 Rozhraní ve WPF a jazyce C#

Pro využití knihovny pro výpočet parametrů paralelního filtru bez vazby na prostředí MATLAB byla vytvořena aplikace *Fitting GUI* v jazyce C# s grafickým uživatelským prostředím vytvořeným pomocí WPF (obr. 7.1). Poskytuje stejné možnosti jako aplikace v prostředí MATLAB – návrh filtru s definicí zobrazených parametrů a export výsledných koeficientů filtru do souboru ve formátu CSV. Ve stejném formátu souboru jsou také požadována vstupní data.

Jelikož je aplikace psaná v jazyce C# a vnitřní algoritmus pro návrh filtru je napsaný v jazyce C++, bylo potřeba implementovat rozhraní (API), které bude využívat funkce napsané v C++. V souboru **ParFiltDesign.cs** nalezneme deklarace funkcí, které zajistí obsluhu funkcí implementovaných v C++. Funkce jsou vázané na knihovnu zkompilovanou do souboru DLL.

V adresáři `C#/DllWrapper` nalezneme projekt *DllWrapper*, kde v souboru **main.cpp** jsou definice funkcí se stejnou hlavičkou, jako v souboru **ParFiltDesign.cs**. Funkce jsou však napsané v C++ a mohou tak plně využívat algoritmus pro návrh filtru. Spuštěním projektu v konfiguraci *Release* se vytvoří soubor **ParFiltDllWrapper.dll**. Pro správný chod aplikace je nutné, aby byl se tento soubor ve stejném adresáři.



Obr. 7.1: Grafické uživatelské rozhraní aplikace *Fitting GUI* (WPF).

7.2 Optimalizace parametrů návrhu filtru

Kapitola 5.2 je věnovaná popisu použitého algoritmu PSO pro výpočet optimálních parametrů pro návrh filtru. Implementace algoritmu je napsaná v prostředí MATLAB a zdrojové soubory jsou v adresáři `MATLAB/PSO`. Samotný algoritmus je ve v souboru **parfilt-PSO.m** a nastavení a popis parametrů je k nalezení např. v souboru **pso_engl.m**.

Samotný algoritmus PSO při použité topologii paralelizovat nelze, avšak lze paralelně vypočítat několik návrhů současně při využití sady nástrojů *Parallel Computing Toolbox*. Paralelní výpočet několika návrhů současně bude použitý později při analýze úspěšnosti návrhů.

7.3 VST plug-in simulující paralelní filtr

Pro simulaci navrženého filtru v reálném čase byl navržen plug-in *ParFiltSimulation* jako VST3 zásuvný modul s využitím frameworku JUCE (`https://juce.com/`). Disponuje jednoduchým grafickým uživatelským rozhraním (viz obr. 7.2), které umožňuje načíst vybrané soubory s koeficienty a poté aktivovat filtr.

Podporovaný datový formát je CSV, data ve stejné struktuře jako vyexportuje aplikace pro návrh číslicového filtru v MATLABu nebo v C#. Po načtení a validaci dat je možné aktivovat filtr tlačítkem *Set Filter*. Více informací poskytne informační tlačítko. Hotový modul je nalezení v `Output/VST3`.

Implementace paralelních sekcí má strukturu první kanonické formy (viz obr. 1.2) bez koeficientu b_2 a při použití vhodného kompilátoru jsou některé výpočty paralelizovány SIMD instrukcemi.

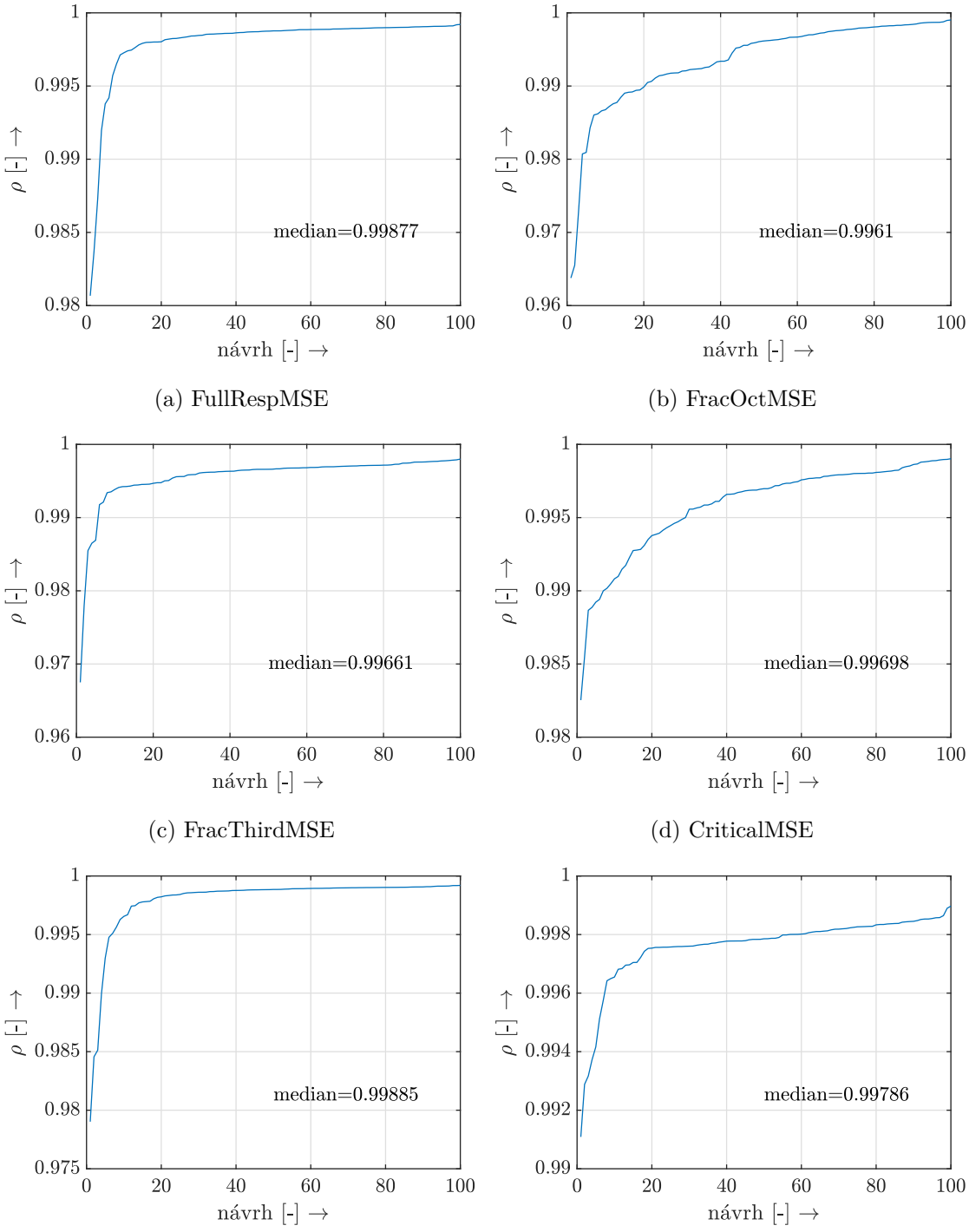
Obr. 7.2: Grafické uživatelské rozhraní VST3 zásuvného modulu pro simulaci navrženého paralelního filtru.

7.4 Výsledky návrhů filtrů

Pro znázornění přesnosti návrhů filtru pomocí algoritmu PSO slouží soubor **Analysis.m** ve složce `MATLAB/PSO`. Pro 4 zadané kmitočtové charakteristiky (Engl, Marshall, Numark, Fabia) se vypočítá 100 návrhů s použitím postupně všech fitness funkcí popsaných v kapitole 5.3. Výsledná hodnota fitness a kmitočtová charakteristika filtru se uloží do souboru s výsledky.

Pro jednotné vyhodnocení všech výsledků byla zvolena fitness funkce využívající Pearsonův korelační koeficient, která nejlépe vypovídá o podobnosti dvou průběhů. Všechny navržené kmitočtové charakteristiky se tímto způsobem porovnají se zadáním a seřazené výsledky budou vyneseny do grafu.

Na obr. 7.3 až 7.6 jsou vidět výsledky jednotlivých zařízení s vypsanou hodnotou mediánu průběhu. Porovnáme-li mediány v rámci jednoho zařízení, jako nejúspěšnější vychází fitness funkce vyhodnocující korelační koeficient kmitočtových charakteristik, naopak nejméně úspěšné je vyhodnocení MSE v oktávových pásmech.



(a) FullRespMSE

(b) FracOctMSE

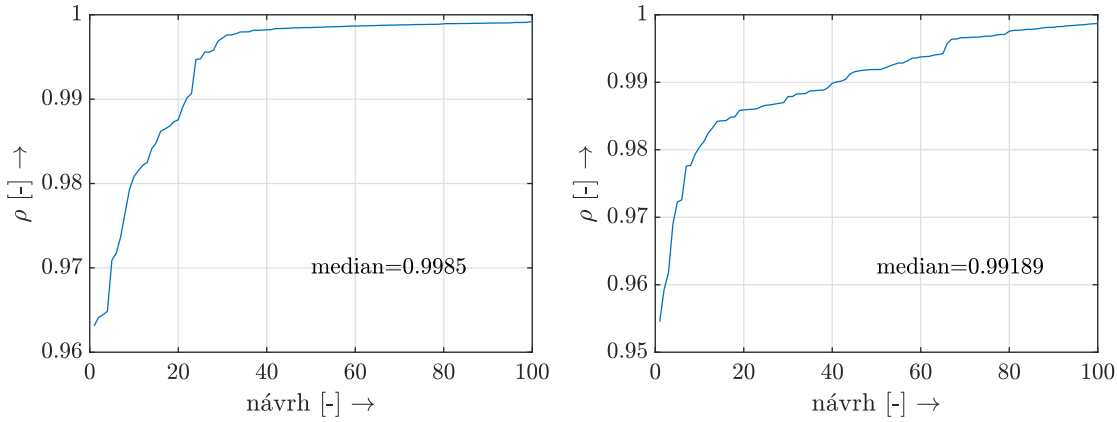
(c) FracThirdMSE

(d) CriticalMSE

(e) Pearson

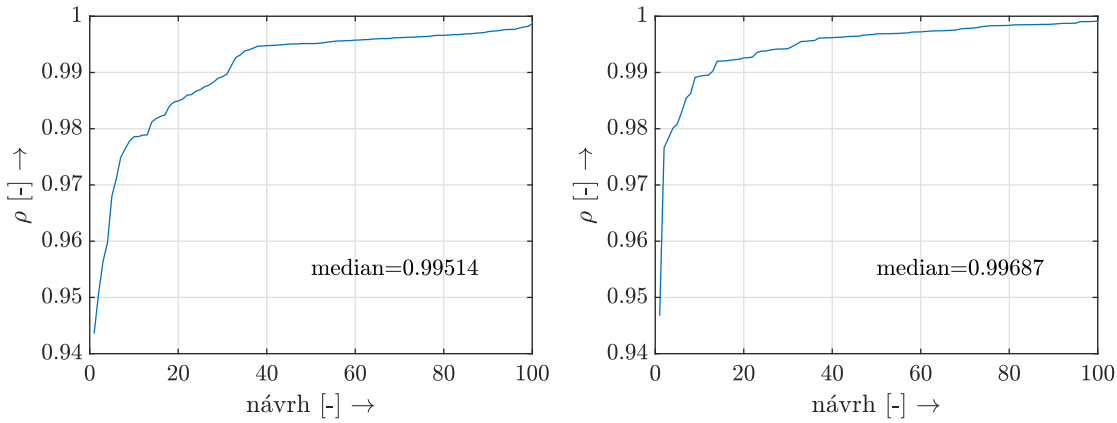
(f) SpectDev

Obr. 7.3: Výsledky návrhů filtrů pomocí PSO – Engl.



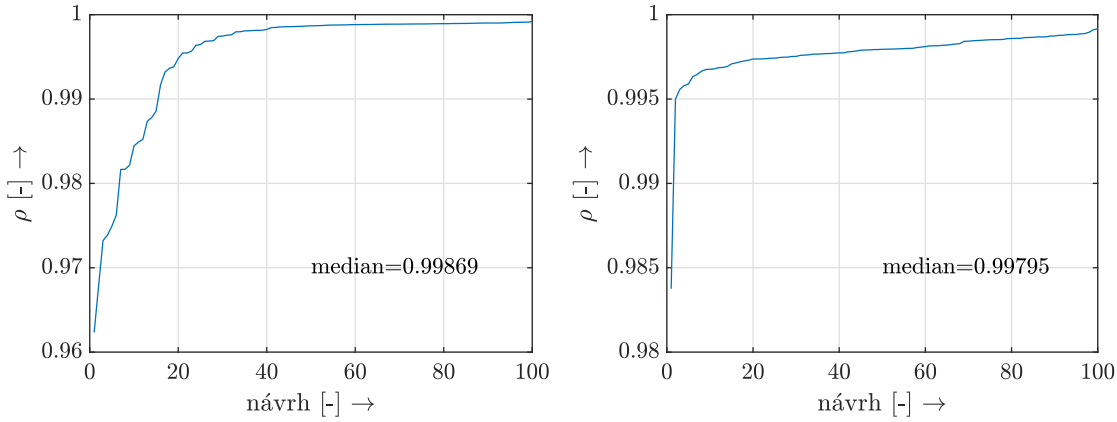
(a) FullRespMSE

(b) FracOctMSE



(c) FracThirdMSE

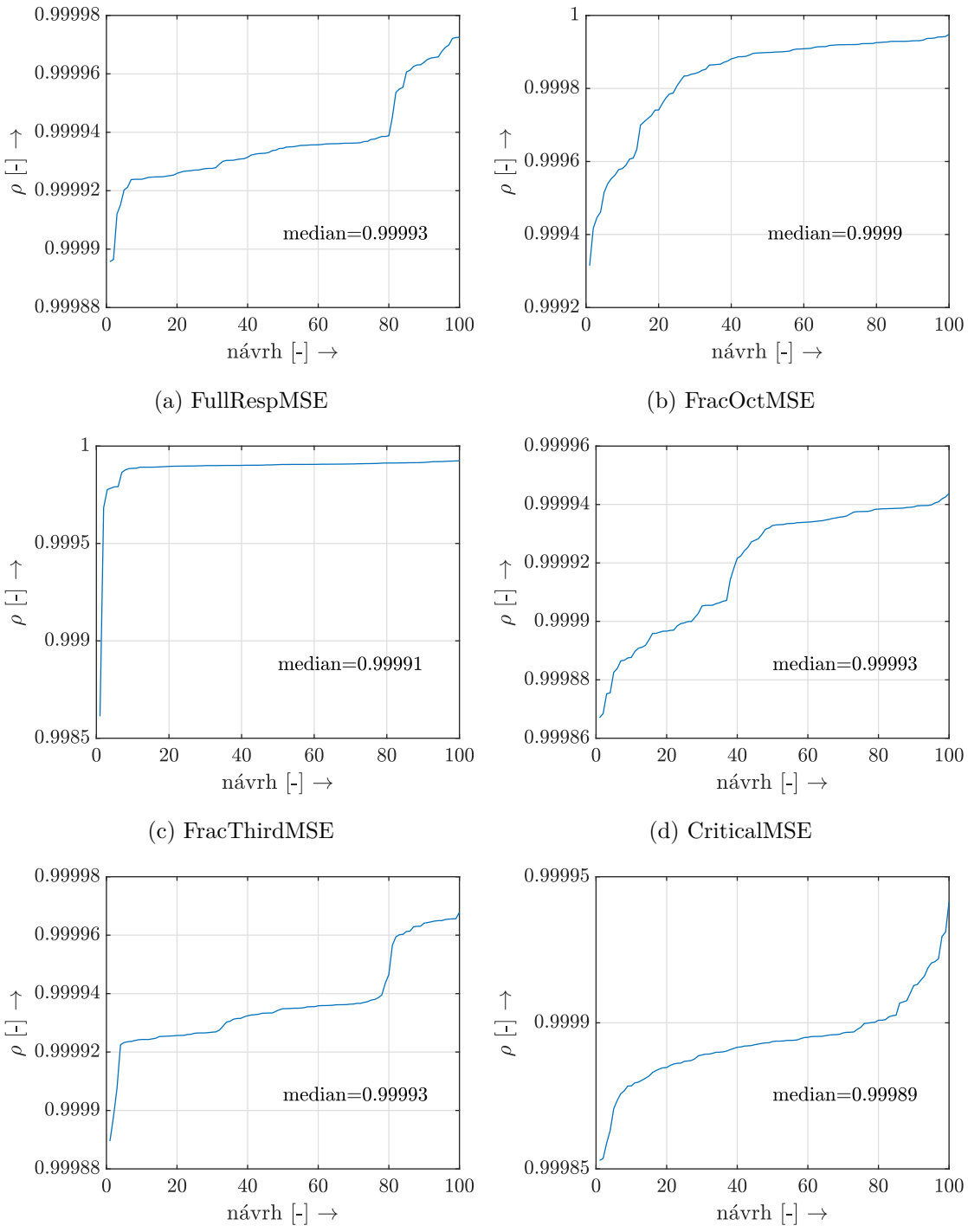
(d) CriticalMSE



(e) Pearson

(f) SpectDev

Obr. 7.4: Výsledky návrhů filtrů pomocí PSO – Marshall.



(a) FullRespMSE

(b) FracOctMSE

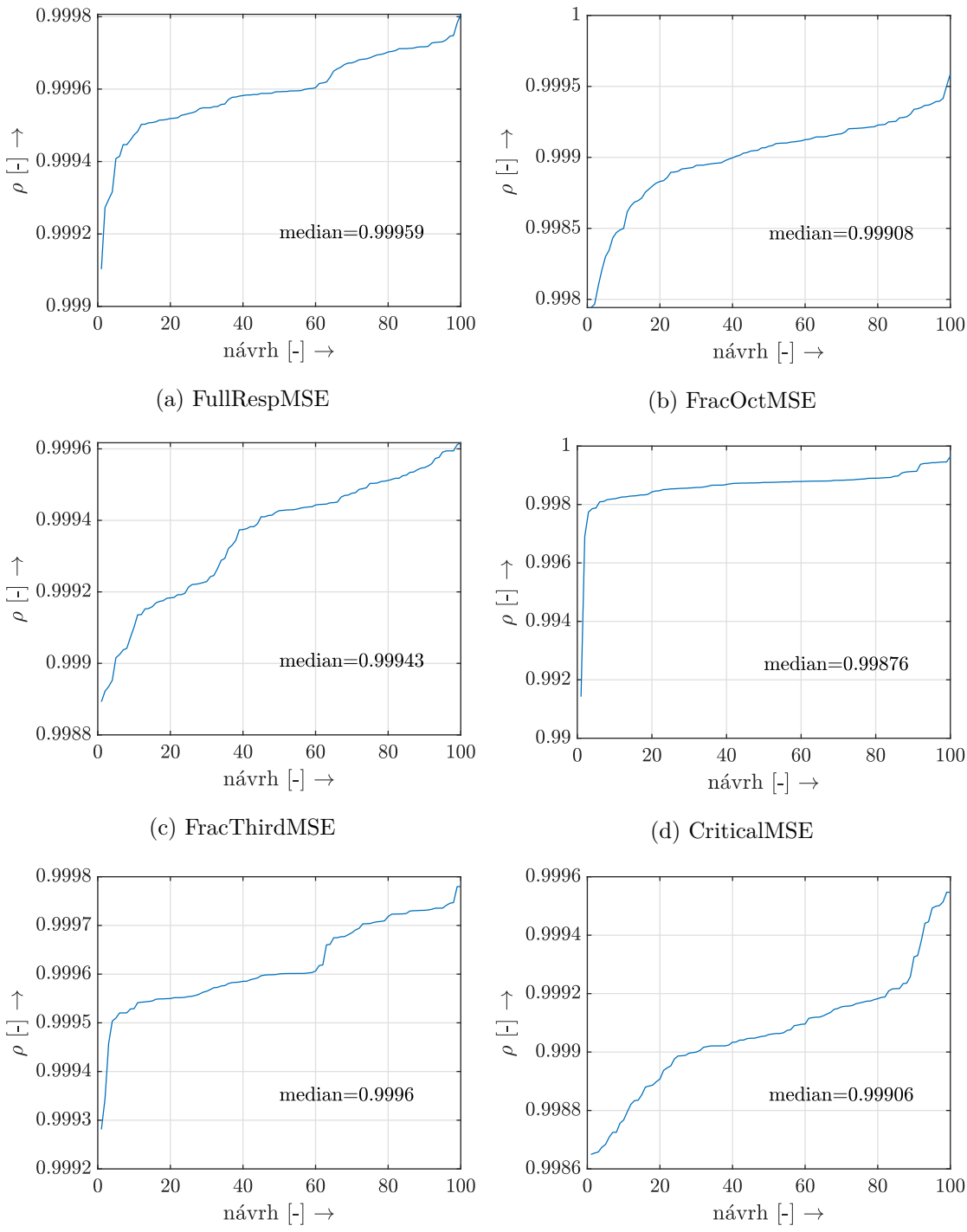
(c) FracThirdMSE

(d) CriticalMSE

(e) Pearson

(f) SpectDev

Obr. 7.5: Výsledky návrhů filtrů pomocí PSO – Numark.



(a) FullRespMSE

(b) FracOctMSE

(c) FracThirdMSE

(d) CriticalMSE

(e) Pearson

(f) SpectDev

Obr. 7.6: Výsledky návrhů filtrů pomocí PSO – Fabia.

ZÁVĚR

Tato práce se zabývá problematikou návrhu číslicových filtrů a modelováním požadované kmitočtové charakteristiky. Na počátku byly zopakovány klasické metody návrhů číslicových filtrů a poté složitější návrhy, které tento problém řeší numericky, přeurčenou soustavou rovnic. Jádrem návrhu číslicového filtru použitého v této práci jsou právě tyto metody.

Návrh filtru probíhá v kmitočtové rovině a se zborcenou kmitočtovou osou, čímž lze návrh optimalizovat s nelineárním kmitočtovým rozlišením, což se daleko více přibližuje způsobu, jakým člověk vnímá zvuk. Oproti klasickým metodám návrhu FIR a IIR filtrů lze při tomto postupu dosáhnout uspokojivých výsledků s použitím filtrů výrazně nižšího řádu.

Struktura filtru pro modelování zadané kmitočtové charakteristiky použitá v této práci je paralelní kombinace IIR sekcí druhého řádu. Tato struktura disponuje nízkým procesním zpožděním a možnou paralelizací výpočtu výstupního signálu, čímž se stává zajímavou pro algoritmy pracující v reálném čase. Jsou ukázány metody výpočtu koeficientů přenosových funkcí jednotlivých sekcí v časové i kmitočtové rovině.

Funkce, které zajistí výpočet těchto koeficientů byly implementovány v C++. Tyto funkce jsou jádrem několika vytvořených aplikací, které slouží k návrhu filtru. Pro porovnání byly funkce implementované i v jazyce Python a jsou k dispozici i zdrojové kódy v prostředí MATLAB navržené Balázsem Bankem a dostupné z [16].

V kapitole 6.5 jsou porovnány výsledky návrhů všech implementací. Podmínkou je dopředná definice pozic pólů filtru, kterou následuje výpočet nulových bodů v jednotlivých paralelních sekcích. Jsou předvedeny varianty ručního zvolení pozic pólů logaritmickým rozmístěním a automatický výpočet poloh pólů metodou Steiglitz-McBride [18] při jednoduchém a dvojitém borcení kmitočtové osy (*jednoduchý, dvojitý warping*).

Výsledky ukazují metodu *dvojitého warpingu* jako nejpřesnější a nejflexibilnější. Pro návrh filtru tímto způsobem je třeba stanovit několik vstupních parametrů – koeficienty borcení kmitočtové osy λ, dělicí kmitočet a délku překryvu a počet pólů v jednotlivých pásmech. Volba těchto parametrů výrazně ovlivňuje výsledný návrh.

Takto se stává výpočet ideálních parametrů multidimenzionálním problémem, pro jehož řešení byla vybrána numerická iterativní metoda *optimalizace hejnem částic* [20, 22, 23]. Algoritmus pomocí bodů představujících jednotlivá řešení prozkoumává N-rozměrný parametrový prostor a snaží se nalézt globální optimum. V kapitole 5.2 je detailní popis tohoto algoritmu a v kap. 6.6 a 7.4 prezentace výsledků.

Zásadní částí této metody je vyhodnocení přesnosti daného návrhu při právě použitých parametrech. Je navrženo celkem šest metod, jak objektivně porovnat kmitočtové charakteristiky a určit tak přesnost návrhu filtru (tzv. fitness funkce). Pro návrh je také nezbytnou součástí váhování rozdílů jednotlivých charakteristik. Jsou představeny metody výpočtu váhovacího vektoru podle kmitočtu, využívající funkci prahu slyšení, nebo podle hodnoty charakteristiky v daném bodě.

Výsledný algoritmus PSO dokáže během několika sekund vypočítat optimální parametry pro dané zadání, rychlost se odvíjí od řádu filtru, délky vstupní kmitočtové charakteristiky a od délky interpolované charakteristiky při použití Hilbertovy transformace.

Při vyhodnocení přesnosti návrhu filtru se nejvíce osvědčila fitness funkce využívající Pearsonův korelační koeficient, hned za ní výpočet MSE ze všech bodů specifikace kmitočtové charakteristiky.

Výsledkem této práce je knihovna C++ funkcí, která zajišťuje výpočet parametrů paralelního filtru. Na tento kód je navázán wrapper pro C# a pro MATLAB, takže je možné jej využívat na více platformách. Algoritmus napsaný v prostředí MATLAB je stále sice asi 1,4× rychlejší, ale nyní je možné využít tento algoritmus i bez vazby na prostředí MATLAB.

V přiložených souborech jsou k nalezení dvě analogické aplikace vytvořené v prostředí MATLAB a pomocí WPF, které umožňují prozkoumat vliv jednotlivých parametrů na návrh filtru a lze pomocí nich také vyexportovat koeficienty pro navržený filtr. Ty lze například využít v přiloženém VST modulu, který umožňuje vytvořený filtr simulovat v reálném čase. Potenciál této možnosti modelování kmitočtové charakteristiky je velký, dá se uplatnit při simulaci zvukových vlastností různých zařízení – sluchátka, reproduktory, mikrofony i charakter těla hudebního nástroje. Přednost při zpracování v reálném čase je možná paralelizace výpočtu, a tedy snížení procesního zpoždění.

Automatické nalezení parametrů pro návrh filtru pomocí numerické iterativní metody má relativně vysokou úspěšnost, avšak doba výpočtu není optimální. Věřím, že v dnešní době, kdy je velké množství složitějších problémů řešitelné pomocí strojového učení, by bylo možné tento problém řešit některou z moderních metod, např. regresním modelem, nebo neuronovou sítí, a dosáhnout tak větší efektivity. Tato oblast ale zatím čeká na probádání.

LITERATURA

[1] BALÍK, Miroslav. *Číslicové zpracování akustických signálů*. Vysoké učení technické v Brně: elektronicky, 2013.

[2] SMÉKAL, Zdeněk. *Číslicové zpracování signálů*. Vysoké učení technické v Brně: elektronicky, 2012. ISBN 978-80-214-4639-7.

[3] VÍCH, Robert a Zdeněk SMÉKAL. *Číslicové filtry*. Praha: Academia, 2000. Česká matice technická (Academia). ISBN 80-200-0761-X.

[4] SYSEL, Petr a Zdeněk SMÉKAL. *Číslicové filtry*. Vysoké učení technické v Brně: elektronicky, 2012. ISBN 978-80-214-4454-6.

[5] JACKSON, Leland B. *Digital filters and signal processing: with MATLAB exercises*. 3rd ed. Boston: Kluwer Academic Publishers, 1996. ISBN 07-923-9559-X.

[6] JAN, Jiří. *Číslicová filtrace, analýza a restaurace signálů*. 2. Vysoké učení technické v Brně: VUTIUM, 2002. ISBN 80-214-2911-9.

[7] KARJALAINEN, M., A. HARMA, U. K. LAINE a J. HUOPANIEMI. Warped filters and their audio applications. *Proceedings of 1997 Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, 1997, 4–. DOI: 10.1109/ASPAA.1997.625615. ISBN 0-7803-3908-8. Dostupné také z: `http://ieeexplore.ieee.org/document/625615/`.

[8] KARJALAINEN, M., A. HARMA a U. K. LAINE. Realizable warped IIR filters and their properties. *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE Comput. Soc. Press, 1997, **3**, 2205–2208. DOI: 10.1109/ICASSP.1997.599488. ISBN 0-8186-7919-0.

[9] KARJALAINEN, Matti, Esa PIIRILÄ, Antti JÄRVINEN a Jyri HUOPANIEMI. Comparison of Loudspeaker Equalization Methods Based on DSP Techniques. *J. Audio Eng. Soc.*, 1999, **47**(1/2), 14–31.

[10] FAJMON, Břetislav a Irena RŮŽIČKOVÁ. *Matematika 3*. Vysoké učení technické v Brně.

[11] Wikipedia contributors. Orthogonality principle. In: *Wikipedia, The Free Encyclopedia* [online]. 2017 [cit. 2019-12-12]. Dostupné z: `https://en.wikipedia.org/w/index.php?title=Orthogonality_principle&oldid=772360559`.

[12] STEIGLITZ, K. a L. E. MCBRIDE. A Technique for the Identification of Linear Systems. *IEEE Transactions on Automatic Control*. 1965, **10**(4), 461–464.

[13] BANK, Balázs. Perceptually Motivated Audio Equalization Using Fixed-Pole Parallel Second-Order Filters. *IEEE Signal Process. Lett.* 2008, **15**, 477–480. Dostupné

také z: https://www.researchgate.net/publication/224315102_Perceptually_Motivated_Audio_Equalization_Using_Fixed-Pole_Parallel_Second-Order_Filters/.

[14] BANK, Balázs. Direct design of parallel second-order filters for instrument body modeling. *International Computer Music Conference Proceedings*. Copenhagen, 2007, **1**, 458–465.

[15] BANK, Balázs. Logarithmic Frequency Scale Parallel Filter Design with Complex and Magnitude-Only Specifications. *IEEE Signal Processing Letters*. 2011, **18**(2), 138–141.

[16] BANK, Balázs. *Transfer function modeling and equalization by fixed-pole parallel filters* [online]. [cit. 2019-12-17]. Dostupné z: http://home.mit.bme.hu/~bank/parfilt/.

[17] BANK, Balázs a Germán RAMOS. Improved Pole Positioning for Parallel Filters Based on Spectral Smoothing and Multiband Warping. *IEEE Signal Processing Letters*. 2011, **18**(5), 299–302.

[18] JACKSON, L. B. Frequency-domain Steiglitz-McBride method for least-squares IIR filter design, ARMA modeling, and periodogram smoothing. *IEEE Signal Processing Letters*. 2008, **15**, 49–52. ISSN 1070-9908.

[19] WATERS, Michael a Mark B. SANDLER. Least Squares IIR Filter Design on a Logarithmic Frequency Scale. Pr*oc. IEEE Int. Symp. on Circuits and Systems*. 1993, 635–638.

[20] KENNEDY, J. a R. EBERHART. Particle swarm optimization. Proceedings of ICNN'95 – International Conference on Neural Networks. IEEE, 1995, 1942–1948. ISBN 0-7803-2768-3. Dostupné také z: http://ieeexplore.ieee.org/document/488968/.

[21] PÁNEK, Ondřej. *Algoritmus optimalizace hejnem částic: vývoj a jeho aplikace*. Praha, 2018. Bakalářská práce. České vysoké učení technické v Praze, Fakulta jaderná a fyzikálně inženýrská, Katedra softwarového inženýrství. Vedoucí práce Ing. Quang Van Tran, Ph.D.

[22] CLERC, Maurice. *Standard Particle Swarm Optimisation*. 2012. hal-00764996. Dostupné také z: https://hal.archives-ouvertes.fr/hal-00764996.

[23] SAHU, Amaresh, Sushanta KUMAR PANIGRAHI, a Sabyasachi PATTNAIK. Fast Convergence Particle Swarm Optimization for Functions Optimization. *Procedia Technology*. 2014, **4**, 319-324. ISSN 2212-0173. Dostupné také z: http://www.sciencedirect.com/science/article/pii/S2212017312003271.

[24] SMITH, J. O. Minimum-Phase Filter Design. *Spectral Audio Signal Processing* [online]. 2011 [cit. 2020-05-25]. ISBN 978-0-9745607-3-1. Dostupné z: https://ccrma.stanford.edu/~jos/sasp/Minimum_Phase_Filter_Design.html.

[25] IEC 61260-1:2014. *Electroacoustics – Octave-band and fractional-octave-band filters – Part 1: Specifications*. International Electrotechnical Commision, 2014.

[26] ZWICKER, Eberhard a Hugo FASTL. *Psychoacoustics: Facts and Models*. 3. New York: Springer-Verlag Berlin Heidelberg, 2007. ISBN 3-540-65063-6.

[27] RYDER, Simon A. Methods for comparing frequency response analysis measurements. *Conference Record of the the 2002 IEEE International Symposium on Electrical Insulation (Cat. No.02CH37316)*. Boston, 2002, 187–190.

[28] Solve systems of linear equations Ax = B for x – MATLAB mldivide \. *MathWorks – Makers for MATLAB and Simulink* [online]. The MathWorks Inc., 2020 [cit. 2020-04-27]. Dostupné z: `https://www.mathworks.com/help/matlab/ref/mldivide.html`.

[29] 1-D data interpolation (table lookup) - MATLAB interp1. *MathWorks - Makers for MATLAB and Simulink* [online]. The MathWorks Inc. , 2020 [cit. 2020-05-24]. Dostupné z: `https://www.mathworks.com/help/matlab/ref/interp1.html`.

[30] Chapter 3 – Piecewise Polynomial Inteprolation. VAN LOAN, Charles F. *Introduction to Scientific Computing: A Matrix-vector Approach Using MATLAB*. 2nd. Prentice-Hall, 2000. ISBN 0-13-949157-0.

[31] CVLBook. *Home | Department of Computer Science* [online]. 2020 [cit. 2020-04-30]. Dostupné z: `http://www.cs.cornell.edu/courses/cs4210/2015fa/CVLBook/new_page_1.htm`.

[32] TERHARDT, Ernst. Calculating virtual pitch. *Hearing Research*. 1979, **1**(2), 155–182. DOI: 10.1016/0378-5955(79)90025-X. ISSN 03785955. Dostupné také z: `https://linkinghub.elsevier.com/retrieve/pii/037859557990025X`.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

API	Rozhraní aplikace (*Application Programming Interface*)
AR	*Autoregressive*
ARMA	*Autoregressive Moving Average*
CSV	*Comma-separated values*
DLL	Dynamicky linkovaná knihovna (*Dynamic Linking Library)*
DP	Dolní propust
FFT	Rychlá Fourierova transformace (*Fast Fourier Transform*)
FIR	Konečná impulsová odezva (*Finite Impulse Response*)
GUI	Grafické uživatelské rozhraní (*Graphical User Interface*)
IIR	Nekonečná impulsová odezva (*Infinite Impulse Response*)
HP	Horní propust
LSE	*Least Squares Error*
LTI	Lineární časově invariantní (*Linear Time Invariant*)
MA	*Moving Average*
MSE	Střední kvadratická odchylka (*Mean squared error*)
PP	Pásmová propust
PSO	Optimalizace hejnem částic (*Particle Swarm Optimization*)
PZ	Pásmová zádrž
SIMD	*Single Instruction, Multiple Data* – typ instrukce
WPF	*Windows Presentation Foundation* – knihovna pro tvorbu GUI, součást .NET frameworku

SEZNAM PŘÍLOH

A POPIS SOUBORŮ IMPLEMENTACE NÁVRHU FILTRU V C++

V adresáři `C++/ParallelFilters/ParallelFilters` se nachází zdrojové soubory k projektu, který implementuje návrh paralelního filtru. Účelem projektu je ladění vytvořených funkcí, které jsou postupně při návrhu volány.

Projekt lze sestavit a spustit. V hlavním projektovém souboru **main.cpp** jsou v kódu nastaveny parametry návrhu (počet pólů, parametry λ apod.). Zadáním názvu souboru zdrojových dat do konzolového okna (např. „Engl“) proběhne návrh paralelního filtru metodou dvojitě logaritmicky rozmístěných pólů, jednoduchého a dvojitého borcení kmitočtové osy. Zdrojová data se nacházejí v adresáři projektu ve složce `input_data_csv`, výsledné koeficienty filtru jsou uloženy do formátu CSV do složky `coeffs_data`.

Zdrojové soubory však své uplatnění nachází na více místech, jejich kopie se nachází v adresáři `MATLAB/mex`, kde jsou zkompilovány do mex funkce pro prostředí MATLAB, a na soubory se odkazuje i v projektu ve složce `C#/DllWrapper`, kde se funkce zkompilují do dynamické knihovny DLL. Potom je lze využívat v externích aplikacích.

Struktura je rozdělena do několika souborů a jmenných prostor (*namespace*), které budou popsány níže.

ParallelFilters

Soubory **ParallelFilterDesign.h** a **ParallelFilterDesign.cpp** obsahují implementaci zásadních funkcí ve jmenném prostoru `ParallelFilters`, které realizují návrh filtru.

- `freqpoles` – realizuje transformaci zadaných kmitočtů na póly podle rovnic (4.11), (4.12) a (4.13).
- `warpPolesFr` – realizuje výpočet pólů dle se zborcenou kmitočtovou osou dle parametru λ na míru zadané kmitočtové charakteristice.
- `dualWarpPolesFr` – realizuje výpočet pólů *dvojitým warpingem*. Jádrem obou funkcí je funkce `lsidFr`.
- `lsidFr` – funkce navrhne AR filtr pro zadanou kmitočtovou charakteristiku. Jádro funkce je Kallmanova pseudokorelační metoda popsaná v kapitole 3.3.2 na kterou navazuje Steiglitz-McBride iterační metoda popsaná v té samé kapitole.
- `parFiltDesignFr` – realizuje návrh filtru v kmitočtové rovině přesně podle způsobu popsaným v 4.3.1 a jejím výstupem jsou koeficienty čitatele a jmenovatele přenosové funkce filtru a koeficienty FIR části modelu.
- `solve` – funkcionalita odpovídá operátoru „\“ v MATLABu – dělení rovnice zleva. Vzhledem k vlastnostem matic je použitý algoritmus LDL.
- `minPhaseN` – funkce vytvoří z modulové kmitočtové charakteristiky komplexní kmitočtovou charakteristiku Hilbertovou transformací.

VectorExtensions

Soubor **VectorExtensions.h** obsahuje jmenný prostor `VectorExtensions`, ve kterém jsou definovány pomocné a zjednodušující funkce pro práci s vektory a některé užitečné funkce odpovídající funkcím MATLABu (`log`, `exp`, `real`, `imag`, `find`). Využití nalézají takřka ve všech sofistikovanějších funkcích.

Extensions

Soubory **Extensions.h** a **Extensions.cpp** obsahují implementaci řady funkcí, které doplňují funkcionalitu hlavního jmenného prostoru. Nalezneme zde funkce pro práci se soubory využívající knihovnu *Rapidcsv*[1], několik funkcí nahrazující interní funkce prostředí MATLAB a také funkce realizující Fourierovu transformaci. Ty čerpají z knihovny *Simple-FFT*[2]. Také se zde nachází implementace funkcí realizujících kubickou interpolaci využívající volně dostupnou C++ knihovnu[3].

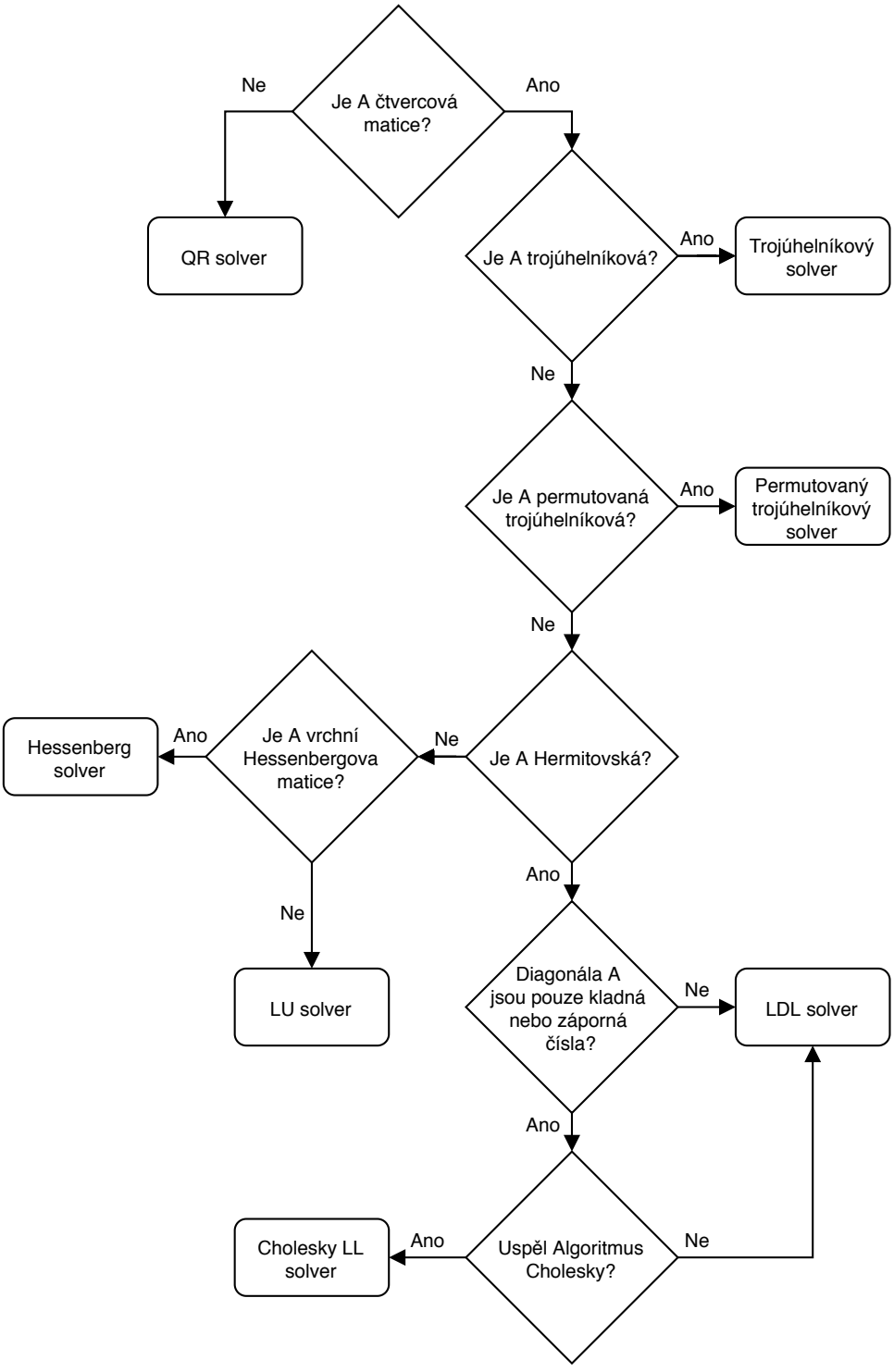
A.1 Použité knihovny a algoritmy

Protože zásadní operace v jednotlivých krocích návrhu filtru jsou maticové operace, je třeba použít vhodnou knihovnu, která bude práci s maticemi usnadňovat. Zvolil jsem knihovnu *Eigen*[4], která zaručuje velkou podporu pro práci s maticemi i vektory, její implementace a rozhraní je jednoduché a její použití není vázáno licencí.

Pro řešení soustavy rovnic nabízí knihovna podobně jako MATLAB několik algoritmů, včetně *Cholesky* – LL nebo LDL.

[1] Rapidcsv – `https://github.com/d99kris/rapidcsv`
[2] Simple-FFT – `https://github.com/d1vanov/Simple-FFT`
[3] Spline knihovna dostupná z `https://github.com/ttk592/spline/`
[4] Eigen – `http://eigen.tuxfamily.org/`

B DIAGRAM PRO ZVOLENÍ ŘEŠITELE SOUSTAVY ROVNIC – MATLAB



Obr. B.1: Diagram pro zvolení vhodného algoritmu řešení soustavy rovnic [28].

C OBSAH PŘILOŽENÉHO MÉDIA

Na přiloženém médiu naleznete vytvořené algoritmy v různých implementacích a k nim příklady použití.

Pro projekty v jazyce C# a C++ bylo použito vývojové prostředí Visual Studio 2019. Projekt v C# využívá rozhraní .NET Framework 4.7.2 a projekty v jazyce C++ jsou psané ve standardu C++14.

Pro testování kódu v jazyce MATLAB byla použita verze MATLAB R2018b.

Pro kód napsaný v jazyce Python byla použitá verze 3.8.2.

Stejný obsah je k nalezení na mém veřejném repozitáři na adrese `https://github.com/TheMates/Diploma-thesis-public`.

- . . . . . kořenový adresář přiloženého média
- `C`
  - `DllWrapper` . . . . . složka projektu pro wrapper C++/C#
  - `FittingGUI` . . . . . složka projektu WPF *Fitting GUI*
  - `packages` . . . . . složka použitých balíčků
  - `WPFFitting.sln`
- `C++`
  - `ParallelFilters` . . . . . složka projektu obsahující implementaci algoritmu v C++
  - `ParFiltSimulation` . . . . . složka projektu v JUCE pro VST plug-in
- `MATLAB`
  - `common_code` . . . . . společný kód
  - `devices data` . . . . . složka se zdrojovými daty kmitočtových charakteristik
  - `fittingGUI` . . . . . složka projektu pro aplikaci *fittingGUI*
  - `mex` . . . . . soubory pro wrapper C++/MATLAB
  - `PSO` . . . . . složka obsahující algoritmus PSO
  - `response extrapolation` . . . . . příklady prodloužení kmitočtových charakteristik
- `Output`
  - `MATLAB` . . . . . obsahuje zkompilovanou aplikaci *Fitting GUI*
  - `VST3` . . . . . *ParFiltSimulation* plug-in
  - `WPF` . . . . . obsahuje zkompilovanou aplikaci *Fitting GUI*
- `Python` . . . . . algoritmus implementovaný v Pythonu
- `Text` . . . . . text práce