ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Mobilní aplikace pro transkripci zvukového záznamu do notového zápisu
Student:	Šimon Lomič
Vedoucí:	Mgr. Petr Matyáš
Studijní program:	Informatika
Studijní obor:	Teoretická informatika
Katedra:	Katedra teoretické informatiky
Platnost zadání:	Do konce zimního semestru 2017/18

Pokyny pro vypracování

- Zmapujte a nastudujte algoritmy sloužící pro transkripci monofonních melodií ze zvukového záznamu, zaměřte se na analýzu výšky tónu, tempa, tóniny a délek not.
- Definujte vhodné metriky pro výběr algoritmů, aby byly použitelné v mobilní aplikaci pro operační systém iOS a dosáhly nejlepší kvality transkripce. Vyberte vhodné algoritmy na základě testování na těchto metrikách.
- Implementujte algoritmy v mobilní aplikaci pro iOS.
- Své řešení důkladně otestujte. K testování použijte synteticky generované zvuky pomocí MIDI a reálné nahrávky s referenčními MIDI transkripcemi z datasetů UIOWA Musical Instrument Samples či MAPS.

Seznam odborné literatury

[1] MCLEOD, Philip, 2008. Fast, Accurate Pitch Detection Tools for Music Analysis. ISBN 406372628.
[2] GOVREEN-SEGAL, Dael, Michal M RADAI, Yakov SIVAN and Shimon ABBOUD, 1999. Real-Time PC-Based System for Dynamic Beat-to-Beat QT-RR Analysis. Computers and Biomedical Research [online]. vol. 32, no. 4, pp. 336–354. Retrieved z: doi:10.1006/cbmr.1999.1514.
[3] TEMPERLEY, David, 2002. A Bayesian Approach to Key-Finding. In: Computer Music Modeling and Retrieval. Berlin, Heidelberg: Springer Berlin Heidelberg, Lecture Notes in Computer Science, pp. 195–206. ISBN 978-3-540-44145-8.

doc. Ing. Jan Janoušek, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 23. února 2016

České vysoké učení technické v Praze

Fakulta informačních technologií

Katedra Teoretické Informatiky

Bakalářská práce

Mobilní aplikace pro transkripci zvukového záznamu do notového zápisu

Šimon Lomič

Vedoucí práce: Mgr. Petr Matyáš

16. května 2017

Poděkování

Děkuji svému vedoucímu, panu Mgr. Petru Matyášovi, za možnost zpracovávat tuto práci pod jeho vedením. Dále děkuji všem svým blízkým, zejména své přítelkyni a rodině za podporu, pomoc a cenné rady.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 16. května 2017 .....

České vysoké učení technické v Praze

Fakulta informačních technologií

Odkaz na tuto práci

LOMIČ, Šimon. *Mobilní aplikace pro transkripci zvukového záznamu do notového zápisu.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

Tato práce se zabývá řešením problému automatické hudební transkripce. Analyzuje existující algoritmy sloužící k přepisu akustického monofonního signálu do hudební notace a hledá řešení problémů detekce výšky tónu, not, rytmu a tóniny. Tyto algoritmy jsou v práci implementovány a vzájemně porovnávány za pomoci nového testovacího frameworku pro analýzu hudební skladby. Porovnání algoritmů je prováděno na základě pevně definovaných metrik navržených za účelem volby vhodných algoritmů pro implementaci hudební transkripce v mobilní aplikaci. Na základě výsledků srovnání je z nejlepších algoritmů navržen transkripční systém.

Klíčová slova Automatická hudební transkripce, detekce výšky, detekce rytmu, detekce not, detekce tóniny, spektrální analýza

Abstract

This thesis focuses on automatic music transcription. It analyses existing algorithms solving the task of converting the acoustic monophonic signal into musical notation. It achieves so by solving problems of pitch detection, onset and offset detection, rhythm analysis, and key detection. Algorithms that solves these problems are implemented and evaluated. These algorithms are compared by using a new testing framework designed for this purpose. New metrics for the comparison are designed to find the best-suited set of algorithms for usage in a mobile application for music transcription. New transcription system is made based on the results of this evaluation.

Keywords Automatic music transcription, Pitch detection algorithms, Beat detection, Music notation, Key detection

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V moderním světě se hudba stává téměř všudypřítomnou – od kapesních přehrávačů, který dnes vlastní téměř každý mladý člověk, po reprodukovanou hudbu znějící z každého obchodu. Jakým způsobem však člověk vnímá hudbu? Jaké informace si z prostého mechanického vlnění, kterým zvuk je, člověk odnáší?

Představme si člověka poslouchajícího hudbu. Brouká si hlavní melodii, klepe si do rytmu a z jeho tváře lze číst, jakým způsobem na něj skladba působí. Shrňme si tyto informace pod tři základní pojmy: *melodie*, *rytmus* a *harmonie*. Melodií rozumíme sekvenci tónů, která je pro nás hudebně zajímavá a často si ji snadno zapamatujeme. Je úzce provázaná s rytmem, který určuje pravidelnosti a opakující se vzorce v hudební skladbě. Harmonie pak v sobě zahrnuje vzájemné vztahy použitých tónů.

Hudební transkripcí zde rozumíme přepis těchto informací do hudební notace. Tu můžeme chápat jako přesné instrukce pro hudebníka potřebné k reprodukování skladby. Při hudební transkripci tedy provádíme jakousi zpětnou analýzu hudebního signálu, hledáme „recept“, pomocí kterého skladba mohla vzniknout. Pro tento účel lze zvolit libovolnou symbolickou notaci, naším cílem však bude reprodukci skladby co nejvíce usnadnit. Pro většinu hudebníků západní kultury je dnes nejpřístupnější *moderní hudební notace*.

Tvořit hudbu je dnes populárnější než kdy dříve. Hudební transkripce je přitom dosti nelehký úkol vyžadující cvičený relativní sluch, cit pro rytmus a harmonii a pokročilou znalost hudební teorie. Pro vyprodukování zajímavých a hodnotných hudebních nápadů ale většina těchto schopností třeba není, a proto je zde snaha tento úkol co nejvíce zjednodušit.

Otázka, zda je tento náročný proces možné kompletně automatizovat, je řešená již od 70. let minulého století (Piszczalski et al., 1982), kdy byl prvně představen pojem *automatická hudební transkripce*. Pomocí výpočetních modelů a metod zpracovávání signálu se snažíme dosáhnout přepisu do hudební notace tak, jak by to udělal člověk. V podstatě je tedy nutné naučit počítač „rozumět" hudbě, tedy vytvořit model lidského vnímání hudby. Problémem je, že současné psychoakustické poznatky nestačí k tomu, abychom dokázali vnímání zvuku v mozku zcela pochopit a napodobit. Jedním z důvodů je, že lidská mysl dokáže rozpoznat velmi drobné změny v hudebním signálu a sledovat jejich strukturu. Přitom je však zároveň schopna tolerovat velké nepřesnosti a výkyvy od dané struktury a interpretovat ji tak, jak bylo zamýšleno. Proto se stávající modely dokonalému výsledku jen přibližují a ve složitějších případech si často pomáháme metodami strojového učení.

Automatickou hudební transkripci lze chápat jako analogii k problému automatického rozpoznávání řeči. Oproti převodu mluvené řeči do textu převádíme hudební nahrávku do hudebního písma. Obě úlohy jsou velmi komplikované. Protože je však o analýzu řeči mnohem větší komerční zájem, přitáhla daleko více pozornosti a obor rozpoznání hudby je oproti oboru rozpoznání řeči výrazně opožděn (Klapuri, 2006).

Hudební transkripce má kromě přímé aplikace k získání hudební notace pro reprodukci hudby i další praktická využití. Jelikož je akustický signál nejběžnější způsob přenosu hudby, je k dispozici nezměrné množství záznamů, pro které máme jen málo informací o jejich struktuře. Provedení automatické hudební transkripce by nám usnadnilo vyhledávání v těchto záznamech a jejich indexaci. Na výsledné transkripci by se též daly uplatnit metody vytěžování

znalostí z dat, které by pomohly různým muzikologickým odvětvím. Hudební notace navíc poskytuje hierarchickou strukturu nad časovými intervaly hudebního záznamu, které mají podobné parametry. Tato znalost může pomoci mnohým kompresním algoritmům k dosažení vyšší úrovně komprese. Mnohé interaktivní hudební systémy umožňují z akustického vstupu doplněné notací vygenerovat automatický doprovod.

Cíle práce

Hlavním cílem této práce je zmapovat algoritmy pro transkripci monofonních melodií ze zvukového záznamu. U nich se zaměříme především na problémy detekce výšky tónu a nástupů not, které tvoří stěžejní součást každého transkripčního systému. Dále se budeme věnovat analýze rytmu a tóniny hudební skladby. Dalším cílem bude nalezené algoritmy implementovat a otestovat na vhodných metrikách. Tyto metriky navrhneme tak, aby pomohly zvolit algoritmy použitelné v mobilní aplikaci pro operační systém iOS.

Struktura práce

V první kapitole si popíšeme základní pojmy týkající se hudební transkripce, kterou si zde formálně definujeme. Označíme si ji jako transformaci hudebního signálu do hudební notace a tento vstup a výstup si zde představíme.

Ve druhé kapitole na tyto pojmy navážeme analýzou problému automatické hudební transkripce. Popíšeme si jeho vývoj a současný stav řešení. Poté si tento problém dekomponujeme na menší podproblémy, se kterými se nám bude lépe pracovat.

Hlavnímu z těchto podproblémů, detekci výšky, věnujeme následující kapitolu. V ní se nejprve seznámíme se samotným pojmem výšky tónu a jeho komplikovanou povahou. Ve zbytku kapitoly se budeme věnovat způsobům jak výšku detekovat.

Další kapitola je věnovaná druhému stěžejnímu problému hudební transkripce, kterým je detekce not. V této kapitole se seznámíme s problematikou důkladněji a zmapujeme její možné řešení.

Pátá kapitola se zabývá problémem detekce rytmu. Představíme si v ní kompletní řešení analýzy rytmu za pomocí počítačového programu, který se snaží modelovat způsob, jakým člověk vnímá rytmus.

Šestá kapitola popisuje poslední podproblém hudební transkripce, kterým je detekce tóniny. Definuje základní pojmy problematiky tonální analýzy a představuje algoritmy k jejímu řešení.

V sedmé kapitole si definujeme vhodné metriky pro otestování algoritmů zmiňovaných v této práci. Poté si ukážeme testovací framework, který jsme implementovali společně s těmito algoritmy pro jejich snazší testování. Závěrem této kapitoly budou výsledky testování na zde definovaných metrikách.

Kapitola 1

Základní pojmy

V této kapitole si definujeme základní pojmy, které budeme potřebovat k porozumění problému hudební trankripce. Nejprve si definujeme samotný problém (kap. 1.1), pak se podíváme důkladněji na jeho vstup (hudební signál, kap. 1.2) a na jeho výstup (hudební notace, kap. 1.3).

1.1 Hudební tranksripce

Jak popisuje Moorer [1975], převést hudební záznam zpět do hudební notace, dle které vznikl, je prakticky nemožné. V hudební notaci se vyskytuje mnoho informací, které se při interpretaci ztrácí. Nejsme například schopni odhadnout, zda délka dané noty pochází z noty půlové, či čtvrťové. Pro správné formulování problému potřebujeme přesnou definici transkripce, která je řešitelná.

Heinsworth [2004] definuje hudební transkripci jako „přepis kompletní reprezentace všech hudebních struktur a k nim příslušejících informací vyskytujících se v daném zvukového záznamu“. Tato definice nám neříká přesně v jaké formě máme tyto informace reprezentovat, ani co přesně je onou hudební strukturou, víme jen, že má být kompletní. Jelikož však téměř každá kultura na světě produkuje hudbu trochu odlišnou, je takřka nemožné kompletně popsat všechny její struktury.

Ryynänenova definice [2008] říká, že hudební transkripce je „analýza hudebního signálu za účelem získání parametrické reprezentace not vyskytujících se v signálu“. Tímto se omezujeme pouze na informace o notách. Stále nám však zbývá definovat, jaké parametry má výsledná symbolická notace reprezentovat.

Klapuri [2006] ji definuje jako „transformaci akustického hudebního záznamu do symbolické reprezentace, která nám dá dostatečné informace k reprodukci daného hudebního díla za použití dostupných hudebních nástrojů“. Je však nutné pevně stanovit, jak se může výsledná reprodukce lišit od původního signálu. Nahrávka obsahuje kromě dat formulovatelných v moderní hudební notaci též mnohé další informace. Jedná se např. o změny hlasitosti hraných not a barvy tónu, které je u mnohých nástrojů hráč schopen snadno ovládat. Dále jsou v signálu přítomny i informace o zvukových vlastnostech prostředí, kde byla nahrávka pořízena, např. ruchy, šumy a ozvěna. Ať se tyto informace v signálu vyskytly záměrně či nezáměrně, je třeba definovat, zda se při transkripci vyřadí nebo budou převedeny do nějaké formy symbolické reprezentace. Problém s definicí hudební transkripce tedy spočívá v její závislosti na použité hudební notaci. Některé prvky v nahrávce totiž pro určitý hudební styl můžou být pouze ruch v signálu, zatímco pro jiný styl by se jednalo o požadovaný prvek, který se v jeho notaci musí objevit. Tento fakt bohužel nereflektuje žádná ze zmíněných definic.

Definujme si proto hudební transkripci jako *přepis akustického hudebního signálu do takové symbolické reprezentace, která nám dá dostatečné informace pro vytvoření záznamu, který je s původním záznamem z hlediska použitého notačního systému totožný.*

Definice 1. Nechť $\mathcal{S}$ je množina všech hudebních signálů a $\mathcal{N}_\mathcal{H}$ množina všech notací hudebního stylu $\mathcal{H}$. Pak definujeme *hudební transkripční systém* $\mathcal{T}$ jako dvojici $\mathcal{T} = (t, I)$, kde zobrazení $t : \mathcal{S} \to \mathcal{N}_\mathcal{H}$ odpovídá *transkripci* hudebního signálu na jemu odpovídající notaci a $I \subseteq \mathcal{N}_\mathcal{H} \times S$ je relace *interpretace*, která říká následující: $\forall\, [n, s] \in I$: Notace n lze pomocí nějakého hudebního nástroje interpretovat jako hudební signál s.

Námi definovaná transkripce má následující vlastnosti:

Vlastnost 1 (Idempotence)**.**

$$\forall s_1, s_2 \in \mathcal{S} : t(s_1) \neq t(s_2) \quad \Rightarrow \quad s_1 \neq s_2$$

Tato vlastnost plyne z definice jako základní vlastnost zobrazení t. Říká nám, že dva stejné hudební signály nemohou mít rozdílnou transkripci. V případě randomizovaných algoritmů transkripce by mohlo dojít k porušení této vlastnosti. Není to však žádoucí a v takovém případě bude na místě diskuze o míře porušení.

Vlastnost 2 (Nezávislost interpretace)**.**

$$\forall n \in \mathcal{N}_{\mathcal{H}}, \forall s : [n, s] \in I : \quad \Rightarrow \quad t(s) = n$$

Jinými slovy, pokud provedu transkripci libovolné interpretace dané hudební notace, dostanu zpět tuto notaci. Tato vlastnost pouze omezuje definici interpretace. Protože je běžné, že se interpretace skladeb od různých autorů často liší, je třeba definovat, zda se tyto změny mají promítnout v jejím přepisu. Často jde pouze o drobné změny časování, rozdíly v hlasitosti a barvě tónů. Většina hudebních notací tyto informace neobsahuje, čímž vlastnost nezávislosti splňuje. Pokud však interpret provede větší změny např. v délkách dob, je třeba stanovit práh, od kterého se ve výsledné transkripci změna projeví. Tento problém adresujeme dále ve 2. kapitole při analýze problému transkripce.

Důsledek 1 (Jednoznačnost)**.** Z předchozí vlastnosti vyplývá, že *pokud se transkripce dvou různých interpretací shoduje, musela vycházet ze stejné notace.* Pokud by tomu tak totiž nebylo, nebyla by splněna nezávislost interpretace, která by jeden ze zdrojů přetvořila na ten druhý.

Důsledek 2 (Unikátnost interpretace)**.** Dále z předchozích vlastností plyne, že *interpretace dvou různých notací nemůže nikdy vést ke stejnému hudebnímu signálu.* Pokud by tomu tak bylo a chtěli bychom dodržet vlastnost nezávislosti interpretace, musela by být porušena vlastnost idempotence, jelikož bychom chtěli ze stejného signálu provést dvě různé transkripce.

Výše uvedených vlastností se budeme při návrhu transkripčního systému řídit. Pokud by některá z použitých metod tyto vlastnosti porušovala, budeme se snažit se jí vyvarovat nebo obhájit toto porušení.

1.1.1 Pojem hudební transkripce

Je nutné zmínit, že pojem hudební transkripce má kromě výše zmíněného významu také další. Může též znamenat úpravu či přepracování notového zápisu na nový. Také může značit negativně laděné hodnocení skladby, která převzala nápad ze skladeb jiných autorů. Třetí význam, který je předmětem této práce, je *přepis zvukových nahrávek do hudebního písma*, jemuž odpovídá i naše definice. (Šrámková, 2016)

1.2 Hudební signál

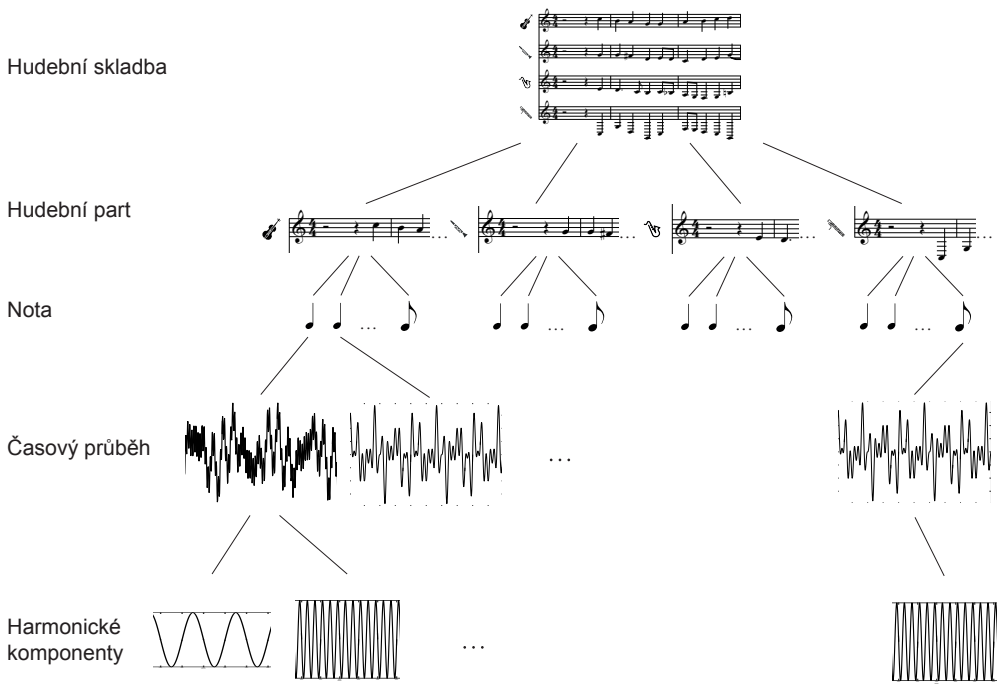
V předchozí definici jsme vstup do transkripčního systému označili jako *hudební signál*. Popišme si nyní jeho vlastnosti, informace v něm obsažené a jeho možné reprezentace.

Hudební signál svými vlastnostmi fascinoval učence již od dob Pythagora (Walmsley, 2001). Od té doby vzniklo nespočet vědeckých prací věnujících se jeho parametrům a možnostem jejich extrakce. Hudební signály jsou analyzovány z mnohých hledisek. Fyzikální akustika se zabývá jejich původem, šířením a vnímáním. Psychoakustika studuje jejich *percepční* vlastnosti, tedy vlastnosti související s vlastním smyslovým vjemem hudby. Hudební akustika se jím zabývá obecněji, studuje interdisciplinárně všechny příčiny a důsledky přenosu hudebního signálu a jeho vlastnosti z různých úhlů pohledu, např. akustiky, hudební teorie, estetiky a dalších. (Syrový, 2013)

Hudební signál definujeme jako *zvukový signál, který je nositelem hudební informace*. Konkrétněji jde o mechanické vlnění v látkovém prostředí, které je schopno vyvolat zvukové vjemy charakteristické pro hudbu, a to jak v krátkých časových intervalech (např. výška, barva, hlasitost), tak i v intervalech delších (např. melodie, rytmus, harmonie).

1.2.1 Informace v hudebním signálu

Hudební signály jsou charakteristické svou pevnou hierarchickou strukturou. Většinou se skládají ze zvuků souběžně generovaných z jednoho či více různých zdrojů, nejčastěji hudebních nástrojů. Hudební nástroje dělíme do dvou hlavních kategorií: harmonické a perkusní. Zvuky, které generují, se seskupují



Obrázek 1.1: Hierarchická struktura hudebních signálů ukázána na skladbě Ach Gott und Herr (Bach, 1893a)

do jednotných harmonických celků udávané tóninou, rytmickými strukturami, hudebními tématy a motivy. Harmonické zvuky se skládají z akordů, které se skládají z not. Noty jsou definovány svou výškou a barvou a jsou složeny z uspořádané série harmonických komponent. Náhled této hierarchické struktury můžete vidět na obrázku 1.1.

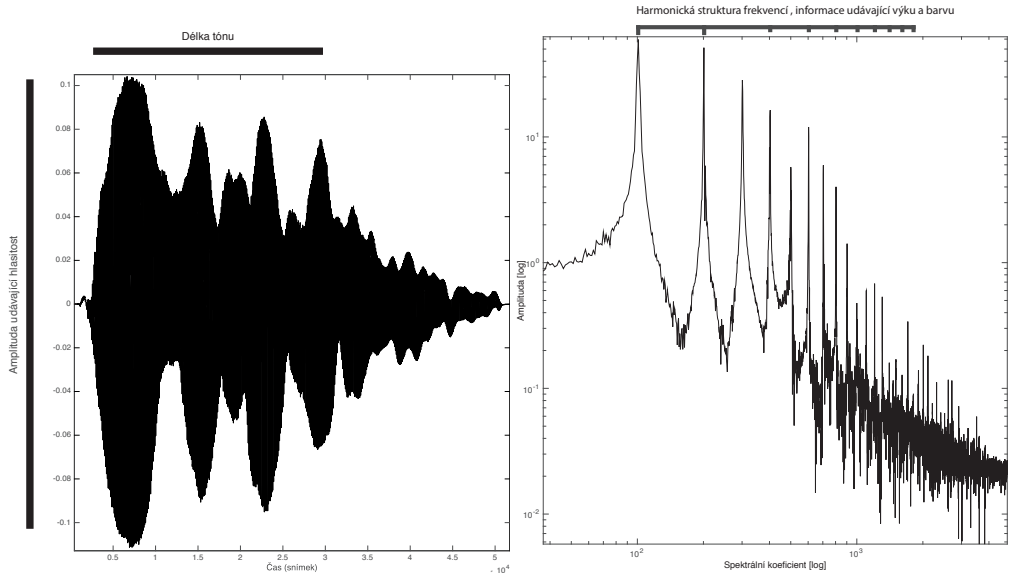
Podívejme se na tyto informace důkladněji a formálně si je definujme. Na hudební informace můžeme nahlížet z percepčního a fyzikálního hlediska. Fyzikální vlastnosti jsou objektivní, exaktně definované a snadno je dokážeme modelovat. Proces odrazu těchto vlastností na naše vědomí je však ovlivněn mnoha faktory a může být subjektivní. Nejprve si definujme percepční vlastnosti a hudební pojmy. Jejich fyzikální ekvivalenty, kterým je nutno rozumět při procesu transkripce, si odvodíme později.

Hudební skladbu zde definujeme jako takový úsek hudebního signálu, který působí na naše vědomí „celistvě“, tedy libovolný kratší úsek skladby dává smysl v kontextu zbytku. Některé dlouhodobé hudební parametry (rytmus, tónina) by měly zůstat v celé skladbě relativně neměnné, či by k jejich změně nemělo docházet často.

Skladba se skládá ze zvuků generovaných *hudebními nástroji*. Části signálu, který má původ v jednom z nástrojů, budeme říkat *part* neboli *hlas*. *Polyfonní* (vícehlasé) signály mají vstup z více nástrojů nebo z nástroje, který je schopen generovat zároveň více zvuků. *Monofonní* (jednohlasé) signály se oproti tomu vyznačují tím, že v libovolnou chvíli může znít hudební zvuk maximálně z jednoho zdroje. Zpravidla se jedná o hlavní melodii bez doprovodu. Protože se tato práce zaměřuje na monofonní signály, budeme se dále věnovat pouze pojmům k nim relevantním.

Hudební nástroje dělíme dle způsobu tvoření zvuku, tedy dle způsobu, jakým rozkmitávají vzduch. Zvuk z *perkusního nástroje* je charakteristický náhlým přísunem energie, která pak postupně zeslabuje. Oproti tomu *neperkusní nástroje* dodávají energii tónu průběžně. V kapitole 4 se podíváme na jednotlivé typy nástrojů důkladněji a popíšeme si vliv nástroje na vlastnosti signálu.

Nota Základním prvkem konkrétního partu skladby je *nota*. Lze ji definovat dvěma způsoby: označuje symbol, který je používán v hudební notaci pro zvuk hudebního nástroje. Také může značit zvukový signál, který je produkován hudebním nástrojem, když je na něj tento symbol „hrán“ (Klapuri, 2006). Mezi hlavní atributy noty patří její délka, výška, barva a prominence. Struktura signálu noty je pak popsána tzv. *ADSR modelem*, který si podrobněji popíšeme ve 4. kapitole (Bello; Daudet et al., 2004). Na obr. 1.2 můžeme vidět časový průběh noty A_4 hrané na housle a jeho spektrum spočítané z jeho zvučné části. Na těchto grafech jsou znázorněny základní informace o notě: délka, hlasitost a spektrální komponenty, které mají vliv na výšku a barvu noty.



Obrázek 1.2: Znázornění informací v hudebním signálu noty A_4 hrané na housle na jejím oscilogramu a spektru.

Tón Pojem tón je ekvivalentní se zvukovým signálem noty, či jeho podčástí. Jinými slovy jde o *libovolný signál, který vytváří silný dojem výšky*. Na rozdíl od noty tón nemusí mít pevnou strukturu. Jeho výška by však měla být relativně neměnná po celou dobu jeho trvání., což pro notu platit nemusí (např. když ji hráč hraje *vibrato*).

Výška tónu Jedná se o percepční vlastnost, která umožňuje uspořádat tóny od nejnižšího po nejvyšší. Přesněji výšku tónu definujme jako frekvenci sinusového signálu, s nímž má při subjektivním posuzování posluchačem s normálním sluchem zkoumaný tón stejnou výšku (Klapuri, 1997, kap. 1). Je úzce spjata s periodou a základní frekvencí signálu. Když se budeme bavit o frekvenci noty či tónu, budeme mít na mysli frekvenci odpovídající jeho výšce.

Hlasitost tónu je percepční atribut odvozený od fyzikální *intenzity* zvukového signálu. Odpovídající měřitelnou veličinou je hladina akustického tlaku. Její vnímání je závislé na frekvenci zvuku, typu signálu (sinusový tón, šum, řeč) a na sluchové citlivosti člověka. (Syrový, 2013) Stejně jako jiné percepční vjemy je hlasitost dle Weber-Fechnerova zákona[1] vnímána logaritmicky (Syrový, 2013). Intenzita zvukového signálu se měří v *decibelech*, kde zvuk o intenzitě I v jednotkce W/m^2 a dané referenční intenzitě $I_0 = 10^{-12}\ W/m^2$ má hladinu hlasitosti rovnou $L(I) = 10 \log \frac{I}{I_0}\ dB$. Subjektivně vnímaná hlasitost tónu se udává v *sonech*, kde 1 son odpovídá hlasitosti tónu o frekvenci 1000 Hz a intenzitě $40\ dB$. Další používanou jednotkou, která oproti sonu používá lineární závislost na frekvenci, je *fón*, který lze ze sonu vypočítat vztahem: x sonů $= 40 + 10\ log_2 x$ fónů. Percepční detekci hlasitosti tónu ztěžuje tzv. efekt *maskování*. Znění určitého tónu vyvolává totiž současně dočasný posun sluchového prahu směrem k vyšším hodnotám. Vjem dalšího tónu může být proto zeslaben. (Syrový, 2013, str. 57)

Barva tónu je pojem, který shrnuje zbylé vlastnosti zvuku, pomocí kterých „posluchač od sebe rozpozná dva zvuky stejné výšky a hlasitosti“ (Klapuri, 2006, kap. 1). Tyto vlastnosti hudebníci běžně kategorizují pojmy „světlý, temný, drsný, tvrdý, apod“. Barva tónu je též silně ovlivněná jeho *kvalitou*, jak z hlediska tvorby tónu, tak z hlediska kvality nahrávky. Barva hraje výraznou roli v tom, jak člověk rozpoznává hudební nástroje. Byť je tato vlastnost silně subjektivní, lze ji popsat i objektivně pomocí analýzy vývoje amplitudy jednotlivých harmonických komponent (Syrový, 2013, kap. 5). Protože má barva noty výraznou závislost na podobu signálu, popíšeme si ji podrobněji v kapitole 3.

Ladění Nota může mít dle předchozí definice libovolnou výšku, která je v rozsahu slyšitelnosti. V hudební notaci však pracujeme pouze s diskrétními hodnotami výšky. Také na většině hudebních nástrojů máme pouze omezené množství výšek not. Proto vybíráme pouze tóny, které lze spolu použít v nějaké hudební skladbě. V té nesmí chybět tóny, které spolu zní libozvučně. Již Pythagoras si povšiml faktu, že tóny spolu vjemově ladí nejlépe, když poměr se mezi jejich frekvencemi, reprezentován zlomkem v základním tvaru, skládá

[1] Intenzita smyslového vjemu je logaritmický závislá na intenzitě fyzikálního podnětu

z co nejmenších čísel (A Cheveigné, 2008). Z toho plyne, že souzvuk dvou různých tónů je nejladnější, když jsou jejich frekvence v poměru $1 : 2$. Tomuto intervalu se říká *oktáva*. Jejich podobnost je tak výrazná, že se nazývá *oktávovou ekvivalencí* a všem tónům v této vzdálenosti se dává stejné jméno (viz dále). Pro získání dostatečného množství různých tónů pro jejich použití v harmoniích a melodiích hudebních skladeb potřebujeme tento interval ještě v několika bodech rozdělit. Problémem výběru konkrétních intervalů, které budeme používat v notacích a v nástrojích, se zabývá *ladění*. Napříč hudební historií a kulturami vzniklo velké množství ladění, my se zde zaměříme na *temperované ladění*. (Syrový, 2013)

Jméno noty v temperovaném ladění je definováno následovně. Skládá se z písmena udávající *výškovou třídu tónu* (pitch class) a čísla udávající oktávu. Tón A_1 v temperovaném ladění odpovídá frekvenci 440 Hz. Od ní odvodíme všechny ostatní tóny. Jména tónů jsou definovány následující posloupností půltónů:

$$\{A_0, \ldots, C_1, Cis_1, D_1, Dis_1, E_1, F_1, Fis_1, G_1, Gis_1, \mathbf{A_1}, Ais_1, H_1, C_2, \ldots\}.$$

Jména not, které končí na příponu „is“, budeme nazývat *zvýšené*, jelikož zvyšují předchozí notu o půltón (např. C_1 na Cis_1). Na stejné noty se lze dostat snížením vyšší noty, proto pro tyto noty máme alternativní názvy:

$$\{A_0, \ldots, C_1, \mathbf{Des_1}, D_1, \mathbf{Es_1}, E_1, F_1, \mathbf{Ges_1}, G_1, \mathbf{As_1}, A_1, \mathbf{B_1}, H_1, C_2, \ldots\}.$$

Těmto budeme říkat noty *snížené*. Ostatní tóny pojmenujeme *základní*. Frekvenční poměr sousedních půltónů činí $\sqrt[12]{2}$. Tedy např. tón A_2 má frekvenci 880 Hz, protože je od A_1 vzdálen 12 půltónů (při použití výše popsaného vztahu získávám: $440 \cdot (\sqrt[12]{2})^{12} = 880$). Výšková třída je množina všech tónu v intervalu rovnou násobku tzv. oktávy, což je interval odpovídající 12 půltónům (např. výškové třídě A odpovídá množina $PC_A = \{A_1, A_2, \cdots\}$).

Hudební interval Protože je výška tónu vnímána logaritmicky, vzdálenost mezi dvěma tóny vychází z poměru jejich frekvencí. Hudební intervaly nám umožňuje měřit vnímané vzdálenosti výšky. Jejich jednotkou je 1 tón, jenž odpovídá dvěma půltónům. Uvědomme si, že tato jednotka je odlišný význam

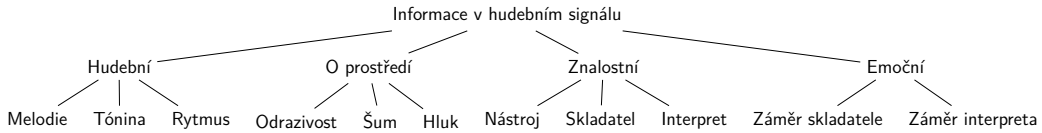
slova tón, než v definici uvedené výše. Dále v textu bude kladen důraz, aby bylo z kontextu možné rozpoznat, o kterém významu se bavíme. *Půltónem* pak označíme vnímanou vzdálenost dvou výšek, jejichž frekvence je v poměru $\sqrt[12]{2}$. Pro přesnější měření intervalů se používá jednotka *cent*, pro který platí vztah: 1 cent = $1/100$ půltónu. Z toho plyne, že tóny jsou od sebe vzdáleny 1 cent, když je jejich frekvenční poměr roven $\sqrt[100]{\sqrt[12]{2}} = \sqrt[1200]{2}$ (McLeod, 2009, str. 35).

Prominence noty udává míru, jak výrazně na nás nota působí v kontextu okolních not. Je důležitá především pro rytmickou analýzu skladby, ale hraje roli též ve vnímání tóniny. Prominence je daná především hlasitostí, barvou (a s ní související kvalitou tónu), přesností a délkou dané doby.

Hudební stupnice a tónina Na hudební stupnici lze nahlížet jako na určitou „paletu“ not (obdobně jako malíř má paletu barev), ze které čerpáme při tvorbě hudebních skladeb. Pokud je část skladby napsaná za pomocí not z jedné stupnice, říkáme, že je napsaná v její *tónině*. Tóninu lze definovat jako označení sekvence not příslušností k určité stupnici, jež provede průměrný posluchač při subjektivním sluchovým porovnávání s jednotlivými stupnicemi.

Nic autorovi skladby však nebrání v ní použít i noty z jiných stupnic (tzv. *modulace*), nebo během skladby tóninu úplně vyměnit. Často jsou tyto výkyvy žádoucí, jelikož vytváří napětí vyžadující rozuzlení, které dělá hudbu zajímavou (Temperley, 1999).

Formálně stupnici definujeme jako řadu tónů seřazených dle výšky, která je uspořádaná podle určitých pravidel. Stupnici lze jednoznačně určit pomocí počátečního tónu (tzv. *tónika*) a hudebního intervalu mezi sousedními tóny. Stupnice bývají z hlediska intervalů periodické s periodou v oktávě. Např. stupnice C dur má následující noty: $C_0, D_0, E_0, F_0, G_0, A_0, H_0, C_1, \ldots$ Pokud se podíváme na intervaly mezi jednotlivými tóny, zjistíme, že vzdálenosti v půltónech



Obrázek 1.3: Struktura informací v hudebním signálu

jsou následující: $\{2, 2, 1, 2, 2, 2, 1, \ldots\}$. Tato sekvence vzdáleností charakterizuje tzv. *durovou stupnici* a pomocí ní odvodíme libovolnou z nich. Název stupnice odpovídá její tónice. *Molové stupnice*, na rozdíl od durových, jsou charakterizovány následující posloupností vzdáleností: $\{2, 1, 2, 2, 1, 2, 2, \ldots\}$. *Tonalita* skladby popisuje vztahy mezi jednotlivými notami na základě jejich výšky a prominence. Tonální analýza skladby si klade za úkol detekovat příslušnost skladby k jedné či více stupnicím. (Duckworth, 2012).

Délka noty je čas, který uplyne od jeho nástupu (náhlý přísun energie), do chvíle jeho uvolnění (konec přísunu energie). Na obr. 1.2 je na grafu oscilogramu znázorněna vnímaná délka noty. Všimněme si, že konec noty nemusí odpovídat výraznému poklesu amplitudy obálky signálu. To bývá často způsobeno výraznou dokmitávací fází.

Rytmus Hudební skladby jsou charakteristické tzv. pulzováním – periodickým opakováním událostí v určitých časových intervalech. Naše citlivost na toto pulzování souvisí s fyziologickými rytmy v těle člověka, jako např. tlukot srdce a rytmus dechu Při rytmické analýze skladby zkoumáme vzájemné vztahy jednotlivých událostí a jejich prominencí. Hledáme pravidelně se opakující vzory hudebních událostí v závislosti na čase (Sokol, 2004). Rytmické vzory se vyskytují na několika úrovních, které tvoří hierarchickou rytmickou strukturu skladby. Každá úroveň odpovídá určité frekvenci, ve které by si člověk mohl do skladby „klepat“ a popisuje, na které noty „klepnutí dopadne“. Základní jednotkou rytmu je *doba*. Časová vzdálenost mezi dvěma dobami určuje *tempo* skladby. *Metrum* oproti tomu popisuje vzor, podle kterého jsou doby na jednotlivých úrovních seskupeny. Více si o analýze rytmu povíme v kapitole 5.

Kromě hudebních informací signál často nese další informace. Walmsey (2001) je kategorizuje následovně:

- **Informace o prostředí**
  Charakteristiky nahrávacího prostředí, informace o pozici a směru nahrávacího přístroje, šumy a hluky vzniklé při nedokonalém nahrávání…

- **Znalostní informace**
  Rozpoznání typu hudebního nástroje na základě znalosti jeho vlastností, asociace se známou melodií na základě melodické i harmonické podobnosti, znalost stylu hudebníka, rozpoznání hudebního žánru.

- **Emoční informace**
  Rozpoznání emočního záměru skladatele a interpreta.

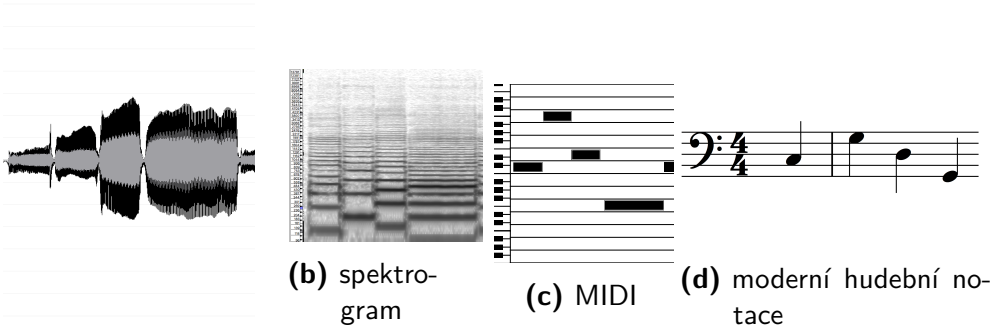
Hierarchickou strukturu informací v hudebním signálu lze vidět na obr. 1.3.

1.2.2 Reprezentace hudebního signálu

Již od počátků hudby byla její reprezentace důležitým tématem. Před vynálezem metody nahrávání hudby byla hudební notace jediným způsobem, jak hudbu zaznamenat. I dnes je otázka, jak hudbu uchovávat a neztrácet při tom žádné informace, stále aktuální. Společně s ní ale hledáme různé vyšší úrovně reprezentací, které nám umožní snadněji pracovat s informacemi ukrytými v hudebním signálu. Některé z těchto reprezentací jsou ztrátové, tedy již neobsahují informace potřebné ke zpětnému převodu (např. moderní hudební notace). Jiné jen zakódovávají původní informaci alternativním způsobem, který je pro nás z nějakého důvodu praktický (např. spektrogram).

Z výše zmíněných definic transkripce je patrné, že v hudební transkripci nám jde o jakousi transformaci reprezentace hudebního signálu. Podívejme se detailněji, jakým způsobem lze tyto informace reprezentovat a jaké jsou výhody různých reprezentací.

Po vzoru Vineta (2004) si rozdělíme typy reprezentací na čtyři úrovně: *fyzikální*, *signálovou*, *symbolickou* a *znalostní*.



(a) časový průběh zvukového signálu

(b) spektrogram

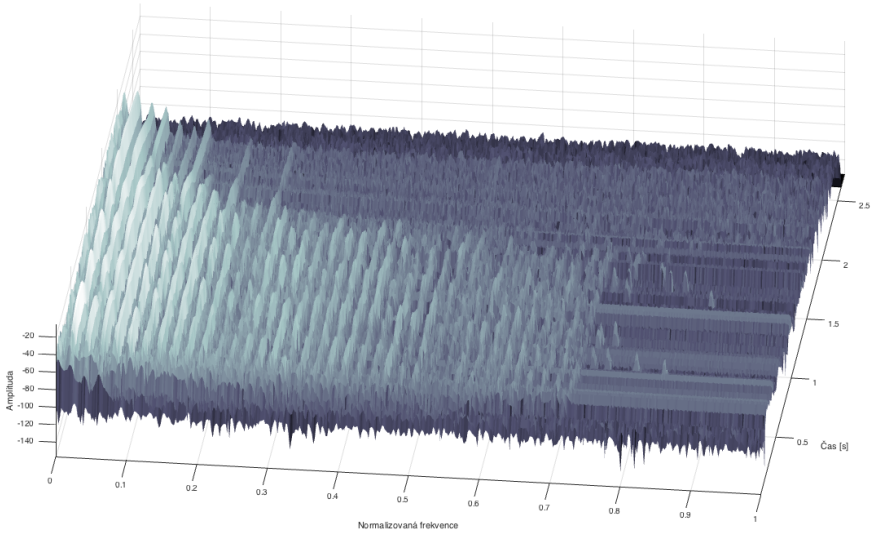
(c) MIDI

(d) moderní hudební notace

Obrázek 1.4: Ukázky reprezentace hudebního signálu

- **Fyzikální úroveň**
  V této úrovni reprezentujeme hudbu v její nejsurovější formě: mechanickým vlněním v látkovém prostředí. Toto vlnění lze popsat jako funkci změny akustického tlaku v čase a v místě. Její hodnoty jsou závislé na zdrojích zvuku (lidské hlasivky, hudební nástroj), vodiči zvuku (obyčejně vzduch), prostředí (akustické vlastnosti prostor, typ a tvar překážek v nich) a na přijímači (lidské ucho, mikrofon).

- **Signálová úroveň**
  Časový signál, kterým zde myslíme popis změny nějaké jednotky v závislosti na čase, získáme z fyzikální úrovně reprezentace hudebního signálu jeho nahrávkou v konkrétním místě. Patří sem nejčastější způsob reprezentace hudby: časový průběh zvukového signálu. Vyjadřuje hodnoty *amplitudy* – intenzity akustického tlaku, kterým zvukové vlnění působí na přijímač – v závislosti na čase. Může mít několik různých forem. *Analogový* zvukový signál uchovává spojitý průběh tohoto měření. *Digitální* zvukový signál je časová řada diskrétních hodnot a získáme ji z analogového signálu procesem *vzorkování* a *kvantování*. Reprezentace časového průběhu je však z hlediska získávání hudebních informací značně nepraktická. Kromě tvaru obálky noty, která v něm často dobře rozlišitelná, nám explicitně poskytuje jen velmi málo znalostí.



Obrázek 1.5: Spektrogram hudebního signálu noty A4 hrané na housle.

- **Meziúroveň**
  V hudební transkripci nám půjde o převod ze signálové úrovně na úroveň symbolickou. Obtížnost tohoto procesu si usnadňujeme iterativním postupem přes tzv. *„meziúrovně“* (mid-level representations), z nichž se každá specializuje pouze na konkrétní informace, které pak při převodu do symbolické notace sloučíme (Kitahara, 2010). Spektrogram, jenž je vizuální znázornění amplitudového spektra signálu závislého na čase, nám např. oproti zvukovému signálu poskytuje mnohem lepší možnosti k analýze vlastností not. Poskytuje trojrozměrnou reprezentaci signálu, kde jsou změny amplitudy promítnuty jak v časové, tak ve frekvenční rovině. Získáme ho rozdělením vstupního digitálního signálu na krátké, překrývající se časové úseky, na kterých provádíme *Fourierovu transformaci*, tedy převod z časové do frekvenční domény (spektra). Získané spektrum nám pak ukazuje sílu jednotlivých frekvenčních komponent v daném úseku signálu, které můžete vidět na obr. 1.5. Ve 3. kapitole se seznámíme s několika dalšími meziúrovňovými reprezentacemi použitými jako mezikrok k převodu do moderní hudební notace.

- **Symbolická úroveň**
  Oproti signálové úrovni, není smyslem té symbolické reprezentovat každou informaci hudebního signálu. Spíše se na ni dá pohlížet jako na instrukce pro hudebníka, které říkají co, kdy a jak má hrát. Na rozdíl od signálové úrovně si je vědoma hudebního obsahu a ukládá informace v závislosti na kontextu. Formalizuje koncepty hudební teorie vyskytující se v signálu. Používá k tomu symboly, kterými značí konkrétní události v čase nebo parametry platné v daném časovém intervalu. Musical Instrument Digital Interface (MIDI) a moderní hudební notace patří mezi nejčastější a budeme se jim věnovat dále.

- **Znalostní úroveň**
  Na této úrovni popisujeme globální charakteristiky díla jako např. hudební žánr, použité nástroje, jméno díla, hudební kvality apod. Výstup těchto informací nemá přesnou strukturu a volba jazyka k jejich formulaci by neměla být podstatná.

Na obrázku 1.4 je znázorněn záznam fagotu hrající začátek skladby Ach Gott und Herr (Bach, 1893a). Jsou zde ukázány čtyři nejpoužívanější reprezentace: časový průběh zvukového signálu (*oscilogram*), jeho spektrogram, MIDI reprezentace vyobrazená pomocí *piano roll* notace a moderní hudební notace.

1.2.3 Digitální záznam hudebního signálu

Popišme si nyní zdaleka nejpoužívanější reprezentaci hudebního signálu – digitální záznam. Ten představuje diskrétní hodnoty jeho amplitudy v diskrétních časových momentech. Nejčastější princip jeho získání je následující. V místě, ve kterém chceme nahrávku provést, umístíme mikrofon, který převádí hodnoty akustického tlaku na elektrický signál, který pak následně převedeme na posloupnost diskrétních hodnot.

V první fázi převodu, tzv. *vzorkování*, dochází k sejmutí hodnoty elektrického napětí ze signálu v pravidelném časovém intervalu. Tento interval je daný parametrem *vzorkovací frekvence*, který budeme značit f_s. Pokud je např. $f_s = 44100$ (nejpoužívanější hodnota), dojde k sejmutí hodnoty ze signálu 44100 krát za sekundu. K tomu se používá obvod *S/A* (sample and hold), který sejme hodnotu napětí průběžně se měnícího analogového signálu a udrží ji po daný časový interval.

Hodnoty vzorkovaného signálu jsou však spojité a mohou nabývat libovolných hodnot. Pro digitální reprezentaci v počítači však máme počet hodnot, kterých signál může nabývat, omezený. V druhé fázi převodu, tzv. *kvantování* proto zaokrouhlíme naměřené hodnoty amplitudy na nejbližší reprezentovatelné hodnoty. Ty jsou dány velikostí paměti, kterou jsme ochotni použít k reprezentaci jednoho vzorku. Této velikosti říkáme *bitová hloubka* digitálního signálu, a nejčastěji se pro ni používají hodnoty 16, 24, 32 a 64 bitů. Při bitové hloubce N dokážeme reprezentovat 2^N různých hodnot, z čehož můžeme odvodit teoretický rozsah dynamiky signálu L, který je dán v *decibelech* následovně: $L = 20 \log 2^N$. Tedy např. 16 bitový zvukový záznam má dynamický rozsah 96,33 dB. (Syrový, 2013)

Během vzorkováni a kvantování zjevně musí docházet ke ztrátě informace. Pokud vysázíme hodnoty digitálního signálu do prostoru, existuje nekonečno možností, jak by jimi mohl analogový signál procházet. Pokud je však dodržena podmínka, že maximální frekvence spektrální složky ve zpracovávaném signálu f_{max} nepřekročí $\frac{f_s}{2}$, tak dle *Shannon-Hartleyova teorému* je spojitý průběh signálu dán jednoznačně (Syrový, 2013). Intuitivně lze tento fakt vysvětlit pomocí faktu, že pro zjištění frekvence libovolného sinusového signálu potřebujeme znát alespoň dvě hodnoty v časovém intervalu menším než jeho perioda. Ke splnění Shannonovy podmínky je třeba použít filtr *dolní propust*, který frekvence vyšší jak $\frac{f_s}{2}$ odfiltruje.

Důsledek kvantování je ve většině případech ekvivalentní s přidáním náhodného šumu o amplitudě rovnou polovině nejmenšího reprezentovatelného intervalu. Při volbě bitové hloubky pak volíme kompromis mezi tím, jaké množství šumu jsme ochotni ve výsledném signálu tolerovat v kontrastu s množstvím dat, které jsme schopni zpracovat či uchovat. (S. W. Smith, c1997)

V této práci se digitalizací signálu více zabývat nebudeme. Na vstupu hudební transkripce budeme očekávat posloupnost x_t (získanou např. ze zvukové karty), která představuje hodnotu vstupního signálu v čase t. Dále předpokládáme, že doba mezi dvěma vzorky $x_{t+1} - x_t$ je rovna $\frac{1}{f_s}$. Pro jednoduchost budeme navíc předpokládat, že každý vzorek x_t je reálné číslo v intervalu $[-1, 1]$. Jelikož se rozsah slyšitelných frekvencí člověka pohybuje v rozsahu 20 Hz–20 kHz, použijeme na základě výše zmíněného Shannonova teorému běžně používanou hodnotu $f_s = 44100$.

Z hlediska teorie signálů bychom tento signál označili za *stochastický*, tedy nahodilý – nepopsatelný jednoznačnou matematickou funkcí, ale pouze prostředky teorie pravděpodobnosti a matematické statistiky. Z hlediska závislosti na počátku časové osy bychom tento signál označili za *nestacionární*, tedy v čase měnící své vlastnosti (Syrový, 2013, str. 114). Tyto signály jsou pro podrobnou analýzu velmi složité. Pokud bychom však vzali v potaz pouze dostatečně krátký časový úsek, jenž by byl z percepčního hlediska neměnný, např. interval ustálené části zvuku jednoho tónu, dal by se z určitého hlediska považovat za relativně *stacionární*. Tohoto faktu využijeme při snaze získávání informací z daného signálu. Pro reprezentaci těchto krátkých časových úseků budeme používat *časový průběh* a *spektrum*. Časový průběh je závislost amplitudy této části signálu na čase, tudíž jej budeme nazývat *časovou doménou*. Spektrem označujeme výsledek převodu tohoto intervalu do tzv. *frekvenční domény*. Tento převod využívá již více jak dvě století známou metodu *Fourierovy transformace*, která převádí *periodický* signál na součet *sinusových* funkcí. Každá složka tohoto součtu je definována svojí *amplitudou*, *frekvencí* a *fází* a budeme ji nazývat *spektrální komponentou*. Pro nás bude důležitá jejich frekvence a amplituda. Grafické znázornění časového průběhu a spektra signálu lze vidět na obr. 1.2. Přísně vzato náš signál periodický není, jelikož je stochastický. Tyto stacionární úseky hudebního signálu však bývají téměř periodické, neboli *kvaziperiodické*, výpočet spektra pro ně bude tedy celkem dobrým odhadem.

1.2.4 Reprezentace hudebních událostí

Další rozšířenou reprezentací hudebního signálu, je výpis tzv. *hudebních událostí* z hudebního signálu s informací o čase jejich výskytu. Hudebními událostmi zde rozumíme nejčastěji nástup (onset) a uvolnění (offset) noty a lze ho vizualizovat pomocí tzv. *piano roll* notace (obr. 1.4c). MIDI je standardizovaný protokol pro ovládání hudebních nástrojů. Součástí jeho normy je popis binární reprezentace ve formátu Standart Midi File (Standartní Midi soubor) (SMF). Ten obsahuje zmíněný popis událostí, kterými mohou mimo jiné být nástupy noty a jejich uvolnění. Informace o notových událostech je obohacena celočíselnou reprezentací výšky noty s přesností na půltón a prominencí.

Reprezentace hudebních událostí patří z hlediska informace mezi nízkoúrovňové symbolické reprezentační úrovně a stále ji nelze triviálně přepsat do hudební notace. Notové události zde totiž mají příliš velkou časovou variabilitu, zatímco notové zápisy mají časové rozlišení tzv. kvantované. Oproti hudební notaci zde také chybí explicitní rytmické a tonální informace a informace o partech. Tato informace je implicitně zakódovaná v časových a tónických informacích a v prominenci not a je třeba ji teprve extrahovat.

Přesto je tato reprezentace vhodná jako mezikrok hudební transkripce, a to především díky její kompaktnosti a tomu, že zachovává všechny důležité informace pro další kroky (Klapuri, 2006). Motivací k jejímu použití může být též její rozšířenost – právě MIDI formát je vstupem mnohých algoritmů pro detekci rytmu či tóniny (Klapuri, 1997, kap. 4).

1.3 Hudební notace

Hudební transkripci jsme si definovali jako přepis hudebního signálu do symbolické reprezentace. V této kapitole si popíšeme, co je touto reprezentací myšleno. Dále si jednu z nich podrobně představíme. Půjde o *moderní hudební notaci*, kterou budeme používat v naší implementaci.

Již od počátků hudební notace byl její hlavní účel uchování hudební skladby pro pozdější interpretace. V době jejího vzniku to byl kromě lidské paměti jediný způsob, jak hudební nápady uchovávat. V předchozí kapitole jsme si ukázali několik úrovní reprezentací signálu, kdy jsme při přechodu na vyšší

úroveň dosáhli pochopení hudebního signálu na abstraktnější úrovni. Poté jsme převáděli informace popsané pojmy z teorie signálů na informace hudební teorie. Při tom však docházelo k nějaké ztrátě informace. Popišme si, jak tomu je u reprezentace hudební notací.

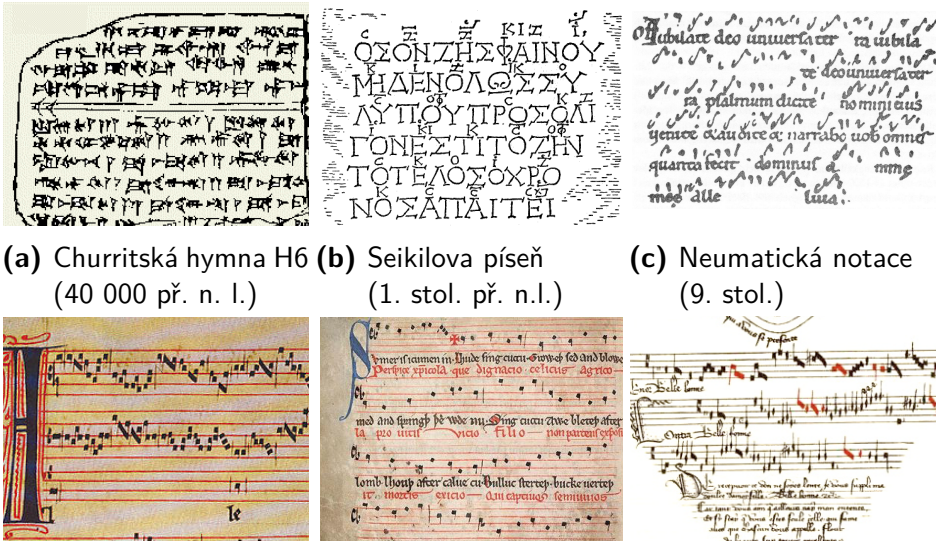
Hudební notace je symbolickou reprezentací specifickou svým účelem: má sloužit jako instrukce pro hudebníka a říkat mu, co, kdy a jak má hrát. Zde se nabízí otázka, zda jsou tyto informace opravdu v hudebním signálu dostupné, tedy zda je transkripce opravdu teoreticky možná. Mnohé notace umožňují zaznamenání informace, která se nikterak v signálu neprojeví, např. prstoklad klavíru v moderní hudební notaci. Z naší definice transkripce však plyne, že při zaznamenání se nesnažíme uchovat způsob, jakým hudebník skladbu zahrál. Spíše nám jde o to dosáhnout takové notace, aby z ní měl hudebník dostatek informací, aby mohl podobného signálu docílit (určitý princip reverzního inženýrství). (Klapuri, 1997)

Hudební notaci definujeme jako *systém použitelný k vizuální reprezentaci instrukcí k interpretaci hudební skladby*. Hudební notace hledá kompromis mezi dostatečnou úrovní abstrakce pro snadné čtení a snahou reprezentovat dostatek informací, aby byl charakter skladby zachován. Charakter skladby lze však chápat různými způsoby a každá hudební notace jej může chápat jinak. To se odráží v možnostech detailnosti zápisu, která určuje, které věci jsou pevně dané a jaká rozhodnutí jsou ponechána na interpretovi.

Kromě již zmíněné reprezentační funkce může hudební notace také sloužit jako tzv. kognitivní medium – médium které povzbuzuje kreativitu a myšlení. Jeho abstrakce umožňuje při práci s ním přicházet na nové nápady úprav skladby, hledání nových souvislostí a vytváření složitějších kompozic, např. skladby pro více jak stočlenný orchestr. (Nielsen, 2016)

1.3.1 Vývoj hudební notace

Hudební notace vznikla nezávisle na více místech světa, chvíli se z historie vytratila a pak byla znovuobjevena. Z počátku sloužila spíše jako paměťová pomůcka a neobsahovala příliš mnoho informací. S vývojem hudební teorie se však zdokonalovala a nabývala na expresivnosti i na přesnosti.



(a) Churritská hymna H6 (40 000 př. n. l.)
(b) Seikilova píseň (1. stol. př. n.l.)
(c) Neumatická notace (9. stol.)
(d) Alleluia nativitas, 4 notové linky (12. stol.)
(e) Sumer Is Icumen In, 5 notových linek (1260-1270)
(f) Belle, bonne, sage, mensurální notace (1350-1400)

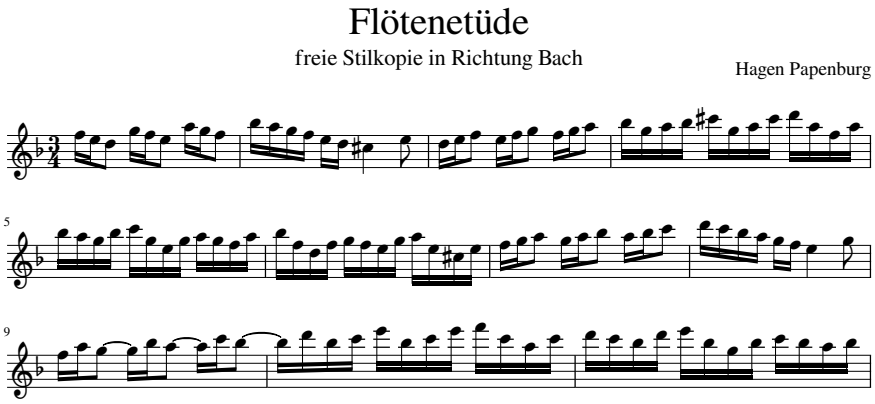
Obrázek 1.6: Historie hudební notace (Paterson, 2015; Cordier, 2017)

Nejstarší známý hudební nástroj je jednoduchá flétna z mamutoviny, vyrobená přibližně v r. 40 000 př. n. l., která byla nalezená na území dnešního Německa. Nejstarší známou písní je *Churritská hymna H6* zaznamenaná kolem roku 1400 př.n.l. na území dnešní Sýrie. Bohužel však existuje mnoho rozporuplných interpretací její notace a kompletní dešifrování již pravděpodobně není možné (Ondříčková, 2012). Oproti tomu nejstarší plně dochovaná skladba zaznamenaná v pro nás čitelné hudební notaci je z 1. stol. př. n.l. Jedná se o *Seikilovu píseň* a pochází ze starověkého Řecka (Paterson, 2015). Notace používaná ve starověkém Řecku a Římě byla textová. K reprezentaci výšky noty používali znaky své abecedy. Naše pojmenování not písmeny od A do G pochází odtud. .

Notace známá pro dnešní západní svět se vyvinula z církevního *gregoriánského chorálu* známého z 5. stol. n. l. z Itálie a Španělska. Tehdejší systém zpočátku spočíval pouze ve značkách (*neumech*) umístěných nad textem písní. Tyto značky popisovaly, zda je aktuální nota výše či níže, než nota předchozí. Tato notace je bohužel z hlediska transpozice volná k interpretaci (neznáme absolutní výšky not), a proto si nemůžeme být jisti, jak přesně dochované skladby z této doby zněly. Problém byl však brzy vyřešen dokreslením notových

linek. Nejprve jedné, v 10. stol. se objevují čtyři a až ve 12. stol se objevuje dnešní notová osnova sestávající z pěti notových linek. Vývoj notace výšky je následovan dodáním notace rytmu a délek not. Ve druhé polovině 13. stol. vznikla tzv. *mensurální* (mensura lat. míra) notace, která používala délky not velmi podobné dnešní notaci. V 16. století se pak objevilo předznamenání udávající tóninu. (Paterson, 2015)

1.3.2 Moderní hudební notace

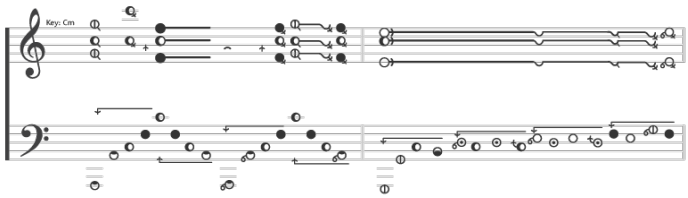


(a) Moderní hudební notace skladby (Papenburg, 2017)

(b) Melodická linka skladby vygenerovaná z nahrávky autora v softwaru Sonic Visualiser (Cannam et al., 2010)

Obrázek 1.7: Flötenetüde od Hagena Papenburga (Papenburg, 2017)

V dnešní době je v západním světě pravděpodobně nejpoužívanější moderní hudební notace (Western Music Notation). Nejprve se ustálila v zápisu vážné hudby, dnes je používána napříč všemi žánry. Její rovnováha mezi expresivností a jednoduchostí je již staletí nepřekonaná. I pro člověka bez hudebních znalostí

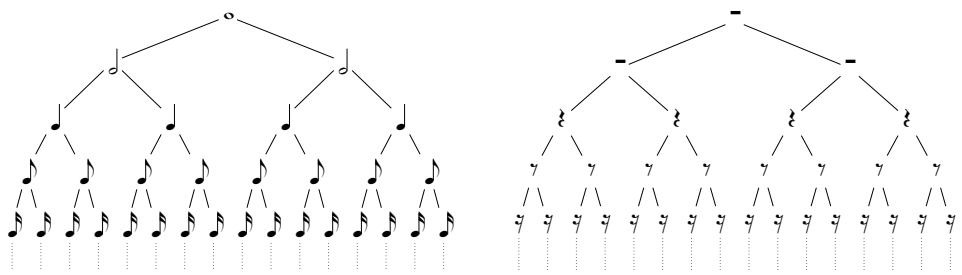


Obrázek 1.8: Alternativní notace Hummingbird (převzato z West, 2014)

není těžké z pohledu na noty pochopit základní melodické a rytmické informace v ní skryté. Pomáhá tomu např. melodická linka kopírující pozici not, jak lze vidět v porovnání notace skladby Flötenetüde a její melodické linky prvních dvou taktů na obr. 1.7.

Kromě té moderní jsou rozšířeny také jednodušší hudební notace, které jsou zaměřené pro konkrétní nástroj. Často jejich vizualizace odpovídá částem nástroje a značky odpovídají instrukcím akcí, které je na nástroji třeba vykonat pro zahrání skladby. Mezi ně patří tabulatura, která dříve sloužila pro loutny a varhany a dnes jí používají především kytaristé. Pro piano existuje méně rozšířená notace *Klavarskribo* a Piano roll notace. Stále se však objevují nové notace, které se snaží zjednodušit či zpřesnit moderní hudební notaci. (Couch, 2017) Příkladem může být notace *Hummingird* (obr 1.8), Rozšířenost moderní hudební notace však pravděpodobně v blízké době nedovolí, aby ji nahradila nějaká jiná. Proto v této práci budeme provádět transkripci právě do ní.

Jelikož použitá hudební notace udává, jaké informace musíme z hudebního signálu extrahovat, důkladně si ji popíšeme. Zaměříme se na způsob vizualizace jednotlivých percepčních vlastností hudebních signálů popsaných v kapitole 1.2.1 a popíšeme si možnosti a limity jejich reprezentovatelnosti.



Obrázek 1.9: Délky dob a pomlk v moderní hudební notaci (každý symbol odpovídá součtu dvou kratších)

1.3.2.1 Informace v moderní hudební notaci

Moderní hudební notaci čteme podobně jako běžný text po řádcích zleva doprava. Na řádky se lze dívat jako na časovou osu, na které je znázorněno, co přesně ve skladbě v daný moment probíhá. Základními symboly notace jsou noty a pomlky, udávající, kdy je na hudební nástroj vydáván zvuk a kdy ne. Délka dané noty či pomlky je znázorněna použitým symbolem. Orientaci na této časové ose nám usnadňuje takt, který rozděluje skladbu na úseky stejné délky (svislé čáry v hudební notaci).

Délky not Způsob znázornění délky noty či pomlky je následující. Jednotkou času v hudební notaci je *doba*. Pro noty ($\mathcal{N}$) i pomlky ($\mathcal{P}$) máme definovanou posloupnost symbolů :

$$\mathcal{N} = \{ \mathbf{o}, \mathbf{𝅗𝅥}, \mathbf{♩}, \mathbf{♪}, \mathbf{𝅘𝅥𝅯}, \mathbf{𝅘𝅥𝅰}, \mathbf{𝅘𝅥𝅱}, \ldots \}$$

$$\mathcal{P} = \{ \mathbf{𝄻}, \mathbf{𝄼}, \mathbf{𝄽}, \mathbf{𝄾}, \mathbf{𝄿}, \mathbf{𝅀}, \mathbf{𝅁}, \ldots \}$$

Označme si délku symbolu x jako $|x|$. Pak pro jejichž délky platí:

$$|\mathcal{N}_1| = |\mathcal{P}_1| = 4\,doby$$
$$|\mathcal{P}_i| = \frac{|\mathcal{P}_{i-1}|}{2}.$$

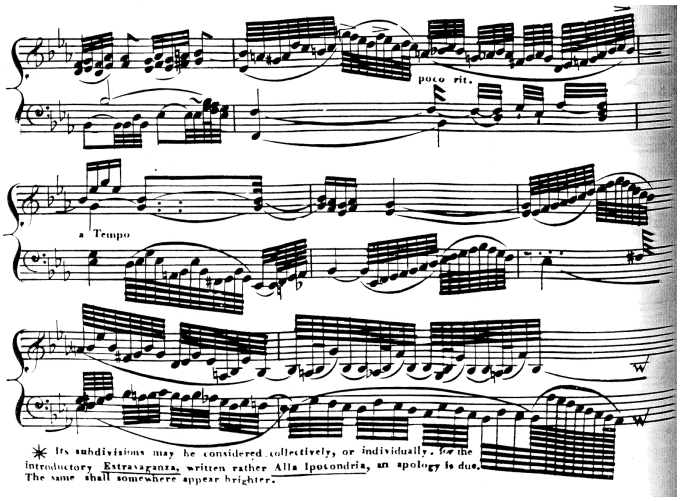
Tedy první symbol v této posloupnosti je dlouhý čtyři doby a každý další má poloviční délku než ten předchozí. Tento vztah reflektuje jejich pojmenování, kde první nota v posloupnosti se nazývá nota celá, resp. pomlka celá, další pak půlová, čtvrťová, osminová, šestnáctinová, atd. Znázornění tohoto vztahu lze vidět na obr. 1.9.

Tyto symboly nás ale zatím umožňují reprezentovat pouze noty o délce $2^{2-i}, i \in \mathbb{N}$. Aby bylo možné vyjádřit i noty jiných délek, používá se tzv. *ligatura*, která obloučkem spojuje noty stejné výšky. Výsledek pak představuje notu o délce součtu délek spojených not. Např následující úsek:

má délku $2 + 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} = 3\frac{31}{32}$ doby.

Jelikož se noty délky jeden a půl doby v hudbě vyskytují často, *tečkovaná notace* nám zkracuje jejich zápis. Abychom nemuseli psát ligatury přes několik not, tečka za notou prodlužuje danou notu o polovinu. Teček za notou je možné psát i více a vztah platí rekurentně. Nota délky t dob s n tečkami má tedy délku $t + \frac{t}{2} + \frac{1}{4} + \ldots + \frac{1}{2^n} = 2t - \frac{t}{2^n}$., tedy např. nota půlová se dvěmi tečkami má tři a půl doby:

Za pomoci ligatur jsme již schopni reprezentovat libovolnou délku noty z digitálního záznamu. To lze ukázat např. konstruktivně: pokud chceme vytvořit notaci noty délky t dob, kde jedna doba odpovídá délce T sekund, jsme schopni vytvořit multimnožinu not N, která, po přenesení na notovou osnovu a spojení ligaturou má délku t. Budeme postupovat iterativně. Nejprve definujme $d_1 = t$. Pak v kroku i provedeme následující operace: Pokud je $d_i \geq 2^{3-i}$, přidáme do multimnožiny N právě $\lfloor \frac{d_i}{2^{3-i}} \rfloor$ not typu $\mathcal{N}_i$ a upravíme $d_i = d_i - 2^{3-i}$. Pak spočteme $d_{i+1} = d_i/2$. Převod ukončíme ve chvíli, kdy platí: $d_i \cdot T \leq 1/f_s$, kde f_s je vzorkovací frekvence vstupního signálu (maximální v signálu reprezentovatelný časový úsek).



Obrázek 1.10: Toccata Grande Cromatica od Anthony Heinricha v moderní hudební notaci

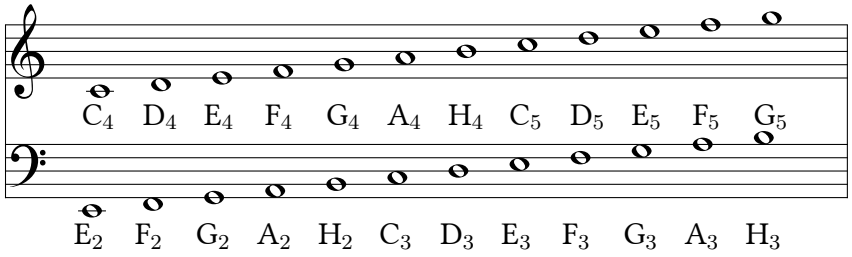
Jelikož po i. kroku platí, že $d_{i+1} \leq 2^{3-i} \Rightarrow d_i \leq 2^{2-i}$, procedura skončí v kroku j, takovém, že $d_j \leq \frac{1}{f_s}$, tedy když $2T^{2-j} \leq \frac{T}{f_s}$. Z toho plyne, že podmínka ukončení je splněna pro $j \geq 2 - log_2(\frac{T}{f_s})$. Např. pokud $T = 0.5, f_s = 44100$, pak $2 - log_2(\frac{0.5}{44100}) \doteq 18.43$, tedy procedura je ukončena v 19. kroku. Jelikož v transkripci nám jde o vytvoření záznamu percepčně odpovídajícímu skladbě, můžeme místo minimální délky udané vzorkovací frekvencí použít maximální časový úsek, který jsme schopni vnímat. Podle Klapuriho (1997) dokáže člověk rozpoznat až $t_{max} = 5\,ms$ dlouhé časové odchylky. Vzdálenost mezi dvěma nástupy not navíc musí být alespoň $t_{int} = 20\,ms$. Výše zmíněná procedura pak může skončit tehdy, když $d_j \leq t_{max}$, tedy v mezním případě $T = t_{int}$ získáváme z výrazu $2 - log_2(t_{max} \cdot t_{int}) \doteq 15.29$, tedy pro libovolný smysluplný vstup má procedura maximálně 16 iterací. Pro noty délky čtyři doby a méně použijeme za využití tečkované notace tedy maximálně 8 symbolů.

Hudební notace tedy umožňuje dostatečnou přesnost pro reprezentaci libovolných časových intervalů. Přesto není výše popsané užití ligatur pro reprezentaci exotických délek not příliš obvyklé – přestože nejkratší použitá nota v publikované hudební notaci je $1/2048$ (poslední nota na obr. 1.10), je to nezvyklý extrém. Druhá nejkratší doba je dlouhá pouze $1/256$ doby v díle Concerto in C for ottavino od Vivaldiho (Byrd, 2017). Při komponování hudby se totiž člověk drží jakéhosi nepsaného pravidla, že „v jednoduchosti je krása“ a v notaci

se vyskytuje velmi omezené množství použitých délek not. Toto pravidlo plyne z rytmického pojetí hudby, které nám zarovnává hudební události do jakési mřížky. Hudební signál na vstupu však toto zarovnání nesplňuje, což je způsobeno záměrným či nezáměrným přičiněním interpreta.

Absolutní délku jedné doby T není povinné udávat. Pokud ale chceme *tempo* skladby v notaci zmínit, máme dvě možnosti. Přesné zadání délky se udává na začátku notace nad prvním taktem, tradičně informací, kolik not jakého typu odpovídá jedné minutě. Např. ♩ $= 120$ značí, že do jedné minuty se vejde 120 čtvrťových dob, tedy jedna čtvrťová doba odpovídá půl sekundě. Druhá možnost zápisu je slovní popis, který udává tempo relativně k rychlosti lidského tepu (60-90 úderů za minutu). Tento způsob zápisu se nejčastěji vyjadřuje pomocí italského názvosloví, kde např. *Allegro* značí rychle (116-120 bpm), *Presto* značí velmi rychle (168–200 bpm), *Andante* značí volně (76–108 bpm) a *Adagio* značí zdlouhavě (66–76 bpm). (Kraemer, 2016)

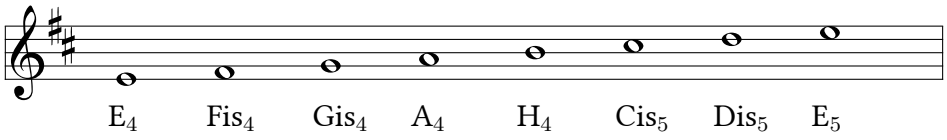
Výška not je znázorněna její pozicí na pěti linkách notové osnovy a tzv. *klíčem* notové osnovy. Z různých klíčů si popíšeme dva nejčastější. *Houslový klíč* (G klíč) umisťuje notu G_4 na první linku zespoda. Ostatní základní tóny (viz definice jmen tónu, kap. 1.2.1) umisťuje střídavě mezi linky a na linku. Nad a pod notovou osnovou se přidávají pomocné linky pro snazší čtení. Houslový klíč při snižování tónů navazuje na *basový klíč* (F klíč). Ten umisťuje notu F_3 na druhou linku shora.



Ze základních tónů se dostaneme na zvýšené (vyšší o půltón) vložením symbolu křížku před notu: 𝄪 . Snížení noty o půltón docílíme vložením „béčka“ před notu: 𝄫 . Tyto symboly platí od jejich použití po doby celého taktu.

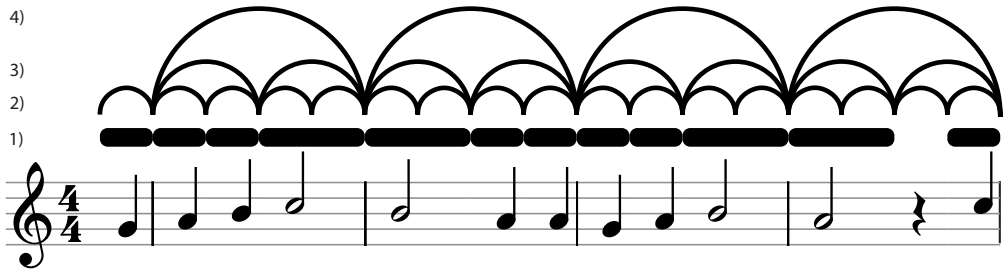
Moderní hudební notace umožňuje znázornit pouze výšky not temperovaného ladění, které jsme si definovali v předchozí kapitole. Je nutné si uvědomit, že z tohoto omezení plyne, že moderní hudební notace neumožňuje zaznamenat libovolnou skladbu. Mnoho hudebních skladeb, zejména těch folklórních, používá netradiční ladění. Např. v prvním systému pro částečně automatizovanou transkripci bylo cílem převést americké černošské písně, na jejichž tónový rozsah moderní hudební notace nestačí. V tomto případě byla místo ní použita tzv. *půltónová mřížka*, do které se melodická linka ručně zakreslovala (Metfessel et al., 1928). My zde však budeme předpokládat vstup odpovídající temperovanému ladění.

Tónina Tóninu jsme si v kapitole 1.2.1 označili jako příslušnost k nějaké stupnici, kterou jsme si definovali pomocí počátečního tónu a struktuře intervalů mezi sousedními tónu. Pokud se podíváme např. na stupnici E dur = E_0, Fis_0, Gis_0, A_0, H_0, Cis_1, Dis_1, E_1, ... Více jak polovina tónů této stupnice je zvýšených, pokud bychom při každém použití těchto not použili křížek pro zvýšení, byl by notový zápis velmi hutný. Proto se zavádí tzv. *předznamenání*, které nastaví křížky či béčka, které budou platit po dobu celé skladby. Předznamenání píšeme přímo za klíč notové osnovy, tedy např. stupnice E dur bude zapsána následovně:



Pokud bychom v této stupnici chtěli použít třeba notu F_4, nepoužívá se pro její snížení z Fis_4 symbol béčka, ale *odrážka*, která ruší efekt předznamenání pro aktuální takt: ♮ .

Rytmus Hudební notace má několik způsobů jak zobrazit rytmickou strukturu skladby. Pokud je rytmus v notaci dostatečně expresivně vyjádřen, výrazně tím usnadňujeme interpretovi čtení a pochopení, jakou rytmickou úlohu má jaká nota v kontextu skladby.



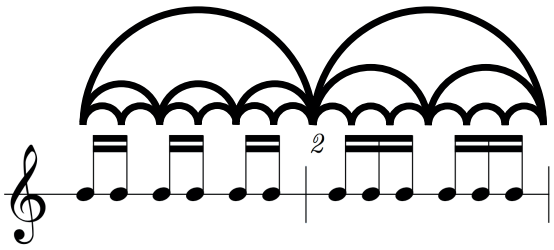
Obrázek 1.11: Vizualizace rytmu obloučkovou notací
1) Piano roll notace skladby
2) Nultá (nejnižší) úroveň – tatum – odpovídající čtvrťové notě
3) První úroveň – odpovídá půlové notě
4) Druhá úroveň – takt – odpovídá celé notě

Úroveň	Perioda	Typ	Fáze
0.	$1/4$ doby	-	-
1.	$1/2$ doby	2 podúrovně	1 doba
2.	1 doba	2 podúrovně	1 doba

Tabulka 1.1: Tabulka popisující rytmické úrovně skladby na obrázku 1.11

Pro snazší pochopení každou rytmickou úroveň můžeme vizualizovat sérií oblouků, které spojují časy, kdy dochází k pulzu na této úrovni. Tuto vizualizaci můžeme vidět na obr. 1.11. Každou rytmickou úroveň lze charakterizovat svou *periodou*, *fází* a *typem*. Perioda určuje délku jednoho pulzu na této úrovni. Fáze určuje, na které době začíná doba úrovně vyšší. Typ určuje, na kolik podúrovní se úroveň dále dělí. Zde budeme uvádět dva základní typy: dvojitá a trojitá. Úrovně budeme číslovat od nejnižší (s nejmenší periodou) od nuly. Tabulka 1.1 popisuje periodu, fázi a typ rytmických úrovní z ukázky na obrázku 1.11.

Rytmické úrovně jsou tzv. *zarovnané*, tedy na každé rytmické úrovni mohu pracovat pouze s časovými událostmi, které jsou použity v předchozí úrovni. Z toho plyne, že všechny rytmicky důležité časy jsou určeny nultou úrovní, vyšší pouze rozhodují, jak mají být seskupeny. Z této obloučkové notace získáváme všechny informace pro zanesení rytmických údajů do hudební notace. Nejvyšší úroveň nám udává délku taktu, kterou v hudební notaci znázorňuje *metrum*. Metrum zapisujeme těsně za klíč notové osnovy a skládá se ze dvou čísel. Čísla $\frac{a}{b}$ nám říkají, že takt se skládá z a prvků délky $\frac{1}{b}$ doby.

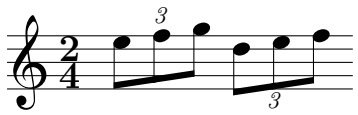


Obrázek 1.12: Použití trámců k notaci rytmu

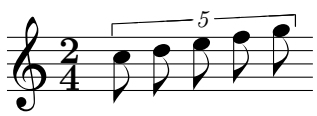
Např. taktu označenému symbolem $\frac{2}{4}$ říkáme dvou-čtvrťový. To znamená, že se takt skládá ze dvou čtvrťových dob (počítáme první-druhá). Na obr. 1.11 lze vidět, že první takt nemusí být úplný, tento kupříkladu začíná na čtvrtou dobu. Tomuto jevu říkáme *předtaktí*.

Další nástroj pro zvýraznění rytmických úrovní jsou *trámce*. Ty spojují osminové a kratší noty do skupin dle rytmu. Na obr. 1.12 vidíme dva různé způsoby notace šesti osminových not. První takt má přízvuk (pulz) první úrovně na každé druhé době, ve druhém taktu vidíme, jak bychom ztvárnili přízvuk na každé třetí době.

Ve skladbě jsou možné určité odchylky od rytmické struktury, např. zahráním sekvence rychlé sekvence not. Tento jev se nazývá dle počtu rychle zahraných not duola, triola, kvartola atd. Tyto noty se hrají tak rychle, aby vyplnily časové místo do délky taktu. Značí se číslicí počtu těchto not, které do této vychýlené sekvence patří:



Zde je příklad *kvintoly*, která vyplnila místo pro dvě čtvrťové noty:



Moderní hudební notace umožňuje zaznamenat mnohé další informace. Více jak sedm století jejího vývoje má za následek výraznou rozsáhlost tohoto notačního systému. Nabízí např. mnoho symbolů pro zaznamenání dynamiky skladby – pokyny, jak hlasitě má hudebník hrát. Artikulační značky (*staccato, tenuto, přízvuk, ...*) upřesňují, jak by se měla konkrétní nota zahrát. Melodické ozdoby doplňují melodii a ozdobují ji. Mezi ně patří např. příraz, odraz, trylek a další. Notační zápis také obsahuje několik možností zkrácení zápisu úseků, které se ve skladbě opakují. Z důvodu rozsahu je zde však rozebírat nebudeme a vystačíme si s výše popsanou podmnožinou hudební notace.

Kapitola 2

Analýza problému

V předchozí kapitole jsme si popsali vstup a výstup hudební transkripce a její obecnou definici. Nyní se zaměříme na způsob jejího řešení. Nejprve se podíváme na obecné přístupy k problému a na jejich vývoj. Dále si popíšeme souvislosti mezi návrhem transkripčního systému a percepčních modelů vnímání hudby. Protože problém transkripce je sám o sobě příliš rozsáhlý, rozdělíme si jej na několik podproblémů, se kterými se nám bude pracovat lépe. Tyto podproblémy si krátce popíšeme a ukážeme si způsob toku informace mezi nimi.

Pro rozsáhlost problému transkripce si také v poslední části této kapitoly přesně stanovíme rozsah řešení a provedeme si analýzu požadavků, které budeme mít na výsledný transkripční systém. Poté si popíšeme jaké důsledky tyto požadavky budou mít na návrh systému.

2.1 Analýza hudebních signálů

Hudební signál lze analyzovat z různých úhlů pohledu. Můžeme na něm např. zkoumat hudební motivy, harmonii, rytmus, hudební styl a další hudební parametry. Tomuto typu analýzy budeme říkat *vysokoúrovňová*, jelikož pracuje s pojmy, které nejsou matematicky definované a operuje na abstraktnějších úrovních jeho reprezentace. Na druhé straně přímou analýzu akustického signálu nazveme *nízkoúrovňovou*, jelikož nevyžaduje žádné informace o hudební podstatě díla (ačkoliv jí může být přínosná).

Hudební transkripce má za úkol ze signálu extrahovat mnoho různých informací. Je to složitý úkol, který vyžaduje analýzu na nízké i vysoké úrovni zároveň. Navrhnout algoritmus, který by na způsob překladového automatu transformoval vstupní signál na sekvenci not, je proto pravděpodobně nemožné. Způsob extrakce těchto informací navíc vychází z široké oblasti výzkumu. Proto je běžné si problém rozložit na jednodušší podproblémy, které budou specifičtější a snadněji řešitelné.

Výzkum hudební transkripce je značně roztříštěn způsobem její definice. Různé transkripční systémy totiž pracují s různými vstupy (akustický či digitální signál, MIDI, ...) a výstupy (melodická linka, MIDI, různé hudební notace, ...). Vstup může být navíc různě omezen z hlediska zdroje signálu. Některé transkripční systémy jsou omezeny na konkrétní nástroj/nástroje, konkrétní styl, monofonní či omezený polyfonní signál (např. bez kolize harmonických komponent), atd. Proto není jednoduché jednotlivé výsledky porovnávat, ani využívat jejich postupů. Dekompozici problému proto provedeme dostatečně podrobně, abychom mohli pro různé podproblémy využít co nejvíce z již existujícího výzkumu.

Jeden ze způsobů přístupů k řešení transkripčního problému je snaha napodobit vědomý proces lidské transkripce. Na základě poznatků psychologie hudby víme, že většina lidí neprovede transkripci na první poslech (sekvenčně). Namísto toho musí skladbu poslouchat vícekrát a iterativně extrahuje nejprve hudební strukturu, hlavní témata, harmonii a rytmické vztahy a následně provádí detailnější analýzu. Tomuto postupu budeme říkat *shora dolů* – od vyšších (abstraktnějších) struktur k nižším. (Hainsworth, 2004) Existují však lidé, kteří jsou schopni provést traksripci na první poslech: Např. Wolfgang Amadeus Mozart ve svých 14 letech po poslechu Gregorio Allegriho devítihlasé skladby Miserere ji dokázal téměř bezchybně zpaměti přepsat (Hainsworth, 2004, str. 92). Bob Milne, současný ragtimový pianista, se podrobil neurologickému pokusu, kde dokázal po prvním poslechu reprodukovat až čtyři symfonie zároveň (Masnick, 2017).

2.2 Vývoj hudební transkripce

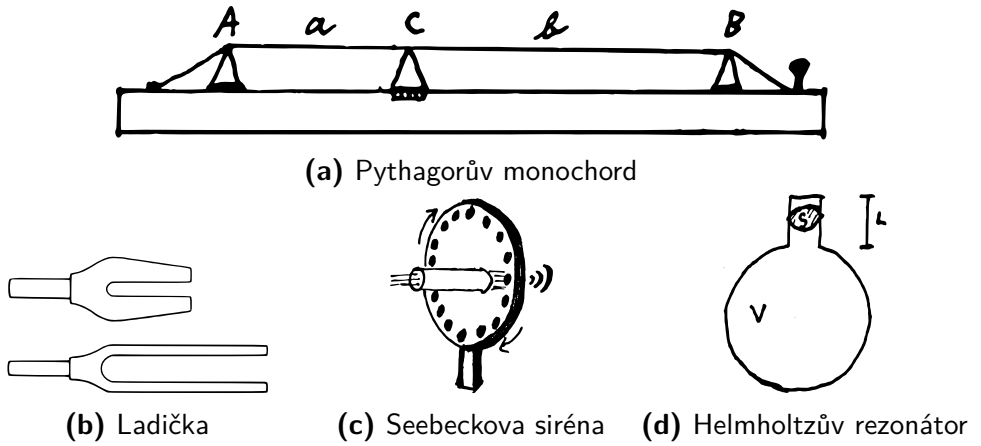
Cesta objevení nástroje provádějícího automatickou hudební transkripci se prolíná s cestou pochopení hudebního signálu, jenž byl odjakživa lákavým tajemstvím mnoha učenců. Na jejím počátku stojíme před zásadním problémem v hudební transkripci: jak lidský mozek vnímá výšku tónu.

Pythagoras v 6. stol. př. n. l. sestrojil *monochord*, jenž je považován za nejranější model vnímání harmonie: Nad ozvučnou desku je umístěna struna upevněná ve dvou bodech. Mezi ně je vložen posuvný můstek dost vysoký, aby napínal strunu (viz obr. 2.1a). Pokud je poměr délek mezi levou a pravou částí struny vyjádřitelný zlomkem malých čísel, bude rozvibrovaná struna vydávat libozvučný tón. Tento experiment také pomohl ukázat souvislost výšky vnímaného zvuku s jeho frekvencí, jelikož kratší struna kmitala rychleji a vydávala vyšší tón (obr. 2.1a). Ve 4. stol. př. n. l. pak dává Aristoxenos první jasnou definici výšky tónu a intervalů, tzv. *tonoi*. Až ve 13. století se však dostáváme k definici výšky, jak ji známe dnes, za pomocí Safi al-Dina. (A Cheveigné, 2008)

Kvalitativní souvislost frekvence a výšky byla tedy již tušena ve starověkém Řecku. Kvantitativní souvislost byla odvozena až v 1. pol. 17. stol. Marin Mersenne nejprve potvrdil teorii, že frekvence kmitání struny je nepřímo úměrné její délce. Pak natáhl strunu dostatečně dlouhou, aby byl schopen spočítat frekvenci její vibrace. Odvodil tak frekvence pro libovolnou výšku tónu a potvrdil jejich přímou souvislost. (A Cheveigné, 2008)

Precizní detekci výšky tónu umožnil až vynález *ladičky* (*tuning fork*, obr. 2.1b) v roce 1711 (Miller, 1916). Jde od dvouramennou tyč, vyráběnou nejčastěji z oceli, která ve kvalitním provedení generuje téměř čisté tóny. Frekvenci těchto tónů lze ovládat hmotností ramen ladičky. To umožňuje vytvořit celou sérii ladiček, která může sloužit k rozpoznání libovolné výšky tónu.

Až do poloviny 19. stol. však bylo pro kohokoli nepředstavitelné, že by výška tónu mohla být složena z více frekvencí. To ukázal až Fourier, který ve své práci z r. 1807 tvrdí, že libovolný periodický signál lze rozložit na součet sinusových funkcí různých frekvencí a fází. Takový převod se nazývá *Fourierova transformace*. Tento objev vyvolal převrat v chápání výšky tónu, přivedl však mnohé na scestí. Např. Ohm v r. 1843 říká, že „pokud má tón výšku



(a) Pythagorův monochord

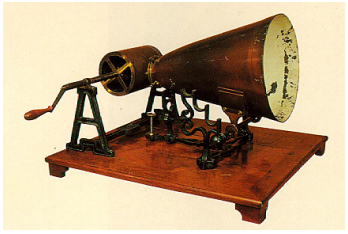
(b) Ladička **(c)** Seebeckova siréna **(d)** Helmholtzův rezonátor

Obrázek 2.1: Nástroje z počátků analýzy výšky tónu

o dané frekvenci, musí existovat ji odpovídající harmonická komponenta“. Tato chybná hypotéza, jež je součástí *Ohmova akustického zákona*, byla pro mnohé velmi lákavou a trvalo dlouho, než byla zcela opuštěna (A Cheveigné, 2008).

Seebeck v r. 1844 Ohmovo tvrzení vyvrátil, když sestrojil slavnou *Seebeckovu sirénu*: skrz konstantní rychlostí rotující disk, který má ve svém obvodu vyvrtané díry, byl prohánìn vzduch. Zvuk, který při průletu diskem vytvářel, byl pak analyzován z hlediska výšky. Bylo zjištěno, že výsledná výška není závislá ani na množství vzduchu diskem prohánìného, ani na velikosti děr, ale pouze na jejich množství (obr. 2.1c). Jelikož disk rotoval konstantní rychlostí, počet děr odpovídal frekvenci zvuku. Pomocí této sirény bylo např. potvrzeno, že dvojnásobná frekvence vede ke zvýšení o oktávu. (McLeod, 2009) Především však poukázal na jev *chybějící fundamentální frekvence*. Pomocí své sirény dokázal vytvořit harmonický zvuk složený pouze ze sudých harmonických koeficientů. Základní frekvence zde tedy chyběla, přesto však byl zaznamenán vjem jeho výšky (Jones, 2017).

První pokus o nástroj, který by sám prováděl detekci výšky tónu, je přisuzován Hermanu von Helmholtzovi, jenž v roce 1862 představil *Helmholtzův rezonátor*: jde o kulovitou dutinu se širokým hrdlem, většinou vyrobenou ze skla či mosazi. Z výšky a šířky hrdla a objemu dutiny lze odvodit tzv. *rezonující frekvence*, což je frekvence, ve které předmět rezonuje, zní li v blízkosti tón jí odpovídající výšky (obr. 2.1d). (McLeod, 2009)



(a) Fonoautograf (*Phonautograph*, 2015)

(b) Fonograf (Bruderhofer, 2005)

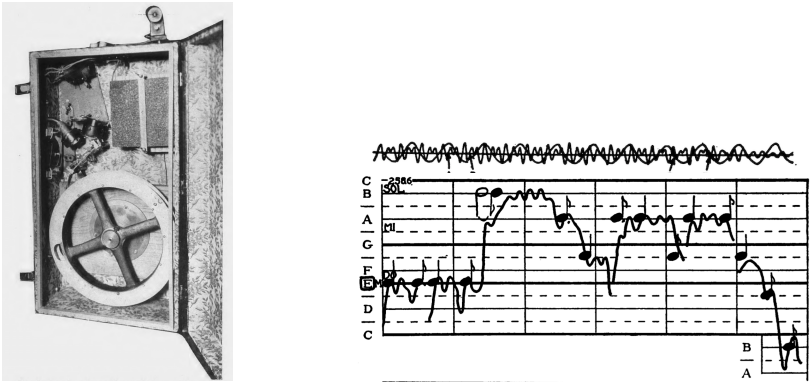
Obrázek 2.2: Nástroje k zaznamenávání zvuku

Pro detailnější analýzu zvukového signálu (např. výpočet Fourierovy transformace) však stále chyběl způsob, jak velmi rychlé kmitání vzduchu s dostatečnou přesností zaznamenat. Historicky první přístroj pro záznam zvuku byl *fonoautograf*, vynalezený v roce 1859 (Miller, 1916). Ten za pomocí tenké membrány stočené do tvaru trychtýře přenášel zvukové vibrace na tenkou jehlu. Ta byla přiložena k hladkému papíru, který byl pokryt jemnou vrstvou popelavého oleje (obr. 2.2a). Papír byl rovnoměrnou rychlostí posouván a jehla hýbající se v rytmu zvukových vibrací na něj tak vyrývala reprezentaci zvukové linky. Pro důkladnou analýzu signálu však tato metoda nebyla postačující. Aby zvukové vlny unesly jehlu, musela být velmi malá a tedy i vzniklý záznam byl velmi titěrný. Navíc třením a hybností jehly docházelo ke značnému zkreslení.

Telefon, vynalezený Bellem v r. 1876, převáděl zvuk na elektromagnetické vlny. Ty byly možné pomocí galvanometru vizualizovat na tzv. osciloskopu (Dudell v r. 1893). Kvůli mechanickým a elektromagnetickým vlivům však docházelo ke značným změnám zvukové linky.

Edisonův *fonograf* (později Bellův *gramofon*) z roku 1877 zvýšil přesnost záznamů z fonoautografu použitím odolnějších materiálů. Navíc tyto materiály nebyly pouhou vizualizací, nýbrž byly prostorové a umožnily tak nahraný zvuk i reprodukovat (obr. 2.2b).

Kýžené kvality vizuální reprezentace však dosáhl až přístroj zvaný *fonodeik* (později *foneloskop*), vynalezený Dayton Millererem v r. 1908 (Lanz, 1931). Historicky první pokusy o transkripci hudební skladby jinak, než podle lidského sluchu, používaly právě jej. Samozřejmě byl k transkripci stále potřeba člověk,



(a) Tělo zvukového fotoaparátu **(b)** Transkripce pomocí fonodeiku

Obrázek 2.3: Přenosný fotoaparát zvukového signálu fungující na principu fonodeiku (Metfessel et al., 1928)

nicméně namísto spoléhání se na sluchový aparát byla používána velmi detailní vizualizace zvukové linky. Na základě této metody Milton Metfessel (1928) vytvořil *přenosný fotoaparát zvukového signálu* (obr. 2.3a). Ten se skládal z membrány, která přenášela zvukové vibrace na zrcadlo. Na ty byl zamířen paprsek světla tak, aby jejich odraz dopadal na rozvinutý fotografický film. Díky pohybu zrcadel napojených na membránu kopírují odražené paprsky křivku zvukového signálu. Při stálém posouvání filmu se tímto dosahuje vizualizace zvukové linky ve velmi dobré kvalitě. Hudební transkripce tohoto záznamu pak vyžadovala náročnou ruční analýzu vyvolaného filmu. Na základě podobnosti opakujících se úseků zvukové linky při ní zjišťovali periodu signálu, ze které již šla odhadnou výška tónu v daném čase (obr. 2.3b). Tuto metodu, kterou zde velmi pracně prováděl člověk, se budeme snažit zautomatizovat v kap. 3.5.2.1.

Ve 20. letech 19. stol. sestavil Backhaus pomocí filtrů úzké pásmové propusti, mikrofonu a zesilovače systém, který dokázal znázorňovat jednotlivé frekvenční komponenty. Vývoj osciloskopu ve 40. letech 19. stol. pak přinesl novou vlnu výzkumu. Ta přinesla např. první mechanické nástroje vypočítávající Fourierovu transformaci, která se až doposud musela vypočítávat z ručně. (J. A. Moorer, 1975, str. 12).

První přístroj provádějící automatickou detekci výšky tónu se nazýval *melograf* a první pochází z r. 1951. Používal sérii pásmových propustí šířky třetiny oktávy, u kterých každé 4 milisekundy hledal maximum. Na jeho výstupech pak pomocí analýzy počtu protnutí nuly detekoval výšku tónu. (J. A. Moorer, 1975, str. 19) Později vzniklo mnoho variant tohoto přístroje a jejich vizualizační schopnosti i schopnosti detekce výšky se postupně zlepšovaly (Nonken, 2014, str. 65).

V průběhu 19. stol. se díky těmto stále se zdokonalujícím nástrojům vyvinulo velké množství technik, které dodnes při hudební transkripci používáme. Největší důraz byl kladen na detekci výšky tónu (Hess, 1983). Objevují se ale i první algoritmy pro detekci tóniny, např. Longuet-Higgins a Steedman z r. 1971 (Temperley, 1999). Výrazným vývojem prošel též výzkum vnímání rytmu a jeho shrnutí lze nalézt v Bengtssonově práci (1977). Mnohé z těchto metod byly však pro danou dobu výpočetně příliš náročné a našly uplatnění teprve když výpočetní výkon dostatečně vzrostl (Askenfelt, 1976).

Důležitým mezníkem ve vývoji transkripčních systémů je rok 1965, kdy Cooley a Tukey publikují algoritmus rychlé Fourierovy transformace (Cooley et al., 1965). Ten dosáhl zrychlení výpočtu diskrétní Fourierovy transformace z původního přímého výpočtu v časové složitosti $\mathcal{O}(N^2)$ na časovou složitost $\mathcal{O}(N \log N)$. Jelikož obor zpracovávání signálů byl tehdy velmi omezován výpočetním výkonem tehdejších počítačů, takové zrychlení pro něj znamenalo výrazný přelom.

První systém automatické hudební transkripce prováděné počítačem je od Jamese Moorera z r. 1975. Jeho transkripční systém byl omezen na noty delší než 100 ms, hrané maximálně dvěma nástroji. Noty se navíc nesměly překrývat ve smyslu, že fundamentální frekvence jedné nesměla odpovídat žádné harmonické komponentě druhé. Detekci výšky prováděl dvoufázovým použitím hřebenového filtru (rozdílové funkce), kde první fáze slouží pro odhad silných frekvencí. Frekvence, jež se ukázaly silné, poté vyfiltroval sérií pásmových filtrů. Na výstup každého filtru pak použil opět hřebenový filtr pro detailní analýzu výšky. Segmentace pak byla provedena heuristickým ohodnocením jednotlivých frekvenčních komponent a jejich následné sjednocení do not. (J. A. Moorer, 1975)

2.3 Současný stav řešení hudební transkripce

Přes značnou snahu vyřešit problém hudební transkripce, prakticky využitelný transkripční systém pro všeobecné použití stále neexistuje (Klapuri, 2004). To platí jak pro polyfonní, tak i pro monofonní signály. Ačkoliv automatická detekce výšky monofonních signálů je z teoretického hlediska považovaná za vyřešenou (Klapuri, 2006), monofonní transkripce má stále výrazné nedostatky, např. při segmentaci not (až 20% chybovost u segmentace zpěvu) (Clarisse et al., 2002). Obecně platí, že čím bohatější polyfonie signálu je, tím složitější je její transkripce (Klapuri, 2004). „State-of-art" transkripce však nelze nijak jednoduše shrnout z důvodu rozsáhlosti jeho definice, rozmanitosti vstupu a heuristickému přístupu řešení hudebních modelů. Popisu často používaných či uznávaných metod jsou věnovány následující kapitoly.

Zásadním problémem, se kterým se při tvorbě transkripčního systému potýkáme, je obtížnost extrakce percepčních informací (Hainsworth, 2004). Jde o ten typ problému (jako je také např. rozpoznávání hlasu či textu), kde se i základní koncepty, jenž jsou pro nás naprosto samozřejmé, v počítači velmi obtížně modelují. Lidská percepce výšky například přiřadí určitou výšku libovolnému signálu, s jedinou výjimkou bílého šumu (Syrový, 2013). Způsob, jakým člověk klepe do rytmu, nemusí v signálu korelovat s ničím výrazným (Rosenthal, 1992). Člověk též dokáže velmi snadno separovat části signálu, které vycházejí ze zdroje, který jej zajímají, ačkoliv separace části signálu je pro stroje velmi obtížná. Tento jev se nazývá *efekt koktejlového večírku* (Syrový, 2013).

Lidská sluchová soustava je pozoruhodný systém, jenž dokáže rozpoznat velmi drobné detaily, ale je zároveň odolný vůči výrazným odchylkám. Například, zvuk ze zbrusu nových prvotřídních houslí a ze starých houslí druhořadé výroby vytváří zásadně rozdílné signály. Přesto, pokud hudebník na ně zahraje určitou skladbu, člověk v obou interpretacích uslyší stejné hudební informace. Určité percepční vlastnosti se však těžko modelují, počítač často potřebuje přesnou definici, jak takové housle mají znít. Výše zmíněný problém vysokého rozlišení v kontrastu s velkou variancí bude třeba řešit na několika úrovních transkripce. Např. při detekci časů nástupů a délek not se tento jev objevuje, když hledáme přesnou rytmickou interpretaci v kontrastu s výskytem časových

nepřesností a dalších nedokonalostí. Člověk se s tímto problémem potýká tak, že při poslechu hudební skladby provede určité hypotézy na několika abstraktních úrovních (např. rytmické a harmonické), podle kterých se pokouší skladbu interpretovat (Rosenthal, 1992).

Hainsworth (2004) popisuje další zásadní problém při extrakci hudebních informací, kterým je využívání informací získané vysokoúrovňovou analýzou při analýze nízké úrovně. Algoritmy zpracovávání signálu, které by měly odpovídat transformacím informací v lidském uchu a podvědomí, by měly být schopny využít kontext hudebních struktur ve skladbě, popsaných ve výše zmíněných hypotézách. Toto bylo potvrzeno např. experimentem, kde u skladby hrané při stálé hlasitosti, určité noty působily na posluchače hlasitějším dojmem pouze díky melodickém kontextu, ve kterém se nacházely (Klapuri, 2004).

Dalším výrazným problémem je rozmanitost validních vstupních signálů. Hudba přepsatelná do not se může vyskytovat v nespočtu různých podob a pocházet z nesmírného množství zdrojů (Jiří Stivín je např. znám hrou na záchodovou trubku) a hudebních stylů (zejména folklorní díla bývají problémem). Mnohé transkripční systémy jsou proto specializovány na konkrétní hudební nástroj, původ díla, či mají technicky popsanou množinu signálů, jenž jejich transkripční systém podporuje (Klapuri, 1997).

Nabízí se otázka, zda je vůbec možné vytvořit obecný transkripční systém, který bude nezávislý na vstupu a nebude se muset pro daný hudební nástroj či styl kalibrovat. Naštěstí existuje jeden příklad takového systému a tím je člověk. Pokud bychom dokázali „naučit“ stroj vnímat a chápat hudbu jako člověk, transkripce by byla mnohem snazší.

2.4 Návaznost na percepční modely

Jelikož je lidská sluchová soustava dosud jediný systém, který je schopen provést transkripci správně, je přirozené se pokusit ve strojovém systému lidský přístup napodobovat. Nízkoúrovňová analýza odpovídá vnímání a snaží se napodobit *percepční modely* lidského vnímání. Vysokoúrovňová zase odpovídá chápání hudby a imituje tak *hudební modely*. Ačkoliv je téměř každý člověk schopen rozpoznat např. rozdíly ve výšce tónu a nástupy not, důkladnému

porozumění hudebních struktur potřebnému pro transkripci se musí i člověk roky učit. (Hainsworth, 2004) Vymodelovat takový systém v počítači je velmi složité též z důvodů abstraktně definovaných percepčních pojmů, na nichž hudební model pracuje. Často platí pravidlo, že na čím vyšší úrovni analýzy se pohybujeme, tím více je heuristická a obtížněji teoreticky obhajitelná (J. A. Moorer, 1975).

Klapuri (2004) podotýká, že je důležité si uvědomit, že hudební notace je primárně zaměřena na mechanickou produkci zvuku, ne jeho vnímání. Problém hudební transkripce a problém předpovědi odezvy hudebního signálu na člověka se fundamentálně liší. Poukazuje na příklad, kdy se hudební notace a hudební percepce naprosto rozchází: pokud je ve skladbě hrána nota několika různými nástroji zároveň, hudební notace popisuje každý tento nástroj zvlášť, zatímco hudební percepce tyto zvuky sloučí do tzv. *chimérického* zvuku. Současné nejlepší transkripční systémy přesto vychází z percepčních a hudebních modelů (Klapuri, 2004). Většina problémů s využitím percepčních modelů se totiž týká pouze polyfonní analýzy. Jsou také vysvětlením, proč většina monofonních systémů nelze při polyfonní analýze využít.

Z důvodu rozsahu této práce a spornosti přínosu percepčních modelů je zde proto důkladně popisovat nebudeme. Metody transkripce budeme analyzovat z hlediska běžně používaných algoritmů, u nichž budeme případně diskutovat jejich návaznost na percepční modely. Při inspiraci z psychoakustických percepčních modelů je však nutné mít na paměti, že nejsme schopni simulovat výkon lidského mozku, ani přesně nevíme, jak funguje. Navíc tyto modely obsahují mnoho aspektů, které vysvětlují různé fyziologické vlastnosti a pro samotnou transkripci nemusí být relevantní. Bude-li nějaká z metod inspirována percepčním modelem, bude na místě diskuze, zda kýženou výhodu opravdu přináší.

2.5 Reprezentace hudební struktury

Při vysokoúrovňové analýze hledáme hudební struktury nad reprezentací hudebních událostí. Tyto struktury mají pevně daný způsob reprezentace v hudební notaci. Ta je však definovaná pouze vizuálně, a proto není praktická pro práci s hudebními strukturami, kdy potřebujeme v reálném čase hudební strukturu číst i upravovat.

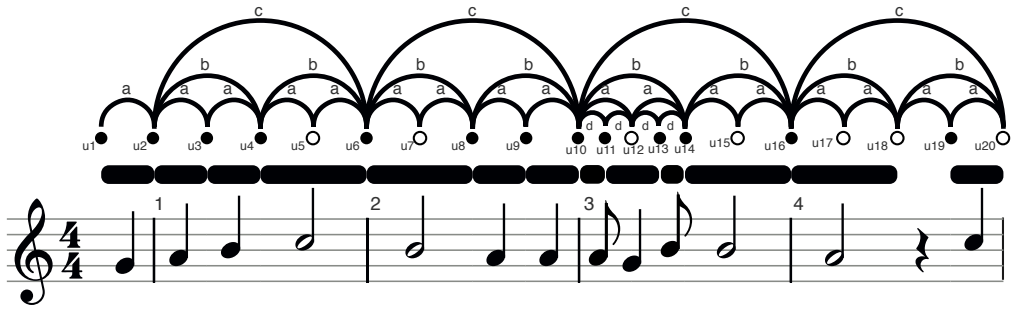
Cílem této kapitoly je definovat reprezentaci, která bude vhodně modelovat hudební strukturu nad reprezentací hudebních událostí. Požadavky na tuto strukturu budou následující:

1. **Paměťová efektivita** – naše reprezentace by neměla zvýšit paměťovou složitost reprezentace hudebních událostí.

2. **Snadný převod do hudební notace** – převod do moderní hudební notace by neměl vyžadovat žádné složitější výpočty, jednotlivé entity by měly být mapovatelné ve vztahu 1:1.

3. **Efektivní přístup k jednotlivým rytmickým úrovním** – pro vysokoúrovňovou analýzu je rytmický kontext velmi důležitý. Proto by cílová datová struktura měla efektivně podporovat operace čtení a přidání not nad jednotlivými rytmickými úrovněmi.

Nechť n hudebních událostí, nad nimiž máme datovou strukturu postavit, je posloupnost $(\mathcal{E}_i)_{i=0}^{n}$. Na systém rytmických úrovní lze nahlížet jako na *orientovaný acyklický graf* $G = \{U, H\}$, jehož množina vrcholů U bude označovat jednotlivé události nástupů not z $\mathcal{E}$, tedy množina $\{\mathcal{E}_i : 2 \big| i\}$ (každý sudý index). Konkrétně vrcholem bude samotná *nota*, která bude obsahovat informace o čase nástupu, délce noty, její výšce a její čistotě. Hodnota délky bude zpočátku nevyplněná, doplní se zpětně po příchodu události uvolnění dané noty. Hrany množiny H budou reprezentovat jednotlivé rytmické úrovně – tempa, ve kterých by si člověk mohl klepat do rytmu. Hrana mezi dvěma vrcholy se bude vyskytovat právě tehdy, když nástupy těchto not reprezentují tyto události „klepnutí“. Jelikož hudební skladby nemají ve všech časových intervalech rytmických úrovní nástup noty, je třeba na chybějící místa doplnit uzly, které budou tyto prázdné doby reprezentovat. Ty budeme nazývat *skryté noty*.



Obrázek 2.4: Graf nad notami hudební skladby

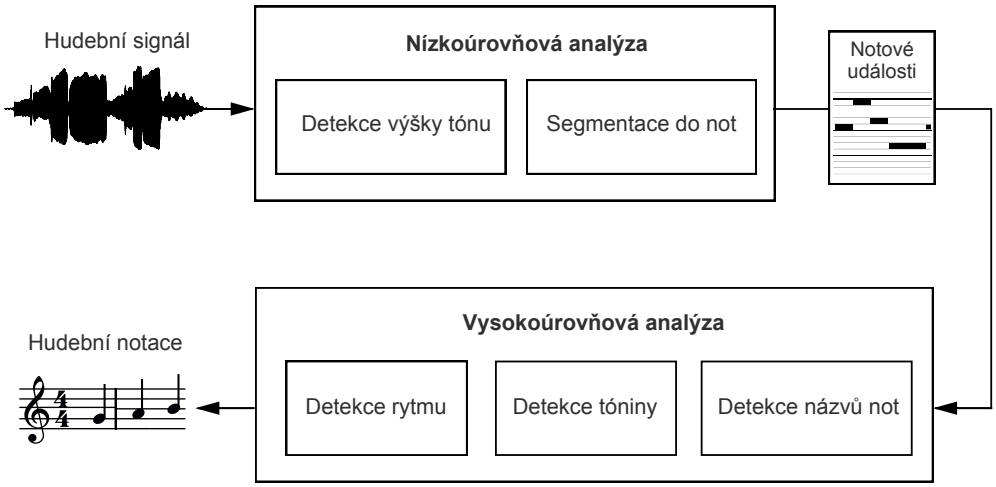
Obr. 2.4 ukazuje příklad této struktury nad krátkým úryvkem hudební skladby. Pro názornost je pod grafem zobrazena stejná skladba v piano roll notaci a hudební notaci. Hrany grafu jsou označeny rytmickou úrovní, jež zde pro přehlednost reprezentujeme malými písmeny. Nejvyšší rytmická úroveň, v grafu označená písmenem c, odpovídá taktu skladby. Jednotlivé noty jsou reprezentovány černými kolečky, skryté noty pak bílými kolečky.

Rytmická analýza je většinou popisována třemi základními rytmickými úrovněmi: *taktem*, *taktusem*, *tatumem* (viz kap. 5). Na obrázku tyto úrovně odpovídají hranám s hodnotami a, b a c. Na začátku třetího taktu se vyskytuje rytmická nepravidelnost, tzv. *synkopa*, jež nespadá do mřížky těchto tří úrovní. Proto se na tomto místě dočasně vynořuje i čtvrtá úroveň označená d.

Ukažme si, že tato struktura splňuje naše požadavky:

1. **Paměťová efektivita:** Jelikož je počet rytmických úrovní, který je člověk schopen vnímat, omezený (Rosenthal, 1992), počet hran je asymptoticky shodný s počtem uzlů. Skryté noty lze navíc v reprezentaci grafu vynechat tzv. kondenzací: pokud mezi uzly u a v, které reprezentují běžné noty je k skrytých not, lze tyto uzly z grafu odstranit a s nimi incidenční hrany nahradit hranou $u - v$, která ponese informaci o k skrytých uzlech. Každá hudební událost pak přidá maximálně jeden uzel, tedy celková paměťová složitost je $\mathcal{O}(n)$.

2. **Snadný převod do hudební notace:** Z obr. 2.4 je zřejmé, že pokud grafová struktura splňuje rytmická pravidla, lze z ní odvodit délky jednotlivých not. Informace o výšce not je uložena v jim náležících uzlech. To dává dostatek informace k převodu not do hudební notace.



Obrázek 2.5: Schéma dekompozice na dvouúrovňovou analýzu

3. **Efektivní přístup k jednotlivým rytmickým úrovním:** Jelikož jde o acyklický graf, jednotlivé úrovně lze reprezentovat jako spojový seznam, pro nějž lze snadno požadované operace implementovat efektivně.

2.6 Dekompozice problému

Kromě výše uvedeného dělení analýzy signálu na vysokoúrovňovou a nízkoúrovňovou je praktické rozdělit si problém na několik samostatných podproblémů, na které se lze dívat odděleně. Hlavní motivací pro tuto dekompozici je samostatný výskyt řešení těchto podproblémů v literatuře, ačkoliv cílem samotná transkripce nebyla. Např. detekce výšky signálu je problém, jehož řešení bylo silně motivováno kvůli svému využití v analýze řeči. Mnoho vzniklých metod lze však využít i pro hudební transkripci.

Další motivací pro abstraktní rozdělení na jednotlivé podproblémy může být též podobné rozdělení v lidském vnímání. V literatuře najdeme mnoho důkazů modularity jednotlivých úloh v lidském mozku, kdy určité části jsou vykonávány v oddělených částech mozku. Např. rozpoznávání výšky tónu je vykonáváno v levé hemisféře, zatímco analýza rytmu a časových souvislostí se odehrává v pravé hemisféře. (Klapuri, 2004).

Na jednotlivé podproblémy se budeme dívat jako na samostatné výpočetní moduly. Transkripční systém pak bude určité propojení těchto modulů, jehož vstupem bude hudební signál a výstupem hudební notace. Dekompozici problému transkripce provedeme v této práci následovně:

1. **Nízkoúrovňová analýza**

    a) **Detekce výšky** (pitch detection) – tento modul má za úkol detekci percepčního vjemu výšky v závislosti na čase. Jde o notoricky známý problém motivovaný analýzou řeči a řešený již více jak 150 let (McLeod, 2009).

    b) **Detekce not** (onset, offset detection) – úkolem tohoto modulu bude detekovat percepční nástup a uvolnění (začátek a konec) not. Detekce nástupů je v posledních letech velmi zkoumaný problém, jehož výzkum byl velmi motivován pro jeho aplikaci v kompresi, indexování a ve vytěžování informací (Bello; Daudet et al., 2004, str. 1). Detekci uvolnění oproti tomu bylo zatím věnováno pouze málo pozornosti (Coler et al., 2014, str. 2). Pro obecné hudební signály se segmentace not stále nepovažuje za vyřešený problém (Klapuri, 2004, str. 2).

2. **Vysokoúrovňová analýza**

    a) **Detekce rytmu** (rhythm tracking) – tento modul má na starosti porozumět rytmické struktuře nad sekvencí not. Jeho úkolem bude extrahovat tempo, metrum, formu hierarchické rytmické struktury a rytmickou roli každé noty.

    b) **Detekce tóniny** (key detection) – tónina nám poskytuje systém, v jehož kontextu nám dávají výšky jednotlivých not hudební význam. Tento modul se pokusí provést analýzu použitých výšek not a rozdělit hudební skladbu na segmenty stejné tóniny.

Na rozmezí nízké a vysoké úrovně transkripce použijeme reprezentaci hudebních událostí popsanou v kap. 1.2.4. Tato reprezentace je vhodná, jelikož na malém datovém prostoru uchovává dostatek informací pro vysokoúrovňovou analýzu a všechny potřebné informace jsou z ní snadno čitelné.

Vysokoúrovňová analýza se provádí na výstupu z analýzy nízkoúrovňové. Tok informací mezi výsledky jednotlivých modulů by však neměl být nijak omezovaný. Jak jsme však diskutovali v kap. 2.3, nízkoúrovňové moduly by měly využívat hudebního kontextu popsaného vyšší úrovní podobně, jak tomu je u percepčních modelů. Využití muzikologických informací na nižších úrovních bylo běžné již v počátcích hudební transkripce. Tehdy bylo hlavní motivací omezit rozsah výpočtu na základě pravděpodobností výskytu různých sekvencí not (J. A. Moorer, 1975). Lidský mozek upřednostňuje hudební informace, které pro něj mají v daném kontextu muzikální smysl a v případě jejich neexistence je schopen si tyto informace i domýšlet (Klapuri, 2004). Proto je na místě provádět v nízkoúrovňové analýze vážení dle jejich významu pro vyšší úroveň analýzy.

V následujících podkapitolách si popíšeme úkol, vstup a výstup jednotlivých modulů.

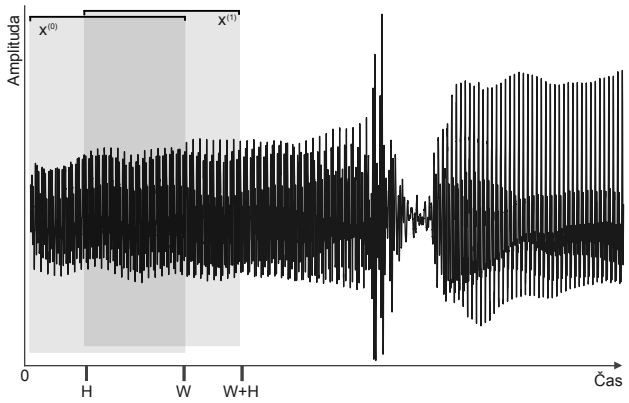
2.6.1 Nízkoúrovňová analýza

Úkolem tohoto modulu je převést digitální akustický signál, popsaný v kap. 1.2.3 do reprezentace hudebních událostí (kap. 1.2.4). Její hlavním úkolem je detekovat výšku, nástupy a uvolnění not. Tuto práci budou mít na starosti moduly detekce výšky a segmentace not.

Vstupem tohoto modulu je tedy posloupnost $(s_t)_{t=0}^{\infty} : s_t \in \mathbb{R} : s_t \in [-1, 1]$. Hodnota s_t odpovídá normované hodnotě t-tého vzorku od počátku signálu, tedy v čase $\frac{t}{f_s}$s.

Vstupem jednotlivých submodulů bude sekvence *oken* signálu délky $W \in \mathbb{N}$. Velikost okna W je důležitý parametr jednotlivých algoritmů a budeme jej diskutovat při jejich analýze. Dalším parametrem algoritmů bude velikost skoku (*hop size*) $H : H \in [1, W]$, o který budeme okno v jednotlivých iteracích posouvat (obr. 2.6). Parametr H nám umožňuje upravovat časovou přesnost signálu v intervalu od původní $\frac{1}{f_s}$ až na omezení přesnosti na $\frac{W}{f_s}$.

Dále si definujeme posloupnost $(x_i^{(j)})_{i=0}^{W} : x_i^{(j)} \in \mathbb{R} : x_i^{(j)} \in [-1, 1]$ jako hodnotu i-tého vzorku j-tého okna signálu: $x_i^{(j)} = s_{jH+i}$, kde $i = 0, 1, 2, \ldots, W, j = 0, 1, 2, \ldots$ Pokud nebude pozice okna v signálu důležitá, budeme horní index opomíjet.



Obrázek 2.6: Parametry délka okna W a velikost skoku H

2.6.1.1 Detekce výšky

Modul detekce výšky provede pomocí algoritmu detekce výšky (*PDA*) odhad průměrného vjemu výšky po dobu daného okna signálu (z každého okna jedna hodnota, viz obr. 2.7).

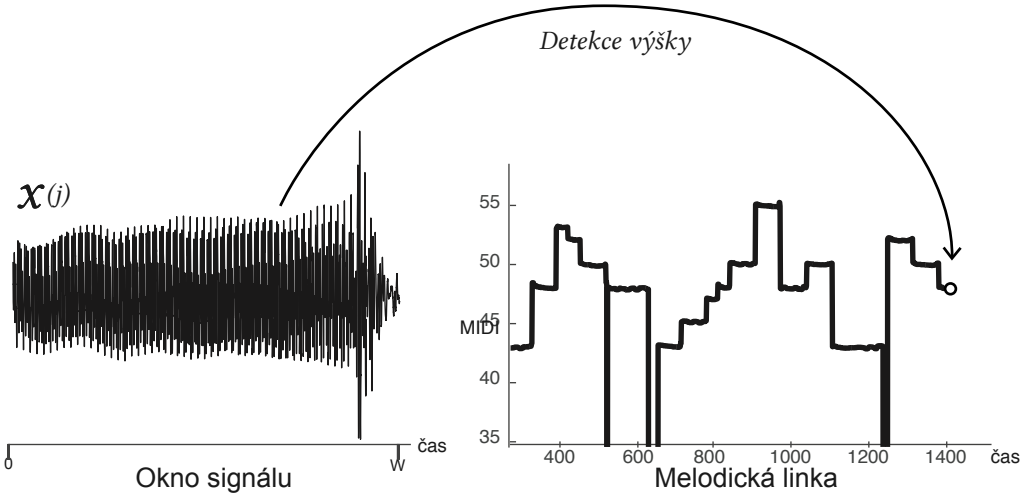
Vstup: Okno hudebního signálu $(x_i)_{i=0}^{W} : x_i \in \mathbb{R} : x_i \in [-1, 1]$.

Výstup: $\mathbb{R} \times \mathbb{R}$: dvojice výška tónu a jeho *čistota*. Výšku udáváme dle standardu MIDI:

$$n(f) = 69 + \log_2\left(\frac{f}{440}\right) \tag{2.1}$$

kde f je frekvence výšky daného okna signálu dle definice v kap. 1.2.1. Výška tónu se doplňuje informací o jeho tzv. *čistotě*, což je reálné číslo v intervalu $[0, 1]$, která udává, jak jednoznačně algoritmus tuto výšku určil. Měl by odpovídat subjektivnímu vjemu této veličiny, jenž se popisuje jako „jasnost“ či „čirost“ tónu (McLeod, 2009).

MIDI standart definice výšky předpokládá temperované ladění při výšce *koncertního A* rovné frekvenci 440 Hz (McLeod, 2009). Tato hodnota se od roku 1934 považuje za standard, nicméně dříve byla častá ladění koncertního A na frekvence 435 HZ, 456 HZ či dokonce 576 HZ (Syrový, 2013). Proto je nutné pro starší nahrávky tuto definici upravit. V poslední době je tendence toto ladění zvyšovat pro tzv. „zjasnění“ tónu (McLeod, 2009).



Obrázek 2.7: Ukázka vstupu a výstupu modulu detekce výšky

Hodnota $n(f)$ může nabývat libovolné reálné hodnoty, což umožňuje reprezentovat libovolnou výšku, ne jen noty pojmenované v kap. 1.2.1. Posloupnost hodnot $m_j = \mathcal{P}(x^{(j)})$ nazýváme *melodickou linkou* (obr. 2.7).

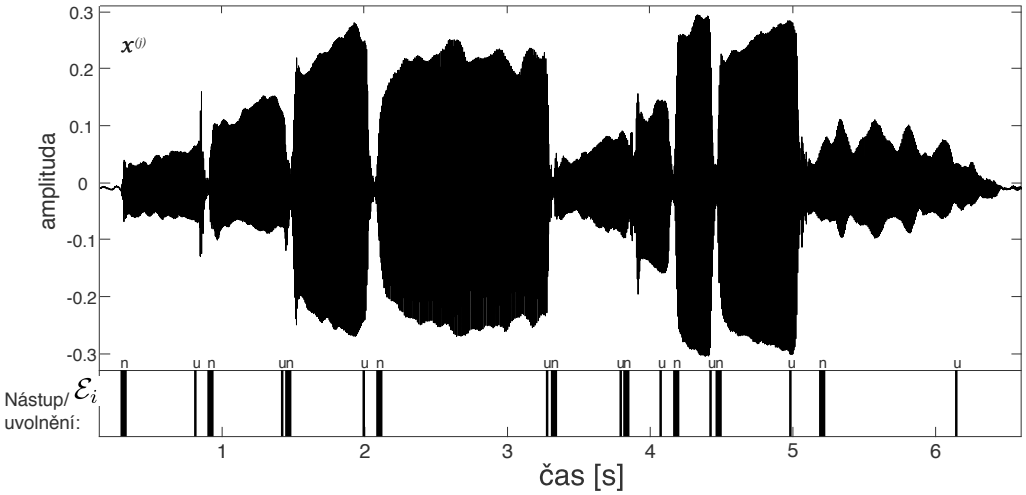
Pro vstupní signál, jenž nemá definovanou výšku, či je síla tohoto signálu příliš malá, může algoritmus vracet nedefinované hodnoty. Toto zjednodušuje implementaci těchto algoritmů, na druhou stranu vyžaduje ošetření těchto mezních případů odděleně.

2.6.1.2 Detekce not

Tento modul má za úkol spravovat datovou strukturu, která na vstupu přijímá okna signálů délky W (v časové přesnosti dané velikostí skoku H) a která spolehlivě vrací informace o *nástupu* (onset) a *uvolnění* (offset) not v signálu. Tímto výstupem provede segmentaci vstupu na vzájemně disjunktní notové segmenty. Parametr algoritmu L určuje, v jaké maximální latenci může algoritmus vrátit informaci o hudební události. Formálněji:

Vstup: Posloupnost oken hudebního signálu:

$$(x_i)_{i=0}^{W} : x_i \in \mathbb{R} : x_i \in [-1, 1]$$



Obrázek 2.8: Ukázka vstupu a výstupu modulu segmentace not

Výstup: Posloupnost hudebních událostí $(\mathcal{E}_i)_{i=0}^{\infty}$: kde $\mathcal{E}_i \in \mathbb{N}$ odpovídá času, kdy nastala i-tá událost. Události se střídají v pořadí nástup, pak uvolnění, atd., tedy času $i = 0, 2, 4, \ldots$ odpovídají nástupům a liché $i = 1, 3, 5, \ldots$ odpovídají uvolnění not. Čas události je v celočíselných hodnotách, kde i-tá hodnota říká, že nástup/uvolnění nastalo v čase $\mathcal{E}_i$-tého vzorku. Tuto informaci musí systém vrátit nejpozději v čase $\mathcal{E}_i + L$.

2.6.2 Vysokoúrovňová analýza

Z nízkoúrovňové analýzy máme dostatek informací pro převod na reprezentaci hudebních událostí. Každá informace o nástupu a uvolnění noty se zkombinuje s informací o její výšce a čistotě.

Úkolem tohoto modulu je tzv. *kvantizace* not, tedy jejich časové zarovnání do mřížky, jež je definována hudební notací (viz kap. 1.3). Jeho dalším úkolem je detekce rytmu a tóniny.

Vstupem tohoto modulu je výše zmíněná reprezentace hudebních událostí. Výstupem je speciálně navržená reprezentace not, jež je přizpůsobená snadné rytmické analýze a převodu do hudební notace (popsaná v kap. 2.5).

2.6.2.1 Detekce rytmu

Modul detekce rytmu provede rytmickou analýzu nad hudebními událostmi. Rytmickou analýzou je zde myšleno nalezení takové interpretace hierarchické rytmické struktury nad vstupní sekvencí, jež znázorňuje, jak by tato sekvence byla rytmicky vnímána člověkem.

Vstup: Posloupnost hudebních událostí $\mathcal{E}$.

Výstup: Graf hudební struktury definované v kap. 2.5. Jelikož je tento graf acyklický a má omezený počet typů hran, budeme každou rytmickou úroveň reprezentovat jako spojový seznam uzlů. Každý uzel bude reprezentován časem nástupu jemu odpovídající noty. Tento spojový seznam nám umožní snadné přidání dalších not při analýze v reálném čase. Pro převod rytmické struktury do hudební notace použijeme druhou reprezentaci grafu: seznam sousedů. Protože k její reprezentaci jsou všechny potřebné reprezentace ve spojových seznamech rytmických úrovní, můžeme seznam sousedů při vykreslování notace vygenerovat a tudíž není nutné spravovat v reálném čase obě struktury.

2.6.2.2 Detekce tóniny

Detekce tóniny bude následovat rytmickou analýzu. Jejím vstupem bude sekvence not definovaných výškou a prominencí. Prominence bude vážená rytmickou rolí not, proto nejsou další informace o rytmické struktuře zapotřebí. Detekce tóniny má za úkol rozdělit vstupní sekvenci na disjunktní intervaly, po které člověk cítí příslušnost této sekvence k určité stupnici.

Vstup: Posloupnost not $\mathcal{N}^n$, kde $\mathcal{N}$ značí notu označenou MIDI výškou (viz kap. 2.6.1.1) a prominencí.

Výstup: $\mathbb{N}^n$, kde i-tý prvek značí příslušnost i-té noty na vstupu ke stupnici označenou tímto prvkem. Označení stupnic je v intervalu $[0, K_{max})$, kde K_{max} je počet různých stupnic a popíšeme si jej v kap. 6.

Na obr. 2.9 lze vidět vstup a výstup, kde číselné označení výšek not a stupnic je nahrazeno jejich názvem. Ve čtvrtém segmentu tohoto vstupu došlo ke změně tóniny.

$$\big\{\{A, G^{\#}, B, D, A\}, \{G, A, H, C, A\}, \{D, H, G^{\#}\}, \{E, D^{\#}, G^{\#}, E, B\}\big\} \\ \rightarrow \{H_{mol}, H_{mol}, H_{mol}, E_{mol}\}$$

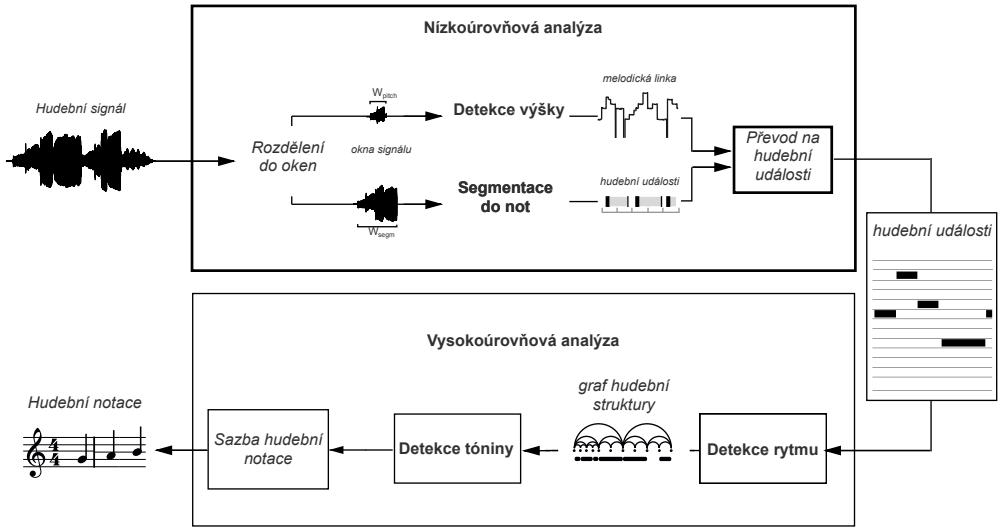
Obrázek 2.9: Ukázka vstupu a výstupu algoritmu detekce tóniny

2.6.3 Schéma transkripčního systému

Vzájemným propojením výše uvedených modulů vzniká transkripční systém. Schéma tohoto systému lze vidět na obr. 2.10. Kromě výše popsaných modulů jsou na něm vyobrazeny pomocné moduly *rozdělení do oken*, jehož princip jsme stručně popsali v kap. 2.6.1, moduly *převodu na hudební události* a *sazby hudební notace* nejsou v rozsahu této práce.

Na jednosměrný tok informací znázorněný na tomto schématu je třeba nahlížet pouze jako na vizualizaci původu těchto informací. Jednotlivé moduly by měly mít možnost využívat libovolné informace, které jsou v čase jejich vykonávání dostupné. Jak jsme diskutovali v kap. 2.3, z hlediska věrného modelování percepčního modelu v nízkoúrovňové analýze je žádoucí využívat hudební kontext představený vysokoúrovňovou analýzou.

Problém předávání informací není aktuální jen pro meziúrovňovou komunikaci. Klapuri (2004, str. 8) diskutuje s ní související pořadí vykonávání detekce výšky a segmentace not. Pokud např. segmentace not předchází detekci výšky, mohou algoritmy detekce výšky předpokládat stacionární povahu segmentovaného úseku signálu a využívat faktu, že úsek vychází z jedné noty (Bello; Monti et al., 2000, str. 6). Opačné pořadí zase umožňuje zpřesnit detekci nástupů not informací o změnách výšky tónu, které často doprovází nástup a uvolnění noty, používané např. v práci Clarisse a spol. (2002, str. 2). Dalším způsobem, který můžeme vidět v práci J. Vaase (2004, str. 11), je nejprve provést úlohy nízkoúrovňové analýzy a získané informace zkombinovat následovně.



Obrázek 2.10: Schéma transkripčního systému

Rozdělení na vysokoúrovňovou a nízkoúrovňovou analýzu má kromě úrovně abstrakce, na které daná analýza pracuje, další opodstatnění. Zatímco algoritmy nízké úrovně zpracují signál a extrahují z něj informace, které nás budou zajímat, a s původním signálem se dále nepracuje, vysokoúrovňová analýza uchovává sekvenci hudebních událostí neporušenou a hudební struktury modeluje nad ní. To jí umožňuje na základě vývoje okolností změnit minulá rozhodnutí, což je pro mnohé hudební modely žádoucí.

2.7 Rozsah řešení

Hlavním cílem této práce je navrhnout monofonní transkripční systém, který by bylo možné implementovat v jednoduché mobilní aplikaci. Popišme si nyní hlavní funkční požadavky této aplikace, jež budou sloužit jako reference k rozhodnutím, které návrh transkripčního systému doprovází.

1. **Úplnost transkripce** – transkripční systém v aplikaci by měl maximálně využívat prvky hudební notace popsané v kap. 1.3.2. Jinými slovy, pokud se v signálu objeví prvek, jenž je reprezentovatelný touto notací, transkripční systém by jej měl vykreslit.

2. **Flexibilita vstupu** – systém by měl být schopen provádět transkripci *monofonních* hudebních signálů vycházejících z libovolného obvyklého *harmonického* hudební nástroje.

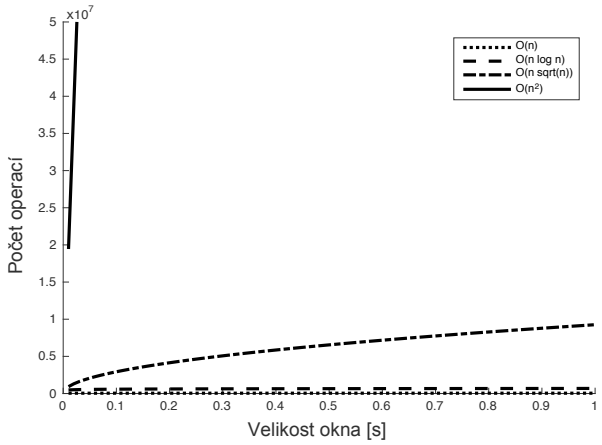
3. **Transkripce v reálném čase** – transkripční systém by měl transkripci provádět v reálném čase. Jinými slovy by v libovolném momentě měla být zobrazená hudební notace dosud nahraného signálu, jako kdyby systém v daném čase transkripci ukončil a vyžadoval výstup. Maximální délka odezvy systému by na zařízeních, jež podporují nejmodernější operační systém, neměla přesáhnout čtvrt sekundy.

4. **Tolerance zvukových nedostatků** – při vývoji transkripčního systému by měl být dán důraz na odolnost algoritmů vůči šumu a zvukům z pozadí, které se dají u využití transkripce v mobilní aplikaci předpokládat.

Z důvodu rozsahu jsme se omezili pouze na podmnožinu moderní hudební notace, která byla představena v kap. 1.3.2. Vynechané prvky mají ten charakter, že nejsou nutnou součástí vybraného základu, spíše jej určitým způsobem doplňují a obohacují. Byly vybrány tak, aby nám jejich absence neporušovala vlastnosti transkripce definované v kap. 1.1. Po vytvoření transkripčního systému fungujícího na této podmnožině lze modulárně přistupovat k jednotlivým chybějícím prvkům a transkripční systém jimi obohacovat.

Omezení na monofonní signály je motivováno především rozsahem této práce a jejím cílem, který by nebyl za současného stavu řešení transkripce polyfonních signálů možný (McLeod, 2009). Vynechání perkusních nástrojů z transkripce je z důvodu odlišnosti v přístupu k takovým nástrojům. Ze stávající analýzy by používaly pouze segmentaci a rytmus, navíc by se musela provádět detekce těchto nástrojů a použití jim přizpůsobené hudební notace, což by zbytečně komplikovalo již velmi rozsáhlý problém.

Požadavek na výstup v reálném čase s sebou nese závažné důsledky. Mnoho známých algoritmů transkripce je navrženo s fází preprocessingu a postprocessingu, jež jsou v aplikaci pracující v reálném čase těžko implementovatelné. Jednotlivé kroky úprav algoritmů pro fungování v reálném čase budeme diskutovat v dalších kapitolách.



Obrázek 2.11: Závislost počtu operací na velikosti okna a složitosti algoritmu

Dalším důsledkem výstupu v reálném čase je požadavek na efektivitu těchto algoritmů, aby bylo možné splnit požadovanou dobu odezvy. Za použití běžné vzorkovací frekvence $f_s = 44100$ a velikosti okna w sekund, nechť nízkoúrovňový algoritmus zpracuje toto okno v časové složitosti $f(44100w)$. Pak musí ve výpočetním čase, jenž přístroj přiřadí aplikaci v každém časovém úseku délky jedné sekundy stihnout kromě obsluhy uživatelského prostředí $f(44100w)$ operací nad desetinnými čísly. Z toho plyne $\frac{f(44100w)}{w}$ operací za sekundu, což například pro kvadratickou složitost a okno délky půl sekundy dává požadavek na circa miliardu operací za sekundu, což není proveditelné. S klesající délkou okna či klesající časovou složitostí se toto číslo snižuje. Na obrázku 2.11 lze vidět závislost počtu operací na velikosti okna na několika základních funkcích. Obecně budeme u algoritmů tedy diskutovat jeho efektivitu, pokud jeho časová složitost $f(n) = o(n \log n)$, nebo pokud jím požadovaná délka okna $w \geq 1$ s.

Tolerance šumu je často adresovaný problém v oboru zpracování signálu. V naší aplikaci můžeme očekávat celkem nízkou kvalitu záznamu způsobenou nahráváním v nepříznivém prostředí z důvodu snadné přenositelnosti zařízení a obtížnosti nastavení správného směru mikrofonu. Proto při volbě algoritmů zohledníme jejich odolnost proti šumu, jak teoretickou, tak na základě testování na reálných datech.

Kapitola 3

Detekce výšky tónu

Výška tónu je percepční vlastnost zvuku, jež nám jej umožňuje uspořádat na stupnici od nejnižšího po nejvyšší. Jiná definice říká, že jde o takovou frekvenci, kterou danému signálu přiřadí průměrný posluchač při porovnávání s čistým tónem dané frekvence. Z hudebního hlediska jde o důležitý atribut každé noty, který muzikant může ovládat. Problém automatické detekce výšky je díky motivaci pro problém rozpoznání řeči již řešen déle než 50 let (Rabiner et al., 1976). Přesto však dodnes zůstává jedním z nejtěžších problémů analýzy hudebních signálů (Sun, 2002).

V následujících kapitolách si nejprve popíšeme několik základních pojmů spjatých s detekcí tónu. V další kapitole se podíváme, jak je výška tónu vnímaná člověkem. Poté si demonstrujeme složitost její detekce na jednoduchých příkladech, kde ukážeme, proč většina intuitivních metod selže. Jelikož mnoho algoritmů detekce výšky využívá spektrální doménu, v následující kapitole si odvodíme výpočet spektra a s ním související problémy. Poté si popíšeme různé algoritmy detekce výšky. U nich budeme analyzovat jejich motivaci, způsob výpočtu a časovou složitost.

3.1 Základní pojmy

V kap. 1.2 jsme si představili percepční vlastnosti hudebního signálu, tedy vlastnosti subjektivní. Mezi ně patří výška, hlasitost a barva tónu. Objektivně tyto vlastnosti porovnáváme na základě fyzikálních vlastností, kterými se snažíme subjektivní vlastnosti aproximovat. Patří mezi ně frekvence, amplituda a spektrální struktura. Důležitý atribut těchto fyzikálních vlastností je, že jsou na sobě nezávislé (např. frekvence neovlivňuje amplitudu), zatímco jim odpovídající subjektivní vlastnosti jsou na sobě závislé (např. změna výšky tónu způsobí posuv ve vjemu hlasitosti) (Syrový, 2013, str. 72).

Sinusoida, neboli *sinusová funkce*[2] je libovolná funkce, která lze zapsat formou:

$$x(t) = A \sin \omega t + \theta,$$

kde proměnná $t \in \mathbb{R}$ je nezávislá, zatímco $A, \omega, \theta \in \mathbb{R}$ jsou pevně dané. Jednotlivé parametry odpovídají následujícím pojmům:

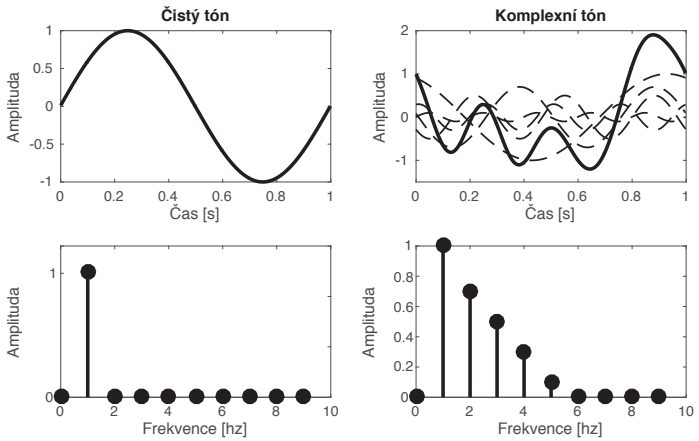
A	$=$	amplituda	$\mathbb{R}^+$
ω	$=$	úhlová frekvence	rad/s
	$=$	$2\pi f$ (frekvence)	Hz
t	$=$	čas	s
ϕ	$=$	fáze	rad
$\omega t + \phi$	$=$	okamžitá fáze	rad

Čistý tón Za čistý tón považujeme tón, jenž je složen z jediné spektrální komponenty. Lze jej tedy zapsat funkcí:

$$x(t) = A \ \sin(2\pi f_0 t), \tag{3.1}$$

kde t je čas, f_0 frekvence výšky, A amplituda a θ fáze. Frekvenci f_0 zde nazveme *základní frekvencí*. Tento tón není uchu příliš příjemný a v přírodě se téměř nevyskytuje. Většinou ho proto vytváříme uměle.

[2] Ačkoliv je rozdíl mezi pojmem sinusová funkce, který popisuje goniometrickou funkci a pojmem sinusoida, který popisuje její křivku, pro jednoduchost zde tento rozdíl zanedbáme.



Obrázek 3.1: Časový průběh a spektrum čistého a komplexního tónu

Harmonický tón Většina hudebních nástrojů vydává harmonické tóny. Ty obsahují kromě základní frekvence také tzv. *vyšší harmonické* (též známé jako *alikvótní*) komponenty, jež mohou nabývat libovolného celočíselného násobku frekvence výšky. Pro tón daný frekvencí f_0 lze komplexní tón zapsat funkcí:

$$x(t) = \sum_{n=1}^{\infty} A_n \sin\left(2\pi f_n t + \theta_n\right), \tag{3.2}$$

$$\text{kde } f_n = n \cdot f_0, \quad n \geq 1. \tag{3.3}$$

K-tému sčítanci v sumě rovnice 3.2 budeme říkat *k-tá harmonická komponenta*. Ukázku časového průběhu a spektra čistého a komplexního tónu lze vidět na obr. 3.1. U časového průběhu komplexního tónu jsou přerušovanou čarou znázorněny jednotlivé harmonické komponenty.

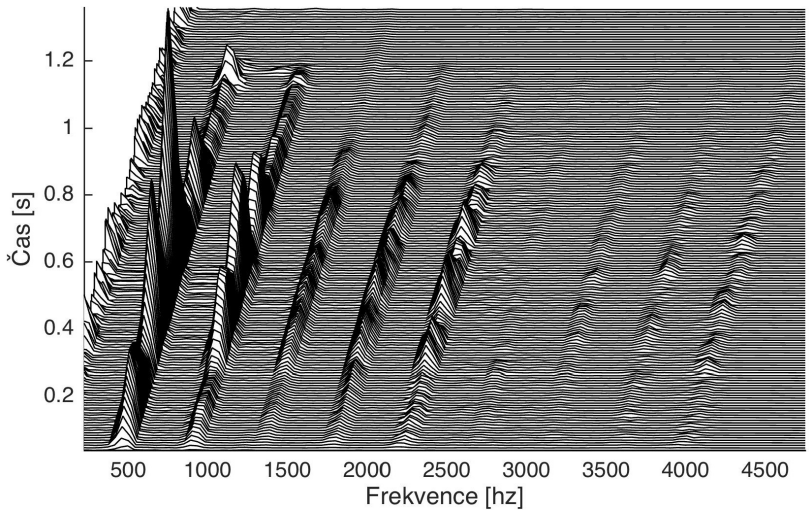
Statické a dynamické pojetí zvuku Přístup k analýze výšky z obr. 3.1 předpokládá *statické* pojetí zvuku. Tón sledujeme v tzv. fourierovském pojetí, jež spočívá na jeho naprosté stabilitě. U běžného hudebního signálu se však všechny jeho vlastnosti mění v čase a musíme pracovat s jeho *dynamickým* pojetím. To znázorňuje obr. 3.2 ve trojrozměrném prostoru, který je vymezen *dynamickou*, *melodickou* a *harmonickou* rovinou. Dynamická popisuje závislost amplitudy na čase, harmonická závislost amplitudy na frekvenci a melodická rovina popisuje závislost vnímané frekvence na čase.

Perioda signálu je nejkratší časový úsek, ve kterém se signál opakuje. Formálně nejmenší kladné T, pro které $s_x = s_{x+T}$. Pro kvaziperiodické stochastické signály tato vlastnost obvykle neplatí, proto za periodu označujeme takový nejmenší časový úsek, ve kterém je křivka signálu nejpodobnější.

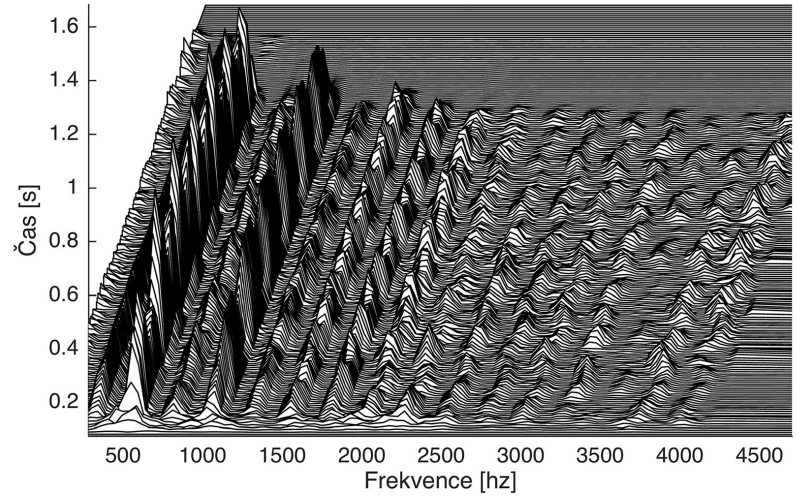
Spektrum Spektrum stacionárního úseku signálu popisuje amplitudy jednotlivých frekvencí sinusoid, na které lze signál rozložit. Ačkoliv je definováno spojitě, často se znázorňuje jeho hodnota v diskrétních frekvencích, jelikož spojité spektrum není snadné získat (viz obr. 3.1 dole), jak uvidíme v kap. 3.4.

Fundamentální frekvence je obrácená hodnota periody signálu, kterou značíme $F_0 = \frac{1}{T}$. Pro komplexní tóny je to nejmenší frekvence spektra. Pokud není perioda určena jednoznačně (nejvyšší podobnost je stejná pro více hodnot T), zvolíme za F_0 tu nejblíže odpovídající percepční výšce tónu (Klapuri, 2004, str. 3).

Harmonický vývoj noty je funkce závislosti amplitud jednotlivých harmonických komponent na čase. Tvar této funkce ovlivňuje tzv. *barvu* tónu, která nám pomáhá od sebe rozeznat různé nástroje (Syrový, 2013, str. 160). Z obr. 3.2 lze číst harmonický vývoj noty A_4 hrané na housle. Tato nota odpovídá frekvenci 440 Hz a na grafu si můžeme povšimnout vysokých hodnot v celočíselných násobcích této hodnoty. Všimněme si též, že průběh všech harmonických komponent není stejný a že výrazně vystupují pouze první čtyři vyšší harmonické. Základní frekvence je v tomto příkladě nejsilnější. Ani to však není pravidlem. Na obr. 3.3 je zobrazen časový vývoj spektra noty H_3, která odpovídá frekvenci 247 Hz. Druhá harmonická komponenta o frekvenci 494 Hz ji zde velmi výrazně převyšuje, přesto však slyšíme výšku tónu odpovídající té první. Dále si zde povšimněme fenoménu *inharmonicity*, který způsobuje nepřesné zarovnání vyšších harmonických frekvencí (zde od desáté dále). Ten je způsoben nedokonalostí hudebních nástrojů, které je lidské vnímání schopné odfiltrovat, v automatické analýze však může způsobit potíže (Klapuri, 2004, str. 22).



Obrázek 3.2: Časový průběh spektra noty A_4 hrané na housle



Obrázek 3.3: Časový průběh spektra noty H_3 hrané na housle

3.2 Psychoakustické vlastnosti výšky

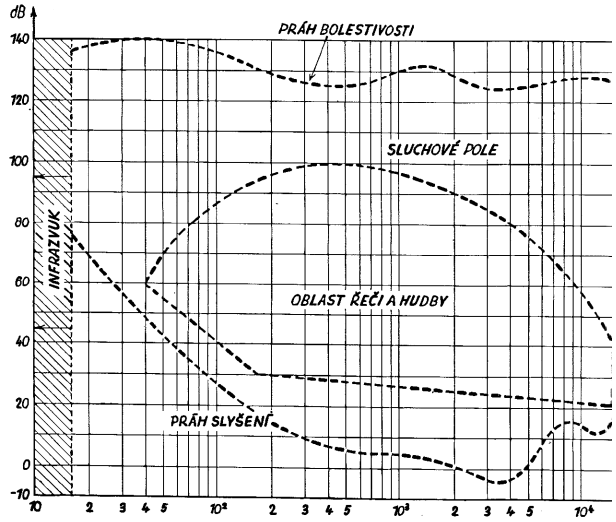
Sluchový vjem výšky zvuku odráží periodické chování jeho časového průběhu. U opravdu periodických průběhů vnímaná výška odpovídá této periodě. I neperiodické zvuky však u nás vytváří vjem výšky, který většinou odráží libovolné náznaky periodicity průběhu. Ty lze sledovat jak v časové doméně (kvaziperiodicita časového průběhu), tak ve spektrální doméně (přítomnost výrazných periodických složek) (Syrový, 2013, str. 60).

Z psychoakustického hlediska lze výšku rozdělit na absolutní a relativní. *Absolutní výška* tónu odpovídá přímé frekvenci zjištěné subjektivním porovnáváním a odpovídá naší definici v kap. 1.2. *Relativní výška tónu* je zase dána hudebním intervalem, který popisuje vnímanou vzdálenost od jiného tónu. Pokud člověk dokáže rozpoznat absolutní výšku tónu bez pomoci referenčního tónu, říkáme, že má *absolutní sluch*. Tím však disponuje málokdo, a proto vnímání výšky může být závislé na jejím kontextu. (Šmidák, 2005) *Subjektivní výška* tónu udávaná v jednotce *melů* byla vytvořena empiricky bez ohledu na frekvenci tónu. Vztahy mezi výškami v této její jednotce mají odpovídat percepčním vztahům. Konkrétně 1000 melů má čistý tón o frekvenci 1000 Hz a hladině 40 fónů (Syrový, 2013, str. 61).

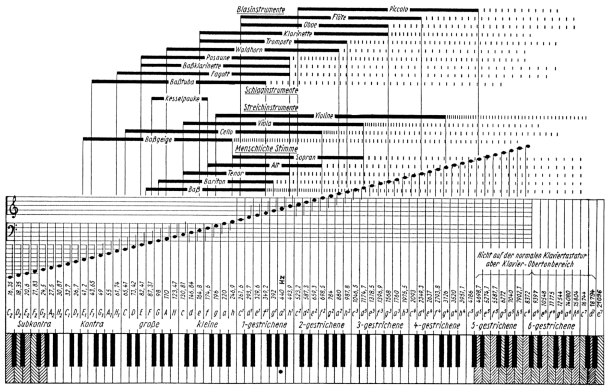
3.2.1 Vymezení rozsahu detekce výšky

Pro naše potřeby detekce výšky si potřebujeme vymezit frekvenční rozsah a frekvenční i časovou přesnost, v jaké chceme detekci provádět.

Interval slyšitelnosti se orientačně aproximuje na rozmezí 16 Hz–20 kHz. Frekvence, které je člověk schopen slyšet, jsou závislé na hlasitosti jejich zvuku a shrnuje je obr. 3.4. Zvuk vyšší frekvence nazýváme ultrazvukem, nižší pak infrazvukem. Pro naše potřeby transkripce not produkovaných běžnými hudebními nástroji se však stačí omezit na interval 40 Hz – 4 kHz, do kterého jejich tónový rozsah spadá (viz obr. 3.5). Jmenovitě jde o tónový rozsah E_1–C_8.

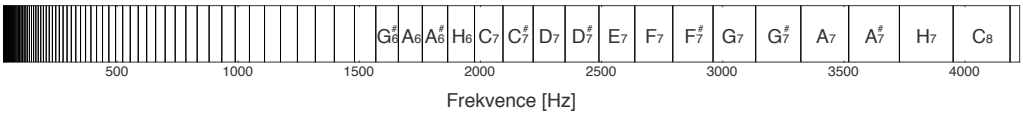


Obrázek 3.4: Rozsah slyšitelnosti (Syrový, 2013, str. 52)



Obrázek 3.5: Tónové rozsahy hudebních nástrojů (převzato z Sengpiel, 1993)

Minimální rozpoznatelná změna výšky tónu, označovaná jako *diference limen*, je závislá na jeho frekvenci i intenzitě. Nejcitlivější je v oblasti frekvence 2 kHz při hladině 60 dB, kde činí pouze 2–4 Hz. Pro vyšší frekvence či nižší hlasitosti rozpoznáváme změnu výšky hůře. (Syrový, 2013, str. 61) Pro aplikace transkripce však postačí, když detekci provedeme na přesnost jednoho půltónu, ta však může být pro výše uvedený nejmenší detekovaný tón E_1 rovna $41 \cdot (1 - \sqrt[12]{2}) = 2,4$ Hz. Pro nejvyšší detekovanou frekvenci tónu C_8 však postačí detekce s přesností $4186 \cdot (1 - \sqrt[12]{2}) = 248,9$ Hz. Rozdělení frekvenčního rozsahu na jednotlivé noty je znázorněn na obr. 3.6.



Obrázek 3.6: Frekvenční rozsah detekovaných tónů

Časová přesnost detekce výšky je sluchovým systémem omezena přibližně na 2 ms. Tato hodnota však odpovídá časovému intervalu, ve kterém si posluchač změny výšky pouze povšimne. Pro úplné „uvědomění“ konkrétní výšky tónu se udává mezní hodnota minimálně 10 ms (Syrový, 2013, str. 62).

3.2.2 Paradoxy vnímání výšky

Stále je třeba mít na paměti, že výška tónu je subjektivní vlastnost, proto je nemožné ji vždy odhadnout správně. (McLeod, 2009, s. 99) Pro obecné řešení musíme předpokládat, že pokud lze nějaký stimul použít ke generování výšky, lze jej použít v hudbě. Problémem je, že téměř libovolný signál dokáže vzbudit dojem výšky (Klapuri, 2004). I bílý šum modulovaný v amplitudě si člověk vyloží jako signál výšky dané modulační frekvence. (Syrový, 2013).

Existuje velké množství známých paradoxů, které poukazují na nečekané či nelogické vnímání výšky tónu. Např. lze vytvořit periodickou sekvenci tónů, jež bude tvořit percepční dojem neustále stoupající či klesající výšky, tzv. *Shepardův tón* (Shepard, 1964). Diana Deutch (1986) popsala experiment zvaný *Tritónový paradox*, kdy sekvenci dvou těchto tónu centrovaných v intervalu půl oktávy lze vnímat dvojznačně – jako rostoucí či klesající. Prokázala navíc, že způsob vnímání této sekvenci tónů je ovlivněn jazykem či dialektem posluchače. Iluze *Tartiniho tónu* je zase slyšena tehdy, znějí-li zároveň dva tóny, jež mají poměr frekvencí udaný v malých číslech (např. 2:3). Frekvence třetího slyšeného tónu je pak rovna rozdílu těchto dvou frekvencí. Fenomén *chybějící fundamentální frekvence* byl dlouho považován za paradox kvůli nedostatečnému chápání způsobu vnímání výšky. Fakt, že lze slyšet výšku frekvence, která není vůbec v tónu obsažená, působil dlouho nesmyslně a vysvětlují jej až dnešní percepční modely vnímání výšky (A Cheveigné, 2008).

3.2.3 Percepční modely detekce výšky

V kap. 2.2 jsme zmínili několik objevů, které vedly k současnému stavu porozumění způsobu, jak člověk vnímá výšku tónu. Mnoho lidí se snažilo jednotlivé teorie a znalosti o percepčních paradoxech sjednotit v modelech vnímání výšky. Zpočátku byly přístupy rozděleny na dvě vzájemně soupeřící teorie, tzv. modely *místa* versus modely *času*. První popisovaly vjem, který dnes nazýváme *spektrální výška*, či *lokus* a souvisí s formantovými oblastmi spektra (výraznými v amplitudě). Druhá popisovala tzv. *výšku periodicity*, která je definována na základě periodicity časového průběhu signálu a je ekvivalentem fundamentální frekvence. (A Cheveigné, 2008, str. 176).

K teorii „místa“, jež své jméno získala ze závislostí na pozici silné frekvenční komponenty ve spektru, velmi přispěl zejména Fourier, Helmholtz, Ohm a Goldstein. V této teorii se zjednodušeně předpokládalo, že lidský sluch funguje jako série Helmholtzových rezonátorů či mechanický nástroj provádějící Fourierovu transformaci. Helmholtz ve své práci popisuje způsob nervového přenášení informace o výšce z ucha do mozku. Přenos jednotlivých frekvencí popisuje pomocí technické metafory upraveného telegrafu, v němž každý drát přenáší pouze jednu zprávu. Alexandr Graham Bell tuto práci četl a nechal se metaforou rozptýlit, přičemž vynalezl telefon. Ten pak inspiruje Rutherforda k nové percepční teorii času. Percepční teorie velmi ovlivnil dlouho opomíjený fakt známý již od starověku, že struna naladěná na určitou frekvenci rezonuje i v celočíselných násobcích této frekvence. Takto slouží jako dobrý model vnímání harmonického tónu. Byl použi až v teorii *rozpoznávání harmonických vzorů* (pattern matching), která je uznávaná dodnes. (A Cheveigné, 2008, kap. 3)

Teorie „času“ zkoumá způsoby, kterými by mohl lidský mozek detekovat periodicitu časového průběhu signálu. Výrazné práce na toto téma pochází od Rutherforda, Schoutena a Wightmana. Nejprve se snažili rozpoznat výšku na základě intervalu mezi body signálu, jež jsou v periodě unikátní (např. maxima amplitudy). Zkoumala se též závislost počtu kmitů (protnutí nuly) za daný časový interval. Postupně byly vynalezeny funkce periody, které v periodě odpovídající výšce dosahují maxima. Šlo o rozdílovou funkci, funkci rozdílu čtverců a autokorelační funkci. Tzv. *autokorelační model* je dalším dnes uznávaným modelem vnímání výšky. (A Cheveigné, 2008, kap. 6)

Mezi běžně používanými algoritmy pro detekci výšky se často objevují přímé či nepřímé implementace těchto modelů. Přesto nelze považovat problém návrhu percepčního modelu za ekvivalentní s návrhem detektoru výšky. Ačkoliv řeší stejný problém, mají pro to k dispozici rozdílné prostředky.

3.3 Základní přístupy k detekci výšky

Vývoj detekce výšky je doprovázen množstvím naivních přístupů, jež by se však i dnes mohly zdát být intuitivním řešením. Pro znázornění obtížnosti problému si několik těchto základních algoritmů ukážeme a pomocí protipříkladu si prokážeme jejich nedostatečnost. Postupně budeme naivní algoritmy upravovat tak, aby splnily všechny protipříklady. Tyto protipříklady pak budou sloužit jako základní referenční model detekce výšky pro test dalších algoritmů.

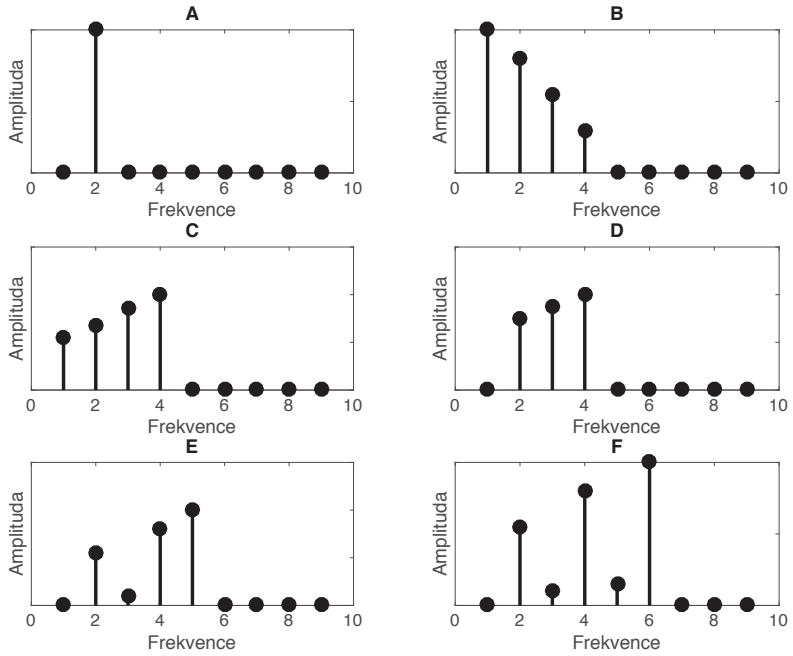
3.3.1 Spektrální metody

Již od počátků snahy detekovat výšku panoval názor, že řešení tohoto problému je nutné hledat ve spektru. Ohmův akustický zákon, který říká, že není vjem výšky bez jí odpovídající spektrální komponenty, tuto teorii podporoval. Popišme si nyní několik základních metod, které hledají výšku za pomoci frekvenční domény.

A) Pokud použijeme čistý tón jako základní model stimulu, bude ve spektru pouze jedna komponenta, jak lze vidět na obr. 3.7-A. Její pozice lze tedy použít pro určení výšky.

B) Pokud však máme harmonický tón, jako ten na obr. 3.7-B, musíme si určitou spektrální komponentu zvolit. Vybrat komponentu s nejvyšší amplitudou se zdá být rozumné a pro tento model funguje.

C) Volba nejvyšší spektrální komponenty funguje však pouze pro harmonické tóny, jejichž základní frekvence má nejvyšší amplitudu. Jak jsme viděli na reálném signálu na obr. 3.3, nemusí to být vždy pravda. Na obr. 3.7-C lze vidět krajní model této situace, kde první čtyři harmonické komponenty vzestupně rostou. Přirozenou úpravou algoritmu by bylo zvolit nejmenší frekvenci, jejíž amplituda je vyšší, než nějaký dobře zvolený práh, např. polovina maximální amplitudy spektra.



Obrázek 3.7: Modelové příklady k naivním spektrálním metodám

D) Na spektru 3.7-D je znázorněn již zmiňovaný fenomén chybějící fundamentální frekvence. Ačkoliv zde tato frekvence chybí, z výsledného signálu ji přesto slyšíme. Algoritmus C však v tomto modelu detekuje frekvenci č. 2. Náš přístup evidentně vyžaduje zásadní změnu. Nebudeme se zabývat pozicemi jednotlivých harmonických komponent, ale raději budeme porovnávat jejich vzájemné postavení. Z definice harmonického tónu (vzorec 3.2) lze vyčíst, že vzdálenost mezi libovolnými sousedními komponentami je rovna základní frekvenci. Upravený algoritmus tedy použije dvě nejmenší frekvenční složky, jejichž amplituda přesahuje stanovený práh a za výšku označí jejich vzdálenost. Pro správné vyřešení příkladu A ještě doplníme, že pokud je těchto složek méně než dvě, použijeme přístup z bodu A.

E) Případ spektra na obr. 3.7-E nám ukazuje, že předchozí algoritmus není postačující. Třetí harmonická komponenta je zde příliš nízká, což vede k detekování frekvence č. 2. Jelikož protipříkladů tohoto typu by šlo nalézt mnoho, zdá se, že je nutné pro detekci výšky zkombinovat všechny

existující harmonické komponenty. Navrhovaný algoritmus spočítá pro každou detekovanou frekvenci f hodnotu $g(f) = \sum_{i=1}^{M/f} X[i \cdot f]$, kde $X[i]$ je amplituda i-té spektrální komponenty a M je délka spektra. Zvolíme takovou výšku f, pro níž dosahuje funkce $g(f)$ maxima.

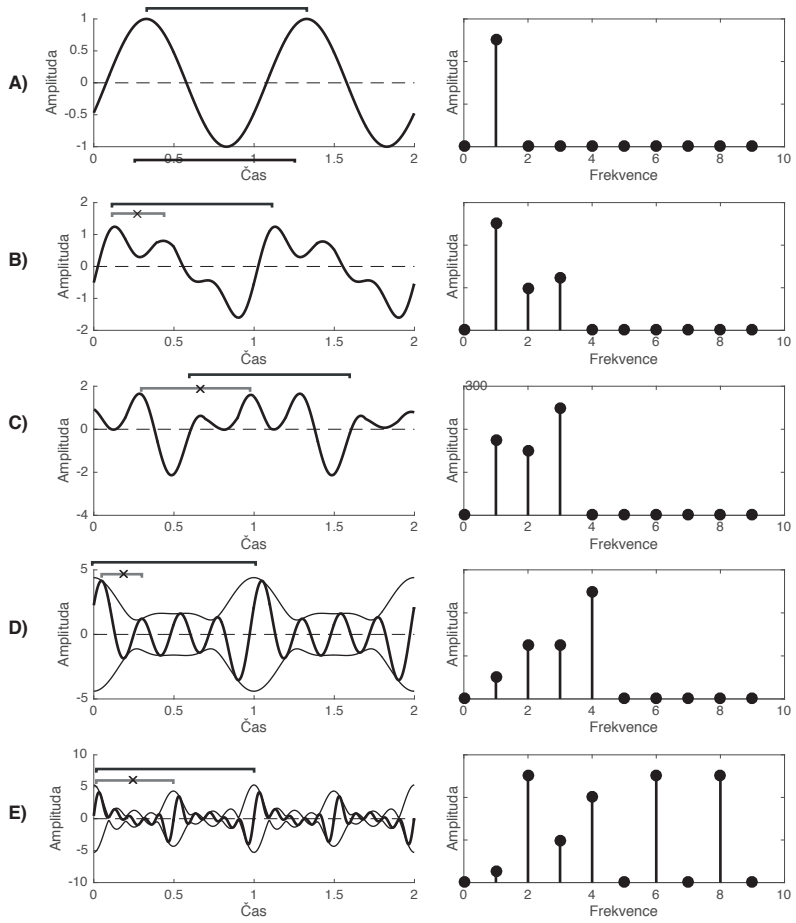
F) obr. 3.7-F ukazuje spektrum signálu, jehož výška je často dvojznačná a může odpovídat jak frekvenci č. 1, jež má všechny liché harmonické komponenty oslabené, tak frekvenci č. 2 s jistou úrovní šumu na frekvencích č. 3 a 5. Dvojznačnost tvrdí i algoritmus z bodu E, kde funkce $g(f)$ dosahuje v těchto bodech stejných hodnot. Chybám při detekci v této situaci se říká *oktávové chyby*. Bez bližších informací o frekvenci a hlasitosti signálu lze těžko odhadnout, jakou výšku posluchač uslyší. I při úplné informovanosti však nemusí být odhad stoprocentní a můžou se najít dva lidé, kteří budou výšku tohoto signálu vnímat rozdílně.

Ani poslední algoritmus však nedetekuje výšku u všech reálných signálů správně. Nejčastější chybou jsou již zmíněné oktávové chyby, jež jsme ukázali na posledních příkladech modelů. Oktávové se jim říká z důvodu, že hudební interval chyby jejich odhadu frekvence je rovný jedné oktávě, tedy došlo k rozpoznání dvojnásobné či poloviční frekvence. Tyto chyby lze u algoritmů, které se snaží modelovat lidské vnímání výšky, očekávat. I člověk totiž vnímá tyto tóny jako sobě nejpodobnější. Podrobnější analýzu dalších algoritmů lze vidět v následující kapitole. Otestování těchto algoritmů na reálných datech se pak nachází v kap. 7.

3.3.2 Metody časové domény

Výpočet spektra používaný ve výše uvedených metodách není však triviální a přináší řadu problémů, které popíšeme v kap. 3.4. Po vynálezu dostatečně přesných způsobů vizualizace časového průběhu signálu, které jsme si popsali v kap. 2.2, vzniklo na základě jeho zkoumání mnoho metod detekce výšky tónu bez provedení spektrální analýzy.

A) Z časového průběhu čistého tónu na obr. 3.8-A lze vidět, že jako periodu výšky lze jednoduše určit vzdálenost mezi libovolnými dvěma vrcholy.



Obrázek 3.8: Modelové příklady k naivním metodám časové domény

B) Pro model 3.8-B tento posup však nefunguje. Algoritmus však jednoduše upravíme tak, že budeme detekovat vzdálenost mezi dvěma vrcholy maximální amplitudy (s povolenou mírnou odchylkou).

C) Na obr. 3.8-C však vidíme signál, který dosahuje za svou periodu své maximální hodnoty dvakrát. Navrhneme nový algoritmus, který bude hledat vzdálenosti mezi dvěma body, kdy amplituda protne nulu směrem z negativní na pozitivní hodnotu.

D) Z obr. 3.8-D je však patrné, že signál může protnout nulu za svou periodu i vícekrát. Zvolíme tedy znovu úplně jiný přístup a podíváme se na tzv. obálku signálu, což je hladká křivka opisující jeho extrémy. Vzdálenost mezi dvěma nejvyššíma vrcholy obálky signálu označíme periodou výšky.

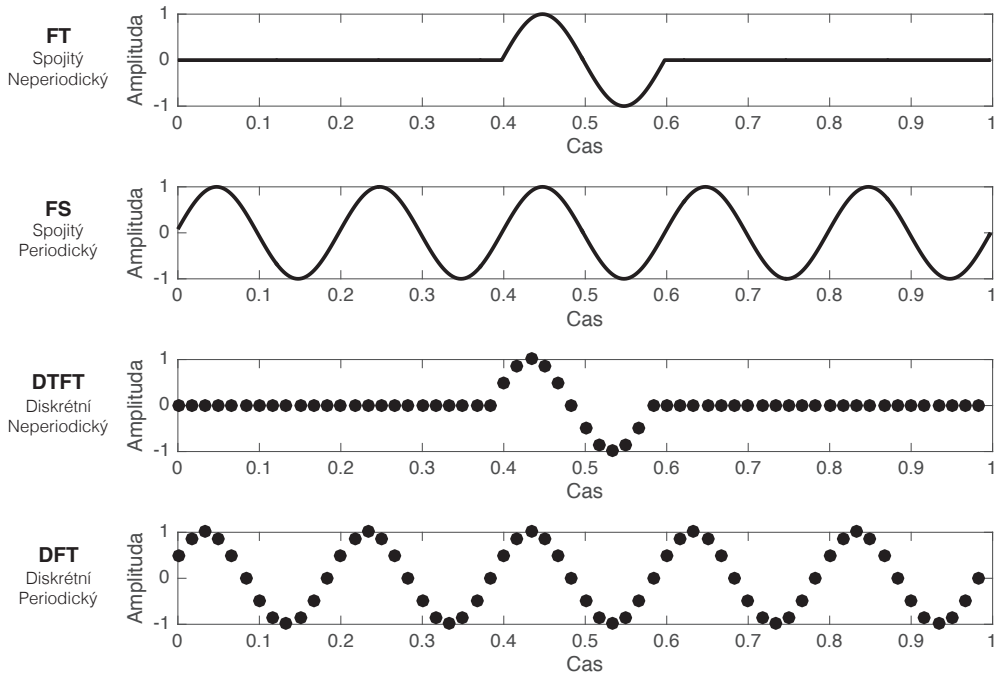
E) Na modelu 3.8-E však vidíme, že ačkoliv je tvar obálky v této periodě téměř stejný, signál se liší. Jde však o podobně dvojznačný případ, jako jsme viděli v posledním spektrálním modelu na obr. 3.7-F. Tento případ lépe rozlišují pokročilejší algoritmy detekce periodicity, které si popíšeme v kap. 3.5.

3.4 Spektrální analýza

Ve své době velmi kontroverzní tvrzení, že libovolný periodický signál lze vyjádřit součtem sinusových[3] funkcí o různé amplitudě, frekvenci a fázi, představil J. B. Fourier Francouzskému institutu v r. 1807. V komisi pro uznání článku byl i slavný matematik J. L. Lagrange, který na základě argumentu, že pomocí sinusových funkcí nejsme schopni vytvořit nespojité skoky v signálu (např. čtverec), práci odmítl. Koncept nekonečna a limitního počtu nebyl v té době na takové úrovni, aby mohl ukázat, že pomocí součtu sinusových funkcí lze reprezentovat i obdélníkový signál (pomocí nekonečného množství sinusových signálu, jež jsou ve spektru reprezentované funkcí *sinc*).

Ve spektrální analýze je naším cílem právě tento převod z časového průběhu do reprezentace frekvenčních složek. Jinými slovy zjišťujeme parametry rozkladu signálu na součet sinusových signálů. Z Fourierova přístupu, který převodem na trigonometrickou řadu řešil parciální diferenciální rovnici popisující převod tepla mezi objekty a který formalizoval ve Fourierových řadách, se vyvinulo mnoho teoretických přístupů řešení spektrální analýzy. Ty se liší zejména typem signálu, které dokáží převést. Z hlediska reprezentace rozlišujeme vstupní signály na spojité a diskrétní. Z hlediska tvaru časového průběhu pak na periodické a neperiodické. Toto rozdělení lze vidět na příkladech na obr. 3.9.

[3]Nebo též kosinusových funkcí. Je třeba si uvědomit si, že funkce kosinus lze zapsat jako funkci sinus s rozdílnou fází $\cos(\theta + \frac{\pi}{2}) = \sin(\theta)$.



Obrázek 3.9: Typy signálů dle metody pro jejich spektrální analýzu

Převod těchto čtyř různých typů signálu pak řeší následující metody. Obecná *Fourierova transformace* (Fourier transform, FT) převádí neperiodický spojitý signál, *Fourierova řada* (Fourier series, FS) pracuje s periodickým spojitým signálem, tzv. *Fourierova transformace diskrétní v čase* (Discrete time Fourier transform, DTFT) operuje na neperiodickém diskrétním signálu a *Diskrétní Fourierova transformace* (Discrete Fourier transform, DFT) převádí periodický diskrétní signál.

Je nutné si uvědomit, že všechny výše zmíněné metody pracují pouze se stacionárními signály nekonečné délky. V kapitole 1.2.3 jsme si však vysvětlili, že abychom mohli považovat náš úsek diskrétního signálu za relativně stacionární, můžeme pracovat pouze s jeho časově omezeným úsekem. Proto jej musíme na nekonečně dlouhý signál nějakým způsobem převést. Nejjednodušeji to uděláme jedním z následujících způsobů. Buď můžeme okolí našeho okna signálu označit nulovým, což nám nebrání použít DTFT. Nebo můžeme daný úsek signálu opakovat do nekonečna, čímž získáme nekonečný periodický signál a můžeme použít DFT.

Jelikož však chceme tento postup implementovat v počítači, potřebujeme v něm získané spektrum nějak reprezentovat. Abychom získali spektrum nekonečného neperiodického signálu, je přirozené, že může být složeno z nekonečně mnoho různých sinusoid, proto nelze DTFT použít. Dále se tedy budeme zaobírat pouze metodou DFT.

Jelikož metody spektrální analýzy jsou v této práci hojně využívány, v následující kapitole se pokusíme o stručné odvození metody Diskrétní Fourierovy transformace. Půjde nám především o intuitivní objasnění zásadních myšlenek, důkladnější analýzu vlastností této metody může zájemce nalézt např. v publikaci od Castanieho (2006).

3.4.1 Odvození diskrétní Fourierovy transformace

Problém: Je dán diskrétní signál x_t délky N, který chceme proložit konečným počtem sinusových funkcí s parametry amplitudy, frekvence a fáze. Jinými slovy pro posloupnost $x_t : |x_t| = N$ chceme nalézt takovou množinu $\{A_i, f_i, \theta_i\}$ tří posloupností délky M, že platí:

$$x_t = \sum_{i=0}^{M} A_i \sin\left(\omega_i t + \theta_i\right) \quad \forall t \in [0, N-1]. \tag{3.4}$$

Nejprve si definujme základní nástroj pro Fourierovu transformaci:
Definice 2. Skalární součin dvou funkcí f a g na intervalu $[a, b]$, někdy též nazývaný jako *korelaci* definujeme následovně:

$$\left\langle f, g \right\rangle_a^b = \int_a^b f(t)g(t)\,dt.$$

Pokud půjde o integraci na intervalu $[-\infty, \infty]$, budeme tyto indexy z notace skalárního součinu vynechávat.

Rozdělme si postup řešení tohoto problému do následujících kroků:

1. **Ortogonalita sinusových funkcí** Zde ukážeme stěžejní postřeh pro Fourierovu transformaci: skalární součin sinusoid rozdílných frekvencí je nulový. To nám umožní vytvořit jakýsi „filtr“, pomocí kterého budeme schopni měřit přítomnost konkrétní frekvence v signálu. Tato vlastnost nám zaručí, že přítomnost jiných frekvencí nebude mít na výsledek tohoto měření žádný vliv.

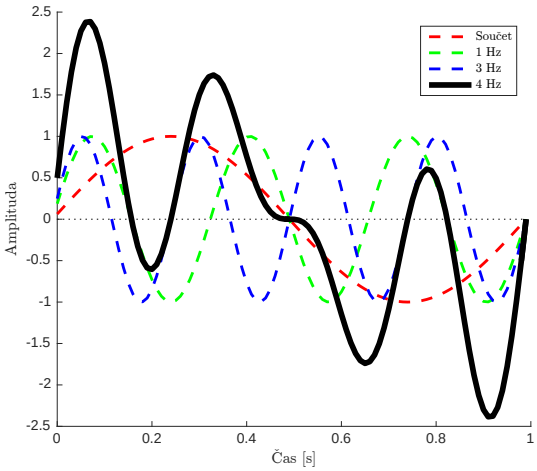
2. **Uzavřenost sinusoid stejné frekvence pro operaci sčítání** Tato vlastnost nám pomůže faktem, že součet libovolných sinusoid stejné frekvence lze reprezentovat pouze jednou sinusovou funkcí dané frekvence. Při řešení výše popsaného problému tedy stačí detekovat amplitudu právě jedné frekvence, což jej výrazně zjednodušuje.

3. **Detekce amplitudy a fáze** V tomto kroku si ukážeme, jaké hodnoty nabývá skalární součin použitý v prvním bodě při použití na dvou sinusoidách stejné frekvence. Uvidíme, že z výsledné hodnoty lze snadno detekovat amplitudu a fázi dané sinusoidy.

4. **Reprezentace v polárních souřadnicích** Zde si předvedeme způsob jak výpočet amplitudy a fáze pro danou frekvenci zjednodušit použitím komplexních čísel a odvodíme si tak běžně užívanou formu DFT.

5. **Převod na diskrétní transformaci konečného signálu** V posledním kroku si stručně ukážeme, jaké důsledky přináší fakt, že pracujeme pouze se signálem omezené délky a že je tento signál diskrétní. Popíšeme chybovost výše popsaného postupu na takových signálech a metody, jak tuto chybovost snížit.

Než se pustíme do popisování těchto vlastností, ukažme si na jednoduchém příkladu, jak Fourierova transformace funguje. Pro zjednodušení uvažujme na chvíli signál, jenž obsahuje pouze frekvence nulové fáze a amplitudy rovné jedné. Formálně $\forall i \in [1, M] : A_i = 1, \theta_i = 0$. Ukázku jedné periody takového signálu lze vidět na obr. 3.10, kde je tmavou čarou znázorněn vstupní signál, světlou pak jeho sinusové komponenty. Signál v ukázce je složen ze tří sinusoid



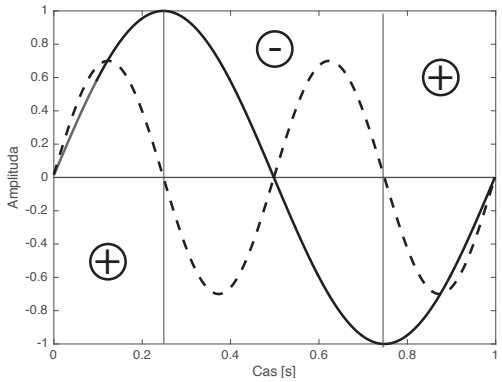
Obrázek 3.10: Ukázka jednoduchého signálu složeného ze sinusových signálu tří frekvencí stejné fáze

o frekvencích 1 Hz, 3 Hz a 4 Hz, lze jej tedy vyjádřit vztahem:

$$x(t) = \sin(2\pi \cdot t) + \sin(2\pi \cdot 3t) + \sin(2\pi \cdot 4t). \tag{3.5}$$

Pokusme se nyní přijít na způsob, jak tento předpis získat pouze z hodnot funkce $x(t)$. Z ortogonality sinusových funkcí, jež si odvodíme v následující kapitole, plyne, že skalární součin výše popsaného vstupního signálu je nulový, pouze tehdy, když se v něm daná frekvenční složka nevyskytuje. Postačí tedy takto testovat všechny možné frekvence, dokud součet všech detekovaných sinusoid nebude roven vstupnímu signálu. Pro ukázku zkusme detekovat postupně jednotlivé celočíselné frekvence:

$$\begin{aligned} \int_0^{2\pi} sin(t)x(t)\,dt &= \pi, & \int_0^{2\pi} sin(2t)x(t)\,dt &= 0, \\ \int_0^{2\pi} sin(3t)x(t)\,dt &= \pi, & \int_0^{2\pi} sin(4t)x(t)\,dt &= \pi \end{aligned} \tag{3.6}$$



Obrázek 3.11: Intuitivní pohled na ortogonalitu sinusových funkcí

Z tohoto výpočtu vidíme, že hodnoty skalárního součinu jsou nenulové pro frekvence 1 Hz, 3 Hz a 4 Hz a nulové pro frekvenci 2 Hz. Po detekování těchto tří frekvencí jsme již mohli pomocí rovnice 3.5 ověřit, že jejich součet opravdu dá původní signál. V následujících podkapitolách si ukážeme, že i za přítomnosti rozdílných amplitud a fází sinusoid lze tento postup využít.

3.4.1.1 Ortogonalita sinusových funkcí

Základní princip Fourierovy transformace spočívá v následujícím faktu: pokud spolu vynásobíme dvě sinusové funkce, jejichž frekvence jsou rozdílné, plocha pod křivkou výsledné funkce bude nulová. Tuto vlastnost lze označit jako ortogonalitu ve smyslu skalárního součinu. Podobně jako když dva ortogonální (kolmé) vektory v prostory $\mathbb{R}^n$ promítneme jeden na druhý, hodnota skalárního součinu bude nulová, pokud na sebe „promítneme“ dvě sinusoidy rozdílné frekvence, výsledek bude též nula. Intuitivně lze na tento jev ještě nahlížet tak, že část signálu, kdy jsou znaménka těchto dvou funkcí rozdílná, odpovídá části, kde jsou tyto znaménka stejná. Ukázku této intuice lze vidět na obr. 3.11, kde je znázorněna na křivkách $\sin x$ a $\sin 2x$.

Věta 1. Skalární součin dvou sinusových funkcí rozdílných frekvencí a stejné fáze je roven nule, formálně:

$$\int_{-\infty}^{\infty} \sin(m \cdot t) \sin(n \cdot t)\, dt = 0 \Leftrightarrow m \neq n$$

Důkaz 1. Z použití známé trigonometrické identity[4] plyne, že

$$\sin(m \cdot t)\sin(n \cdot t) = \frac{1}{2}\big(\cos((m-n)t) - \cos((m+n)t)\big). \tag{3.7}$$

Pokud spočteme skalární součin těchto funkcí, dostáváme:

$$\begin{aligned}\Big\langle \sin(m \cdot t), \sin(n \cdot t) \Big\rangle_{-\infty}^{\infty} &= \int_{-\infty}^{\infty} \sin(m \cdot t)\sin(n \cdot t)\, dt \\ &= \frac{1}{2}\int_{-\infty}^{\infty} \big(\cos((m-n)t) - \cos((m+n)t)\big)\, dt\end{aligned} \tag{3.8}$$

Pro rozdílné hodnoty $m \neq n$ získáváme rozdíl integrálů dvou sinusoid různých frekvencí. Jelikož však integrál sinusoidy je na nekonečném intervalu nulový, tedy platí:

$$\int_{-\infty}^{\infty} \sin(kx)\, dx = 0 \quad \forall k \ \in \mathbb{R},$$

je rovnice 3.8 pro $m, n \in \mathbb{R} : m \neq n$ též rovna nule. Pro případ $m = n$ pak lze za použití další známé identity[5] (nebo po dosazení do rovnice 3.8) a použití známého integrálu[6] tuto rovnici přepsat do formy:

$$\begin{aligned}\Big\langle \sin(kt), \sin(kt) \Big\rangle_{t_1}^{t_2} &= \int_{t_1}^{t_2} \sin^2 kt\, dt = \frac{1}{2}\int_{t_1}^{t_2} (1 - \cos(2kt)\, dt \\ &= \frac{1}{2}\Big[t + \frac{\sin(2kt)}{2k}\Big]_{t_1}^{t_2} = \frac{t_2 - t_1}{2} + c\end{aligned} \tag{3.9}$$

Pro $t_1 = -\infty, t_2 = \infty$ je rovnice 3.9 rovna ∞, tedy nenulová hodnota, což mělo být dokázáno. □

[4] Pro libovolné α, β platí: $\sin\alpha \sin\beta = \frac{1}{2}\big(\cos(\alpha - \beta) - \cos(\alpha + \beta)\big)$

[5] Pro libovolné α platí: $\sin^2(\alpha) = \frac{1}{2}(1 - \cos(2\alpha))$

[6] Pro libovolné a platí: $\int_{-\infty}^{\infty} \cos(ax + b)dx = \frac{1}{a}sin(ax + b) + c$

3.4.1.2 Uzavřenost sinusoid stejné frekvence pro operaci sčítání

Předpokládejme, že je signál daný předpisem 3.4 složený pouze z omezeného počtu frekvencí. Ačkoliv se pro každou frekvenci může vyskytnout nekonečně mnoho kombinací sinusoid mající různé amplitudy a fáze, ukažme si, že součet těchto sinusoid lze reprezentovat jedinou sinusoidou. Jinými slovy konečné množství různých frekvencí implikuje konečné množství různých sinusoid. Dokažme si tedy následující větu:

Věta 2. Libovolný součet sinusových funkcí o stejné frekvenci lze zapsat ve formě jedné sinusové funkce téže frekvence. Tedy pro libovolných N sinusoid dané frekvence ω, kde i-tá sinusoida je daná svou amplitudou A_i a fází θ_i, jsme schopni nalézt parametry jedné sinusoidy (A a θ) tak, že platí:

$$A\,\sin\left(\omega t+\theta\right)=\sum_{i=0}^{M}A_i\sin\left(\omega t+\theta_i\right). \tag{3.10}$$

Důkaz 2. Dle základního goniometrického vzorce pro součet úhlů[7] můžeme jednotlivé sinusové komponenty rozepsat následovně:

$$A_i\sin\left(\omega t+\theta_i\right)=A_i\sin\left(\omega t\right)\cos\left(\theta_i\right)+A_i\cos\left(\omega t\right)\sin\left(\theta_i\right) \tag{3.11}$$

Pokud tuto úpravu provedeme pro celou rovnici 3.10, získáváme:

$$\begin{aligned}A\,\sin\left(\omega t\right)\cos\left(\theta\right)+A\,\cos\left(\omega t\right)\sin\left(\theta\right)=&\\ =\sum_{i=1}^{N}\left(A_i\sin\left(\omega t\right)\cos\left(\theta_i\right)+A_i\cos\left(\omega t\right)\sin\left(\theta_i\right)\right)&\\ =\sum_{i=1}^{N}A_i\cos\left(\theta_i\right)\sum_{i=1}^{N}\sin\left(\omega t\right)&\\ +\sum_{i=1}^{N}A_i\sin\left(\theta_i\right)\sum_{i=1}^{N}\cos\left(\omega t\right)&\end{aligned} \tag{3.12}$$

[7] Pro libovolné α, β platí: $\sin\left(\alpha+\beta\right)=\sin\alpha\cos\beta+\cos\alpha\sin\beta$

Abychom splnili tuto rovnici, stačí, abychom našli A a ω takové, že platí:

$$\begin{aligned} A\ \cos(\theta) &= \sum_{i=1}^{N} A_i \cos(\theta_i) := x \\ A\ \sin(\theta) &= \sum_{i=1}^{N} A_i \sin(\theta_i) := y \end{aligned} \tag{3.13}$$

Získáváme tak dvě rovnice o dvou neznámých (A, θ). Pro výpočet amplitudy A obě rovnice umocníme na druhou a sečteme, získáváme tak:

$$A^2(\cos^2(\theta) + \sin^2(\theta)) = x^2 + y^2 \tag{3.14}$$

Dle základního vzorce [8] tak získáváme:

$$A\ = \sqrt{x^2 + y^2} \tag{3.15}$$

Navíc pokud dáme identity rovnice 3.13 do poměru:

$$1 = \frac{A\cos(\theta)}{x} = \frac{A\sin(\theta)}{y}, \tag{3.16}$$

zbavujeme se A a získáváme:

$$\frac{\cos(\theta)}{\sin(\theta)} = \frac{x}{y}, \tag{3.17}$$

z čehož plyne:

$$\theta = \operatorname{arccot}\left(\frac{x}{y}\right). \tag{3.18}$$

což nám umožňuje zapsat libovolný součet sinusoid stejné frekvence pomocí jedné sinusoidy téže frekvence. □

[8] Pro libovolné α platí: $\sin^2\alpha + \cos^2\alpha = 1$

3.4.1.3 Detekce amplitudy a fáze

V předchozím kroku jsme si ukázali, že pokud chceme detekovat přítomnost sinusoid v signálu, pro každou frekvenci nám postačí jediná kombinace amplitudy a fáze. Nyní si ukážeme, jak z aplikace skalárního součinu získat informaci o amplitudě a fázi.

Jelikož jsme si v prvním kroku ukázali, že skalární součin dvou sinusoid rozdílných frekvencí je nulový a tudíž se na výsledku skalárního součinu neprojeví, budeme pro jednoduchost předpokládat vstup daný pouze sinusoidou detekované frekvence.

Pro detekovanou frekvenci ω a vstupní signál obsahující sinusoidu této frekvenci o amplitudě A a fázi θ aplikujeme skalární součin následovně:

$$\begin{aligned} X = \Big\langle A\sin(\omega t+\theta), \sin(\omega t)\Big\rangle_{t_1}^{t_2} &= \int_{t_1}^{t_2} A\,\sin(\omega t+\theta)\sin(\omega t)\,dt \\ &= \frac{A}{2}\int_{t_1}^{t_2} \cos(\theta) - \cos(2\omega t+\theta)\,dt \\ &= \frac{A}{2}\cos(\theta)\Big[t\Big]_{t_1}^{t_2} - \frac{A}{2}\int_{t_1}^{t_2}\cos(2\omega t+\theta)\,dt \end{aligned} \tag{3.19}$$

Pokud však zvolíme t_1, t_2 tak, aby $t_2 - t_1$ byl násobek $2\omega t$, bude integrál kosinu roven nule a zbývá:

$$X = \frac{A}{2}\cos(\theta)(t_2 - t_1) \tag{3.20}$$

Výše zmíněným krokem jsme získali rovnici popisující vztah amplitudy a fáze. Pro určení těchto dvou neznámých však potřebujeme rovnice dvě. Další vztah můžeme však získat, pokud tento postup zopakujeme skalárním součinem se sinusoidou odlišné fáze. Pro jednoduchost zápisu použijeme fázi $-\frac{\pi}{2}$, jelikož to nám umožní provést skalární součin s funkcí kosinus. Zopakováním postupu za použití vzorce[9] získáváme:

[9]Pro libovolné α, β platí: $\sin\alpha\cos\beta = \frac{1}{2}\big(\sin(\alpha-\beta)+\sin(\alpha+\beta)\big)$

$$
\begin{aligned}
Y = \left\langle A\sin(\omega t+\theta), \sin(\omega t-\frac{\pi}{2})\right\rangle_{t_1}^{t_2} &= \int_{t_1}^{t_2} A\,\sin(\omega t+\theta)\cos(\omega t)\,dt \\
&= \frac{A}{2}\int_{t_1}^{t_2} \sin(\theta)+\sin(2\omega t+\theta)\,dt \\
&= \frac{A}{2}\sin(\theta)\Big[t\Big]_{t_1}^{t_2} + \frac{A}{2}\int_{t_1}^{t_2}\sin(2\omega t+\theta)\,dt
\end{aligned}
\tag{3.21}
$$

Pokud nyní zvolíme t_1, t_2 tak, aby $t_2 - t_1$ byl násobek $2\omega t$, první část závorky půjde k nule a získáváme:

$$
Y = \frac{A}{2}\sin(\theta)(t_2 - t_1) \tag{3.22}
$$

Tímto jsme získali soustavu dvou rovnic o dvou neznámých, která se velmi podobá rovnici 3.13:

$$
\begin{aligned}
X &= \frac{A}{2}\cos(\theta)(t_2 - t_1) \\
Y &= \frac{A}{2}\sin(\theta)(t_2 - t_1)
\end{aligned}
\tag{3.23}
$$

Proto pouze stručně odvodíme amplitudu umocněním a následném sečtením rovnic. Označme $t = (t_2 - t_1)$, pak platí:

$$
\begin{aligned}
A^2t^2(\cos^2\theta + \sin^2\theta) &= 4(X^2+Y^2) \\
A^2 &= \frac{4(X^2+Y^2)}{t^2} \\
A &= \frac{2\sqrt{X^2+Y^2}}{t}
\end{aligned}
\tag{3.24}
$$

Pro fázi pak můžeme použít postup identický s rovnicí 3.16. Protože se rovnice liší pouze v koeficientu, který se zkrátí, získáváme zde:

$$\theta = \operatorname{arccot}\left(\frac{X}{Y}\right). \tag{3.25}$$

Pokud provedeme výše uvedený postup detekce a fáze pro každou frekvenci, která se může v signálu vyskytnout, získáme tak skutečně přesný popis signálu pouze pomocí sinusoid, tedy jsme dokázali Fourierovu transformaci.

3.4.1.4 Reprezentace v polárních souřadnicích

Nyní si popíšeme, jak dva výše zmíněné kroky výpočtu skalárního součinu můžeme zjednodušit na jeden výpočet pomocí komplexních čísel. Komplexní číslo $z = x + iy$ je definované svou *reálnou* částí x, *imaginární* částí y a *imaginární jednotkou* i, pro kterou platí $i = \sqrt{-1}$. Pokud si představíme komplexní číslo z ve dvojrozměrném prostoru, kde horizontální resp. vertikální osa zobrazuje reálnou resp. imaginární část čísla, získáme tzv. *komplexní rovninu*. Číslo na komplexní rovině pak můžeme reprezentovat jeho vzdáleností od počátku $\left|z\right| = r$ a úhlem s počátkem a horizontální osou $\angle z = \theta$. Této reprezentaci říkáme *polární soustava souřadnic*. Převod mezi těmito reprezentacemi lze provést na základě následujících vztahů:

$$x = r\cos\theta, \quad y = r\sin\theta$$

Pro naši aplikaci bude praktická tzv. *exponenciální forma* zápisu komplexního čísla v polární soustavě souřadnic. Ta je daná *Eulerovým vzorcem*:

$$e^{i\theta} = \cos\theta + i\sin\theta$$

V předchozí kapitole jsme počítali nejprve skalární součin s funkcí $\sin(\omega t)$, následovně pak s funkcí $\cos(\omega t)$. Pokud provedeme korelaci signálu s funkcí $e^{i\omega t}$, z výše uvedeného Eulerova vzorce plyne, že reálná část výsledku bude reprezentovat skalární součin s funkcí kosinus, imaginární pak skalární součin

s funkcí sinus:

$$\begin{aligned} Z &= \Big\langle A\sin(\omega t+\theta), e^{i\omega t}\Big\rangle_{-\infty}^{\infty} = \Big\langle A\sin(\omega t+\theta), \cos\theta + i\sin\theta\Big\rangle_{-\infty}^{\infty} = \\ &= \int_{t_1}^{t_2} A\,\sin(\omega t+\theta)\big(\cos(\omega t) + i\sin(\omega t)\big)\,dt \\ &= \int_{t_1}^{t_2} A\,\sin(\omega t+\theta)\cos(\omega t) + i\int_{t_1}^{t_2} A\,\sin(\omega t+\theta)\sin(\omega t)\,dt \\ &= \Big\langle A\sin(\omega t+\theta), \cos(\omega t)\Big\rangle_{t_1}^{t_2} + i\Big\langle A\sin(\omega t+\theta), \sin(\omega t)\Big\rangle_{t_1}^{t_2} \end{aligned} \tag{3.26}$$

Extrakcí reálné a imaginární části výsledku získáme dílčí výsledky. Výpočet lze však výrazně zjednodušit vyjádřením výsledků přímo z komplexního výsledku:

$$A = \frac{2}{t}\big|Z\big| \quad \theta = \angle Z, \quad \text{kde } t = t_2 - t_1 \tag{3.27}$$

Na základě tohoto výsledku si již můžeme definovat běžně užívanou formu Fourierovy transformace:

Definice 3. Fourierovu transformaci signálu s_t definujeme jako zobrazení $\mathbb{R} \to \mathbb{C}$, které detekuje frekvenční složku ω v signálu s_t:

$$\mathcal{F}(s_t, \omega) = \int_{-\infty}^{\infty} s_t e^{i\omega t} dt$$

3.4.1.5 Převod na diskrétní Fourierovu transformaci

Při formulaci problému v kap. 3.4.1 jsme za vstup signálu označili diskrétní signál x_t délky N. Výše uvedený vztah však pracuje se spojitým nekonečným signálem. Popišme si, jak jej v této situaci využít a jaké jsou jeho limitace.

Podívejme se nejprve na jednodušší problém: analyzujme pouze takové signály, jež jsou složeny z frekvencí, které mají v okně celý počet period. Bez újmy na obecnosti předpokládejme, že vzorkovací frekvence je rovna N (délka okna je 1 s). Nyní tedy analyzujeme pouze ty signály, které se skládají z celočíselných frekvencí (1 Hz, 2 Hz, ...).

Z omezeného vstupu víme, že vstupní signál je složen pouze z přirozených frekvencí. Ze Shannonova teorému (viz kap. 1.2.3) víme, že nejvyšší detekovatelná frekvence je rovna $\frac{f_s}{2} = \frac{N}{2}$. Pro libovolnou frekvenci tedy platí: $\forall i \in [1, M]$: $f_i \in [1, N/2]$. Získáváme tak maximálně $N/2$ různých frekvencí. Každá sinusoida této frekvence je, jak plyne z uzavřenosti sinusoid na sčítání (kap. 3.4.1.2), jednoznačně určena svou amplitudou a fází. K jejich detekci si upravíme skalární součin z kap. 3.4.1 pro diskrétní konečné signály:

Definice 4. Skalární součin dvou konečných signálů x_t a y_t na intervalu délky N definujeme následovně:

$$\left\langle x_t, y_t \right\rangle = \sum_{t=0}^{N-1} x_t y_t.$$

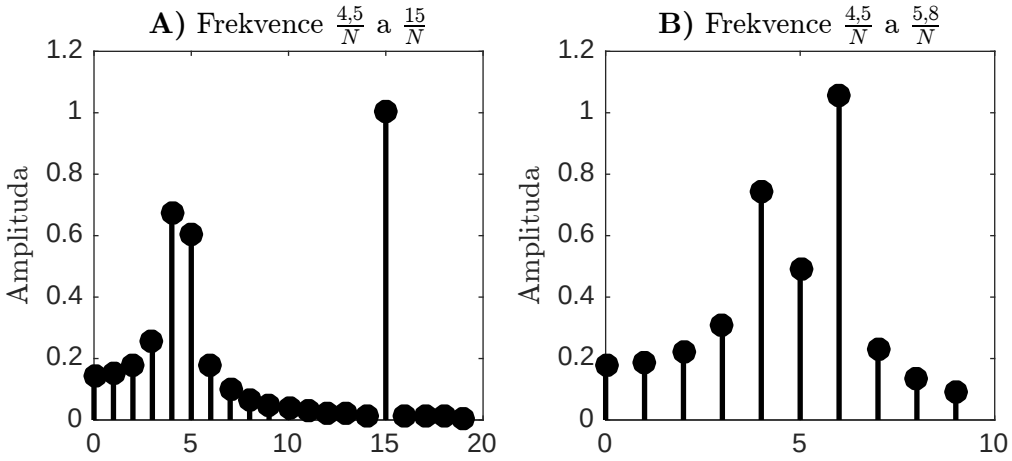
Z předchozích vztahů je těžké odvodit, že pokud má okno opravdu celý počet period, ortogonalita sinusoid je zachována i v diskrétní verzi. Intuitivně lze tento fakt vidět i geometricky z faktu, že každá sinusoida zarovnaná v počátku je na intervalu, na kterém má celý počet sinusoid, středově souměrná se středem tohoto intervalu. Podrobnější důkaz může čtenář nalézt v Sangsinově práci (2002).

Pokud tedy použijeme tento vztah a upravíme Fourierovu transformaci, získáváme:

Definice 5. Diskrétní Fourierova transformaci konečného diskrétního signálu x_t délky N, definujeme následovně:

$$\mathrm{DFT}(k) := X[k] = \sum_{t=0}^{N-1} x_t e^{i\frac{2k\pi}{N}t}, \quad 0 \leq k \ < N,$$

kde $X[k]$ je *k-tý spektrální koeficient*, který udává za pomocí komplexního čísla amplitudu a fázi sinusoidy o frekvenci $\frac{k}{N}$ přítomné v signálu x_t.



Obrázek 3.12: Ukázka prosakování

Nyní se podívejme na původní problém, ve kterém velikost okna nemusí být dělitelná periodami všech přítomných sinusoid. Pro takové případy bohužel rovnice, která by exaktně detekovala frekvenční komponenty, neexistuje (Sangsin, 2002, str. 64). Pokud bychom se takovému případu chtěli vyhnout, musela by být délka okna dělitelná nejmenším společný násobkem period všech přítomných sinusoid. Jelikož však ještě nevíme, jaké sinusoidy se v signálu vyskytují, máme zde problém „slepice a vejce“.

Ukazuje se, že ačkoliv výše uvedená metoda exaktně funguje pouze pro detekci sinusoid s celočíselným počtem period v okně, ukazuje se velmi dobrou aproximací i obecného problému. (Sangsin, 2002, str. 66). Ukazuje se zde jev *prosakování*, kde hodnota spektrální komponenty tzv. přetéká do okolních spektrálních komponent 3.12. Ačkoliv nám tento fakt komplikuje přesnou detekci hodnot jednotlivých sinusoid, ukazuje se být i užitečnou. V předchozí metodě analyzujeme pouze frekvence $1, 2, \ldots, N/2$. Pokud by fungovala tato metoda exaktně a v signálu se vyskytovala např. frekvence $\frac{9}{2}$, náš detektor by ji minul. Takto však jeho hodnota „prosákne“ do vedlejších spektrálních komponent a projeví se i v námi detekovaných 4 a 5. Tento jev je vidět na obrázku 3.12a, kde bylo analyzováno okno délky $N = 500$ a přítomné komponenty frekvence $f_1 = 4,5$ a $f_2 = 15$. Lze zde vidět, že sinusoida frekvence f_2, jež má v okně rovných 15 period, byla detekována přesně, zatímco sinusoida frekvence f_1 je rozptýlena v hodnotách okolních komponent.

Přestože se jev prosakování ukázuje praktický pro detekci frekvencí, jež nemají celočíselný počet period v okně, může nám uškodit. To v případě, že se v signálu vyskytují neceločíselné frekvenční komponenty, jež mají spektrální komponenty blízko sebe. Tento jev je znázorněn na obr. obrázku 3.12b, kde jsou v signálu přítomné frekvence $f_1 = 4,5/N$ a $f_2 = 5,8/N$. Proto je naší snahou vliv prosakování minimalizovat, k tomu však budeme muset pochopit lépe původ tohoto jevu.

Jak jsme zmínili na začátku této kapitoly, Fourierova transformace je určena pouze pro nekonečné signály. Problém prosakování spočívá v jeho převodu na konečný, který se dá vyjádřit vynásobením nekonečné funkce signálu *obdélníkovou funkcí* definovanou následovně:

$$w_{rect}(x) = \begin{cases} 1 & \textit{pro } 0 \leq x < N \\ 0 & \textit{jinak} \end{cases}$$

Okno signálu délky W, se kterým nyní pracujeme, lze pak vyjádřit z původního signálu (kap. 2.6.1) takto:

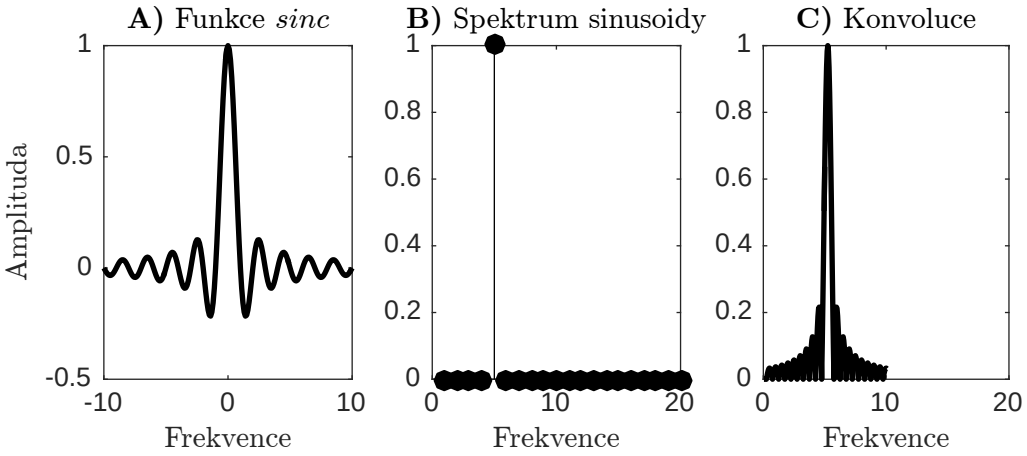
$$x_t = w_{rect}(t)s_t \quad \forall t \in \mathbb{R}.$$

Pro zjištění, jaký efekt má aplikace okna na výsledné spektrum, použijeme *konvoluční teorém*. Ten říká, že pokud násobíme dvě funkce v časové doméně, odpovídá tato operace *konvoluci* ve frekvenční doméně (McLeod, 2009, str. 28). Pro funkce f a g a jejich Fourierovy transformace F a G tedy platí:

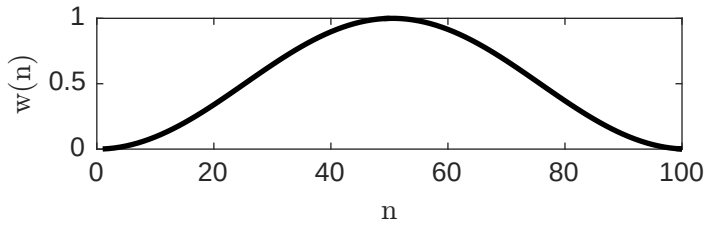
$$\mathcal{F}(f_t \cdot g_t)(k) = F(k) * G(k),$$

kde konvoluce je definovaná následovně:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,d\tau.$$



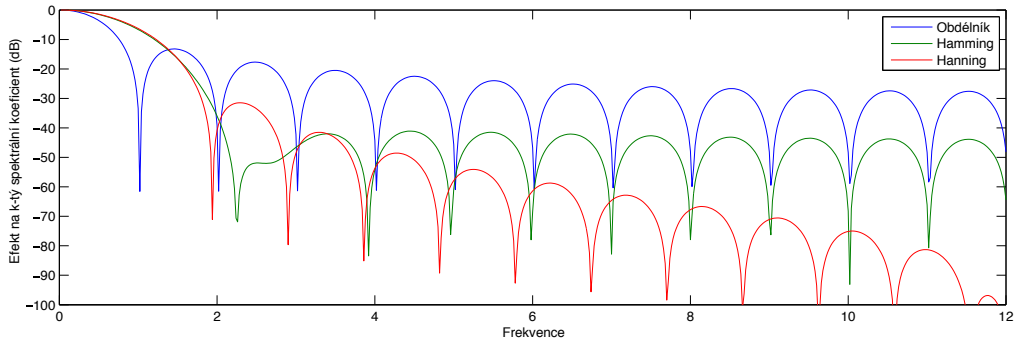
Obrázek 3.13: Ukázka konvoluce spektra s funkcí $sinc$



Obrázek 3.14: Hanningova funkce pro velikost okna $W = 100$

Ukázku efektu konvoluce spektra lze vidět na obr. 3.13. Na jeho podgrafu A lze vidět tvar funkce $sinc$, na podgrafu B spektrum sinusoidy o frekvenci $14,5$ a na podgrafu C toto spektrum konvolvované s funkcí $sinc$. Funkce $sinc$ je nulová ve všech nenulových celočíselných bodech, což potvrzuje fakt, že při konvoluci s frekvencemi, jež obsahují celočíselný počet period v okně, se prosakování neprojeví.

Pro snížení efektu prosakování bylo navrženo mnoho tzv. *okénkových funkcí*, jejichž spektrum je navrženo tak, aby konvoluce s ním působila na výsledné spektrum diskrétní Fourierovy transformaci nejméně rušivě. Tyto funkce použijeme pro vážení jednotlivých vzorků z okna signálu. Ukázalo se, že pro efekt snížení prosakování je nejlepší, pokud je hranice mezi oknem a okolím, jež můžeme pomyslně vyplnit nulami, co nejhladší. Jednu z nejpoužívanějších okénkových funkcí (Hanningovu), lze vidět na obr. 3.14.



Obrázek 3.15: Normalizovaný graf spektra několika běžně užívaných okénkových funkcí (McLeod, 2009, str. 22)

Porovnávání tohoto efektu lze zřetelně vidět na normovaném logaritmickém grafu spektra (obr. 3.15). Zde znázorněný graf vyjadřuje závislost několika funkcí $g(k)$, které lze číst z hlediska efektu konvoluce následovně: hodnota $g(k)$ říká, že amplituda a každé frekvence i přidá hodnotu $a\,g(k)$ ke frekvenční komponentě $i + k$. Indexování se provádí dle výše popsané metody Fourierovy transformace na okně délky N, tedy efekt funkce $g(k)$ se projeví pouze na celočíselných hodnotách. Např. frekvence 2 Hz přidá ke 3. spektrální komponentě hodnotu $g(1)$, ke čtvrté $g(2)$, atd. Protože je frekvence 2 Hz celočíselná, lze z grafu číst, že tyto hodnoty jsou zanedbatelné a prosakování se neprojeví. Oproti tomu pro frekvenci $1,5$ Hz bude výsledek na okolní (celočíselné) spektrální komponenty roven $g(0,5), g(1,5), g(2,5)\ldots$. To jsou hodnoty, kde dosahuje funkce g maxima, výsledný efekt bude tedy značný. Tomu odpovídá i obr. 3.12, kde je znázorněna frekvence $4,5$.

Volba délky okna diskrétní Fourierovy transformace Pro použití diskrétní Fourierovy transformace na reálné signály musíme řešit problém velikosti okna, na kterém budeme transformaci provádět. Pokud jej zvolíme příliš krátké, výsledné spektrum bude mít nízké rozlišení, pokud jej však zvolíme příliš dlouhé, nebude splněna podmínka stacionarity, viz kap. 1.2.3. Tento notoricky známý kompromis mezi časovou a frekvenční přesností při detekci výšky

lze popsat Heisenbergovým principem neurčitosti. Aby šlo všechny požadované frekvence dobře detekovat, obvykle se volí délka okna tak, aby obsahovala alespoň 4 periody každé frekvence (McLeod, 2009, str. 38). Pro nejnižší požadovanou frekvenci 40 Hz tedy potřebujeme okno dlouhé $\frac{1}{40}s$ což odpovídá alespoň $\frac{f_s}{40} = 44100/40 = 1103$ vzorkům.

Časová složitost výpočtu Fourierovy transformace Pokud bychom vypočítávali diskrétní Fourierovu transformaci dle definice 5, Pro získání $N/2$ spektrálních komponent by bylo zjevně zapotřebí provést $N^2/2$ operací komplexního násobení. Jak jsme zmínili v kap. 2.7, tato složitost je pro aplikaci pracující v reálném čase příliš vysoká. Naštěstí existuje efektivní algoritmus, který tento výpočet dokáže provést v čase $\mathcal{O}(N \log N)$. Jeho jméno zní *Rychlá Fourierova transformace* (Fast Fourier Transform, FFT) a je známý od roku 1965 (ačkoliv se vyskytují domněnky, že o něm věděl již Gauss v r. 1805). Jeho principem je metoda rozděl a panuj, kdy v každém kroku vyjádříme vztah pro výpočet diskrétní Fourierovy transformace posloupnosti délky N součtem dvou diskrétních Fourierových transformací vybraných podposloupností délky $N/2$. Tento postup aplikujeme, dokud nemáme podposloupnost délky jedna, jejíž Fourierova transformace je triviální. Aby bylo opakované využití mezivýsledku v implementaci využito maximálně, je vhodné, aby byla délka vstupní posloupnosti ve tvaru 2^m. Pak algoritmus vyžaduje pro výpočet přesně $\frac{N}{2} \log_2 N$ komplexních násobení a $N \log_2 N$ komplexních sčítání. Pro podrobnější rozbor algoritmu doporučuji práci Klejchové (2008).

Inverzní Fourierova transformace Pro potřeby následujících algoritmů si nyní definujme, jak ze spektra získat zpět původní diskrétní signál. To uděláme celkem intuitivním postupem, kde t-tý vzorek spočteme jako součet hodnot jednotlivých sinusoid v čase t. Z naší definice spektra pak získáváme následující definici:

Definice 6. Inverzní diskrétní Fourierova transformaci konečného spektra diskrétního signálu X_t délky N, definujeme následovně:

$$\text{IDFT}(t) := x_t = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{-i\frac{2k\pi}{N}t}, \quad 0 \leq t < N,$$

kde x_t je hodnota daného signálu v čase t.

Povšimněme si, že se vzorec pro výpočet IDFT výrazně podobá vzorci pro výpočet DFT, liší se pouze v znaménku u exponentu e a koeficientu před sumou. Tyto drobné rozdíly nám umožňují použít algoritmus FFT pro výpočet spektra téměř bez úprav. Časová složitost inverzní Fourierovy transformace je tedy též $\mathcal{O}(N \log N)$.

3.5 Algoritmy detekce výšky

V této kapitole si představíme vybrané algoritmy k detekci výšky tónu, jež budou kandidáty pro použití ve výsledné aplikaci. Volba těchto algoritmů byla provedena na základě rozsáhlé rešerše, kde byly zohledněny práce porovnávající jednotlivé algoritmy. Naší snahou bylo počet těchto vybraných algoritmů co nejvíce redukovat na základě existujících srovnání. Jelikož však testování pro hudební aplikace není mezi těmito algoritmy příliš běžné a mnoho prací, jež tyto algoritmy porovnávají, je zaměřeno pouze na řečové signály či je testováno pouze na omezených typech signálů, bude třeba provést testy nové.

U každého algoritmu načrtneme způsob jeho výpočtu, popíšeme si jeho vstupní parametry, určíme jeho minimální velikost okna a zmíníme jeho časovou složitost. Na základě těchto informací a výsledků jejich testování pak budeme moci zvolit vhodný algoritmus pro cílovou aplikaci.

Úkolem algoritmu je provedení detekce výšky daného okna signálu $x_t : \left|x_t\right| = N$. Výšku okna budeme pro jednoduchost udávat v jednotce počtu period za okno ($f = \text{počet cyklů}/N$). Převod na počet period za sekundu lze pomocí vztahu:

$$f_{Hz} = f\frac{f_s}{N} \text{ Hz}.$$

Při výpočtu má algoritmus navíc k dispozici spektrum, jehož výpočet pomocí rychlé Fourierovy transformace byl popsán v předchozí kapitole. Toto spektrum obsahuje $N/2$ koeficientů, kde k-tý z nich značíme X_k.

Většina algoritmů k tomuto problému přistupuje návrhem tzv. *detekční funkce*, která je funkcí periody τ. Detekční funkce je navržena tak, aby dosahovala lokálních extrémů pro taková τ, která odpovídá periodě signálu. Jelikož je signál, který je periodický v periodě T, periodický také v jeho celočíselných násobcích $kT, k \in \mathbb{N}$, a protože pracujeme se stochastickými signály, může být volba

správného lokálního minima komplikovanější. Zvláště pokud vezmeme v úvahu modely signálu popsané v kap. 3.3 na obrázcích 3.8-E a 3.7-F. Abychom tyto případy adresovali zvlášť, budeme způsob volby maxima detekčních funkcí popisovat odděleně v kap. 3.5.3. Algoritmy zde budeme dělit tradičním způsobem na algoritmy frekvenční a časové domény, tedy dle základní reprezentace hudebního signálu, se kterou pracují.

3.5.1 Algoritmy frekvenční domény

Algoritmy frekvenční domény používají k detekci výšky pozici silných frekvenčních složek ve spektru. Jejich motivace vychází z percepčních modelů místa zmiňované v kap. 3.2.3. Model *hledání harmonických vzorů* (pattern matching) patří mezi dva současné nejznámější modely vnímání výšky a vysvětluje mnohé percepční paradoxy. Proto i spektrální algoritmy využívají harmonické struktury. Jednodušší algoritmy této rodiny a signály modelující protipříklady jsme viděli v kap. 3.3.1.

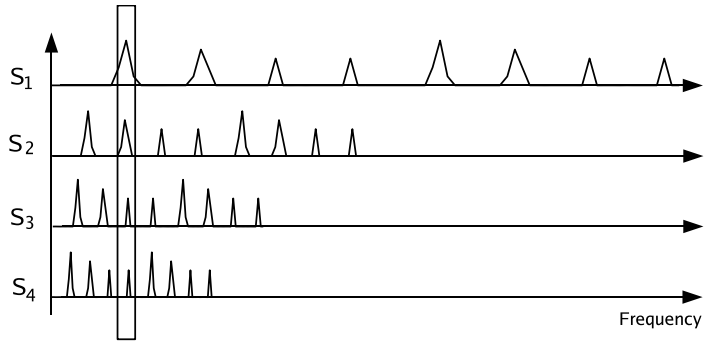
3.5.1.1 Spektrální harmonický součin

Algoritmus, známý pod jménem *Harmonic Product Spectrum* (HPS), vychází ze základní definice harmonického tónu a snaží se najít frekvenci, která tento harmonický vzor splňuje nejlépe (McLeod, 2009, str. 24). Vyhodnocení harmonického vzoru pro k-tou frekvenci je provedeno násobkem přes jeho M prvních harmonických komponent:

$$\mathrm{HPS}(k) = \prod_{n=1}^{min(M, \lfloor \frac{N}{2k} \rfloor)} X_{nk}. \tag{3.28}$$

Výsledný odhad frekvence výšky je pak dán spektrálním indexem, jenž dosahuje v této funkci maxima:

$$\widetilde{f_0} = \arg\max \mathrm{HPS}(k)$$



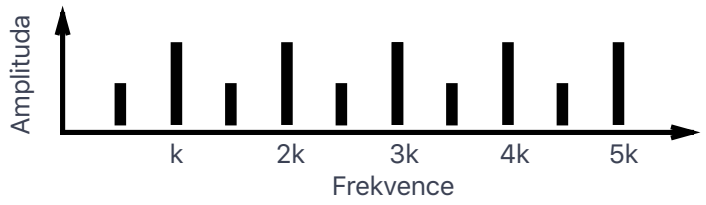
Obrázek 3.16: Znázornění algoritmu HPS (převzato z McLeod, 2009, str. 25)

Tento postup odpovídá vytvoření M převzorkovaných spekter $S_1, S_2, \ldots, S_M$, kde n-té spektrum vznikne z prvního vybranou podposloupností každého n-tého prvku. Výsledné spektrum vznikne vynásobením těchto převzorkovaných spekter po složkách. Ukázku tohoto postupu lze vidět na obr. 3.16.

K rozpoznání největší frekvence f_{max} algoritmus tedy potřebuje spektrum dlouhé alespoň Mf_{max}, které získáme z okna velikosti $2Mf_{max}$. To pro běžné hodnoty $f_{max} = 4186, M = 5$ vede na okno dlouhé přibližně 1 s. Kvůli efektu prosakování by však bylo žádoucí použít ještě většího okna pro dosáhnutí většího rozlišení. Namísto prodlužování okna, které nesmí být pro nestacionární signály příliš dlouhé, se často uměle navyšuje jeho délka metodou doplnění nul (Bahja et al., 2010). Ke zvýšení přesnosti detekce maxima se také používá interpolace křivky parabolou v oblasti lokálních maxim (Gasior et al., 2004).

Časová složitost algoritmu je rovna součtu složitosti spektra a funkce HPS: $\mathcal{O}(N \log N + \frac{MN}{2}) = \mathcal{O}(N \log N + M)$.

Ačkoliv je tento přístup vhodný pro harmonické signály, které mají prvních M harmonických komponent výrazných, v případě chybějících či oslabených komponent může snadno docházet k oktávovým chybám. Další nevýhodou je příliš velká požadovaná délka okna a nevyrovnaná přesnost, jež je daná klesajícím rozlišením u vyšších frekvencí.



Obrázek 3.17: Ukázka pozic subharmonických a harmonických komponent

3.5.1.2 Poměr subharmonických a harmonických složek

Algoritmus *Subharmonic-to-Harmonic Ratio* (SHR) používá velmi podobný vzorec, jako algoritmus HPS. Namísto operace násobení však jednotlivé harmonické komponenty sčítá. Pro řešení oktávových chyb navíc použije součet komponent, jež jsou o polovinu detekované frekvence posunuté (viz obr. 3.17). Těmto komponentám říkáme subharmonické, jelikož leží přesně na rozmezí mezi harmonickými komponentami (Sun, 2002).

Definujme sumu M harmonických amplitud frekvence k jako:

$$\mathrm{HS}(k) = \sum_{n=1}^{M} X_{kn}, \tag{3.29}$$

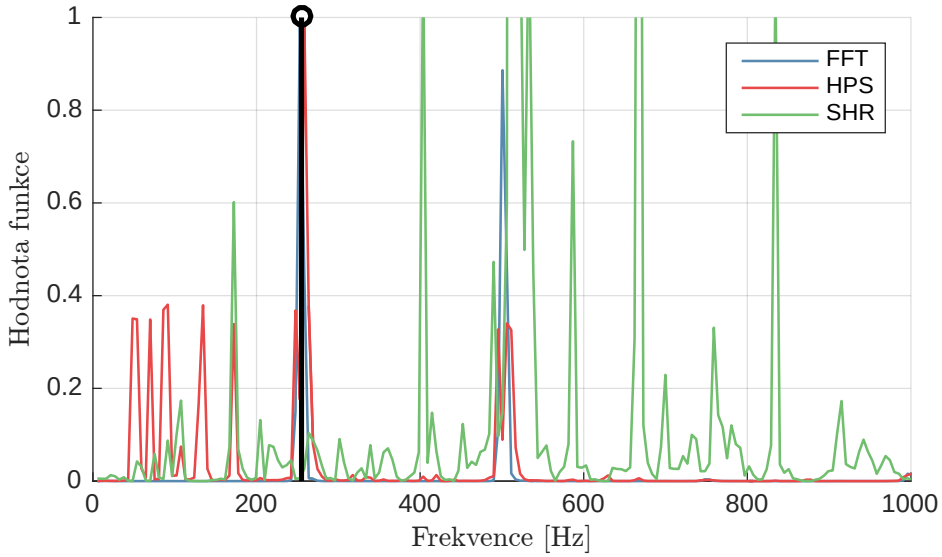
a sumu M subharmonických amplitud frekvence k jako:

$$\mathrm{SS}(k) = \sum_{n=1}^{M} X_{k(n-\frac{1}{2})}. \tag{3.30}$$

Pak je poměr subharmonických a harmonických složek (SHR) roven:

$$\mathrm{SHR}(k) = \frac{\mathrm{SS}(k)}{\mathrm{SH}(k)}. \tag{3.31}$$

Algoritmus nejprve nalezne maximum ve spektru a označí jeho frekvenci f_0 (lze však použít i jiné metody). Pokud je pro detekovanou fundamentální frekvenci f_0 hodnota $\mathrm{SHR}(f_0)$ větší než zvolený práh θ, zvolíme za výslednou výšku frekvenci $\frac{f_0}{2}$, jež leží o oktávu níže.



Obrázek 3.18: Detekční funkce algoritmů frekvenční domény pro úsek noty H_3 (246.94 Hz)

Tento algoritmus pomáhá řešit problém oktávových chyb. Požadovaná velikost okna však zůstává stejná, jako v případě algoritmu HPS a metoda doplňování nul je proto žádoucí. Časová složitost algoritmu je zde také závislá na parametru M a je rovna $\mathcal{O}(N \log N + M)$.

3.5.1.3 Kepstrální analýza

V 70. letech minulého století v Bellových laboratořích přišel Bogert, Tukey a Noll s nápadem, že spektrum ideálního harmonického tónu tvoří periodickou sekvenci s periodou základní frekvence (Noll, 1967). Pro detekci této periodicity komponent ve spektru lze použít opět Fourierovu transformaci. Proto přišli s metodou detekce výšky za pomocí spektrální analýzy spektra signálu, tzv. *kepstrální analýzy* (Cepstral analysis).[10]

[10]Jelikož autoři považovali výraz „spektrum spektra“ příliš matoucí, přišli s novými pojmy jež popisují tuto novou „kepstrální doménu“. Tyto přesmyčky se těžko překládají, proto je uvádíme pouze v originále: spectrum = cepstrum, magnitude = gamnitude, frequency = quefrency, phase = saphe, harmonic = rahmonic, period = repiod, ...

Tento postup si objasníme pomocí příkladu na obr. 3.19. Zde je znázorněna tzv. *kepstrální analýza* signálu noty G_1 $(49Hz)$ hrané na fagot. Na prvním grafu vidíme jeho spektrum vygenerované za pomocí DFT. Pro zvýraznění vrcholů vypočteme výkonové spektrum. Ačkoliv je periodicita spektra znatelná, vrcholy jsou ve spektru velmi úzké, což je dáno velkými rozdíly mezi jednotlivými komponentami. Fourierova transformace by tuto periodicitu neodhalila dosti zřetelně. Abychom dosáhli průběhu podobnějšího „sinusoidě“, zlogaritmujeme toto spektrum. Na dalším grafu je vidět, že logaritmické výkonové spektrum nám zmenšilo rozdíly mezi harmonickými komponentami, čímž dalo možnost vystoupit i komponentám s nižší amplitudou a jeho periodicita je již velmi zřetelná. Na posledním grafu vidíme DFT tohoto spektra.

Jelikož převod do spektra převádí jednotku času (s) do jednotky frekvence (s^{-1}), dvakrát provedená spektrální analýza opět pracuje s jednotkou času. Proto vidíme vysoké hodnoty kepstra v celočíselných násobcích periody výšky tónu, jež v tomto případě odpovídá $T = \frac{f_s}{f_0} = \frac{44100}{49} = 900$.

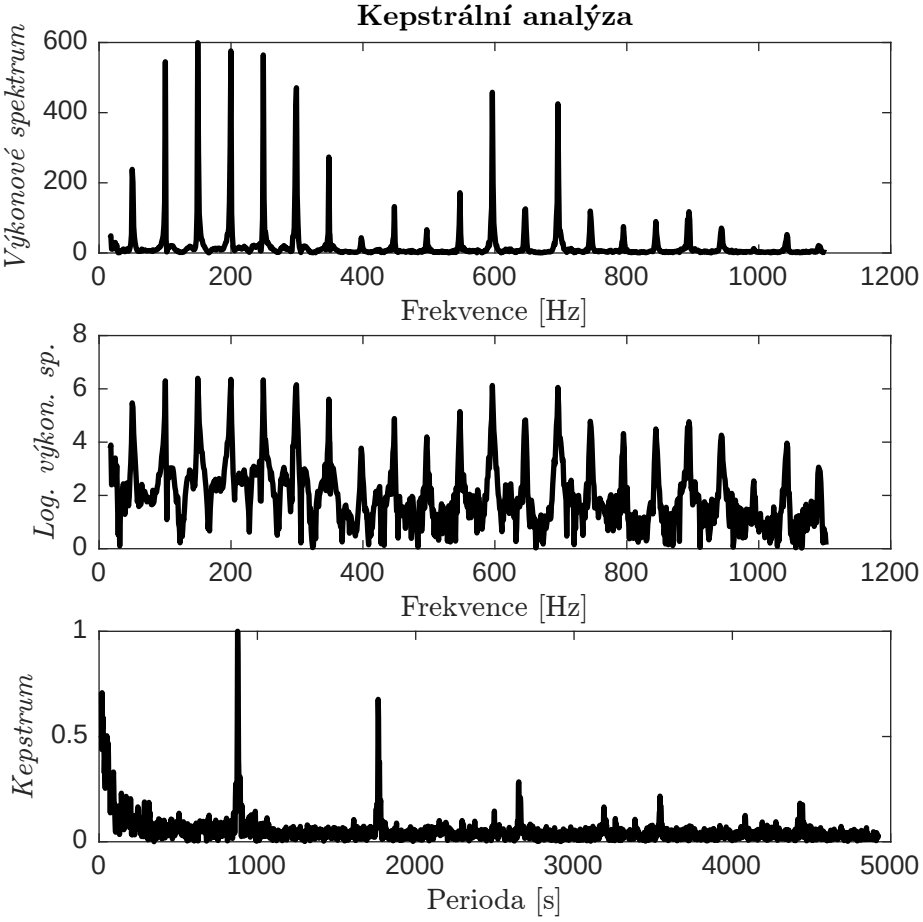
Formálně je kepstrum definováno následovně:

$$\mathrm{CEPST}(\tau) = \mathrm{DFT}\Big(\log\left(\left| \mathrm{DFT}(x_t) \right|^2 \right) \Big) \quad 0 \leq \tau < N \tag{3.32}$$

Odhad periody je pak dán vztahem:

$$\widetilde{T} = \arg\max \mathrm{CEPST}(\tau). \tag{3.33}$$

Jelikož Fourierovou transformací signálu délky N získáme $N/2$ spektrálních komponent, pro zisk N hodnot period potřebujeme okno dlouhé alespoň $4N$ vzorků. Jak jsme zmínili v kapitole 3.4.1.5, samotný převod na spektrum vyžaduje alespoň 4 periody nejmenší detekované frekvence. Pro meze navržené pro naší aplikaci to vede na okna dlouhé $16\,T_{max} = 0,4\mathrm{s} = 17640$ vzorků. Tato časová náročnost je pro použití v reálném čase příliš velká. Jelikož však kepstrum disponuje velkou odolností vůči oktávovým chybám, je za předpokladu, že je v hudebním signálu změna oktávy méně častá, možné jeho výpočtem doprovázet jiný algoritmus detekce výšky. Kepstrální analýzu tak můžeme využívat pro odhad oktávy (McLeod, 2009, str. 99).



Obrázek 3.19: Postup kepstrální analýzy noty G_1 $(49Hz)$ hrané na fagot

Výpočet kepstra se sestává ze dvou výpočtů spektra a jednoho průchodu aplikace logaritmu a druhé mocniny. Proto jej dokážeme v časové složitosti $\mathcal{O}(N \log N)$.

3.5.2 Algoritmy časové domény

Algoritmy *časové domény* využívají toho, že signály, jež vyvolávají dojem výšky, mají specifický tvar časového průběhu. Spočívají především ve hledání jeho periodicity. Většinou jsou ovlivněné percepčními modely času, mezi kterými je dnes nejuznávanější tzv. *autokorelační model* vnímání výšky. To potvrzuje i úspěšnost a oblíbenost algoritmů založené na autokorelační funkci, které dosahují nejlepších výsledků (viz kap. 7).

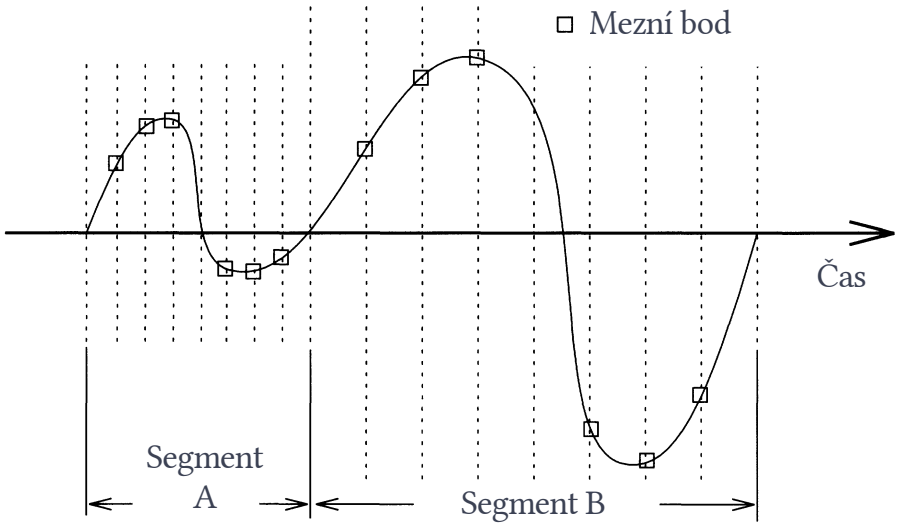
3.5.2.1 Algoritmus mezních bodů (Landmark Points Method - LPM)

Algoritmus, jež využívá postřehů zmiňovaných na základních algoritmech časové domény v kap. 3.3.2, vychází z následující intuice. Pokud se podíváme na časový průběh signálu jako na sekvenci bodů v dvojrozměrném prostoru (času a amplitudy), povšimneme si tvarově podobných vzorů opakujících se v přesném časovém intervalu. Tento interval často odpovídá periodě signálu. Pro hledání této vizuální podobnosti používáme tzv. *mezních bodů*.

Bod pozitivního překročení nuly Z si definujeme jako množinu takových indexů i, pro který platí že $x_{i-1} \leq 0$ a $x_i > 0$. Každý úsek mezi dvěma sousedními body Z_i a Z_j ($|i - j| = 1$) pak označíme *segment* $\langle i, j \rangle$ z množiny segmentů $\mathcal{S}$. Algoritmus využívá pozorování, že opakující část křivky lze identifikovat jeho nejdelším segmentem. Za předpokladu, že je tento segment v úseku periody unikátní, můžeme tímto segmentem periodu jednoznačně identifikovat. Pokud tedy najdeme další segment stejného tvaru, musí být vzdálený od našeho segmentu přesně jednu periodu signálu.

Pro porovnání tvaru křivky jednotlivých segmentů pak používáme tzv. *mezní body*. Ty jsou pro konkrétní segment mezi body $\langle i, j \rangle$ dané následovně: úsek segmentu rozdělíme na K stejně dlouhých intervalů. Pak jsou mezní body intervalu $\langle i, j \rangle$ definované jako množina bodů:

$$\mathcal{S}_{i,j} = \left\{ x_m : m = Z_i + k\frac{Z_j - Z_i}{K}, \quad k = 1, 2, \ldots, K-1, \quad k \neq \frac{K}{2} \right\}. \quad (3.34)$$



Obrázek 3.20: Ukázka mezních bodů

Na obr. 3.20 jsou znázorněny dva segmenty pro parametr $K = 8$. Mezní body jsou všechny hraniční body podintervalů segmentu kromě krajních bodů a bodu prostředního. Pro porovnání dvou segmentů pak definujeme *poměr podobnosti* následovně:

$$P(a,b) = \frac{a \cdot b}{a \cdot b + b \cdot b - a \cdot b}, \quad \text{kde } a, b \in \mathcal{S},$$

kde operace součinu dvou segmentů je definována jako:

$$a \cdot b = \sum_{i=1}^{K-2} a_i b_i.$$

Algoritmus pak provede detekci výšky aktuálního segmentu následovně:

1. Najde všechny body překročení nuly P_z.

2. Úsek mezi každými dvěma sousedními body P_z označí za segment. Pro každý zjistí hodnoty jeho mezních bodů P_m.

3. Nalezne nejdelší segment $\mathcal{S}_{max}$.

4. Spočte poměr podobnosti mezi všemi ostatními segmenty: $P(s, \mathcal{S}_{max})$: $s \in \mathcal{S}, s \neq \mathcal{S}_{max}$ a najde nejpodobnější segment $\mathcal{S}_p$.

5. Označí periodu signálu jako $T = \left|\mathcal{S}_p\mathcal{S}_{max}\right|$, kde $\left|ab\right|$ je vzdálenost mezi levými krajními body segmentů a a b.

6. Výsledná frekvence výšky je rovna $\frac{f_s}{T}$ Hz.

Jelikož segmentů nemůže být více než $\frac{N}{4}$, a nejdelší i nejpodobnější segment lze nalézt jedním průchodem množiny segmentů, je časová složitost algoritmu rovna $\mathcal{O}(N)$. Pro správnou detekci i té nejmenší frekvence potřebuje algoritmus okno alespoň dvě periody dlouhé, tedy $\frac{2}{f_{min}}$ sekund.

3.5.2.2 Rozdílová funkce

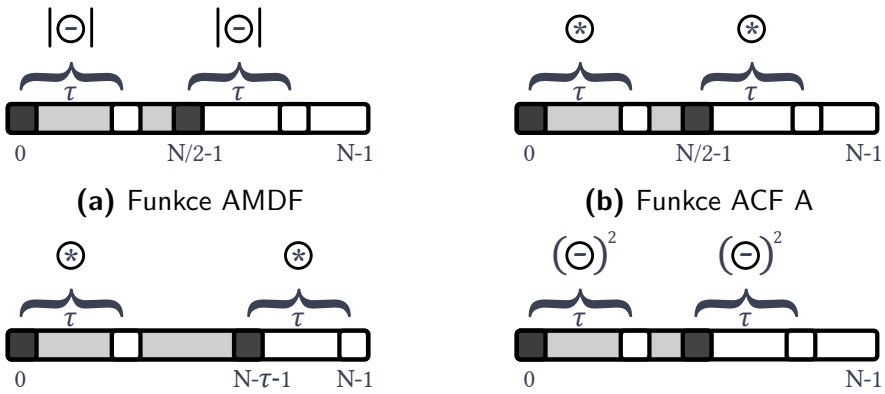
Rozdílovou funkci, též nazývanou *hřebenový filtr*, použil pro detekci výšky již Moorer (1975, str. 19) v prvním známém transkripčním systému (viz 2.2). Rozdílová funkce (Average Magnitude Difference Function, AMDF), podobně jako algoritmus mezních bodů, hledá periodu okna signálu T na základě opakujícího se podobného vzoru v jeho křivce. Tuto podobnost vypočte na základě průměrného rozdílu hodnot signálu ve vzdálenosti jedné periody. Formálně:

$$\text{AMDF}(\tau) = \frac{1}{N}\sum_{j=0}^{\frac{N}{2}-1} |x_j - x_{j+\tau}| \qquad 0 \leq \tau < \frac{N}{2} \tag{3.35}$$

Způsob výpočtu jednotlivých hodnot je znázorněn na obr. 3.21a. Pokud je signál periodický v periodě T, platí $x_t = x_{t-T}$. Z toho plyne $|x_t - x_{t-T}| = 0$, takže hodnota funkce $\text{AMDF}(T) = 0$. Pro kvaziperiodické signály nulová být nemusí, měla by se však nule blížit. Proto je odhad periody pak dán vztahem:

$$\widetilde{T} = \arg\min \text{AMDF}(\tau). \tag{3.36}$$

Jelikož se argument minima funkce na základě její lineární úpravy nezmění, můžeme při implementaci vypustit vážení sumy délkou okna $\frac{1}{N}$. I přes její jednoduchost byla tato metoda ve své době srovnatelná s úspěšností spektrálních metod i s detekcí výšky za pomocí *Kepstra* (viz dále) (J. Moorer, 1974). Hlavní



(a) Funkce AMDF **(b)** Funkce ACF A

(c) Funkce ACF B **(d)** Funkce SDF

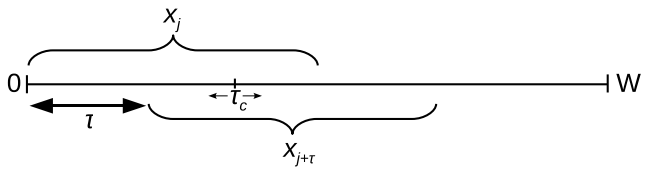
Obrázek 3.21: Znázornění výpočtu funkce AMDF, ACF a SDF

nevýhodou algoritmu AMDF je jeho výpočetní náročnost. Jelikož pro každou detekovanou periodu čteme N hodnot signálu, pro detekování N různých period je zapotřebí $\mathcal{O}(N^2)$ operací. Pro detekci maximální periody T_{max} potřebujeme hodnotu signálu v čase jejího dvojnásobku, tedy je zapotřebí okno veliké alespoň $2T_{max} = \frac{2}{f_{min}}$. Tato velikost bude společná i pro všechny metody využívající SDF.

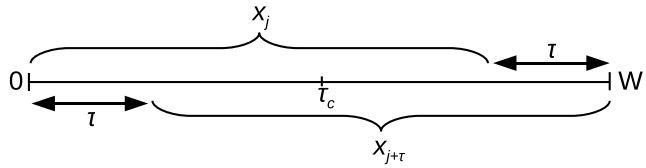
3.5.2.3 Autokorelační funkce

Autokorelační funkce (*Autocorrelation Function*, ACF) provádí korelaci signálu s verzí sebe sama posunutou o určitou hodnotu. S korelací jsme se již setkali při výpočtu Fourierovy transformace, kde bylo zjevné, že tato funkce má velmi dobré schopnosti při měření vzájemné podobnosti dvou periodických signálů. Metoda ACF spočívá ve výpočtu hodnoty korelace signálu v závislosti na „pozdržení“ korelovaného signálu o časový posun τ. Pro hodnotu periody T by tato funkce měla dosahovat maxima, jelikož naprosto periodické funkce se sebou korelují nejlépe při jejich posunu o celočíselný násobek periody. Definujme si tedy *autokorelační funkci* typu A následovně:

$$\mathrm{ACF_A}(\tau) = \sum_{j=0}^{\frac{N}{2}-1} x_j x_{j+\tau} \qquad 0 \leq \tau < N/2 \tag{3.37}$$



(a) Efektivní centrum algoritmu $\mathrm{ACF_A}$



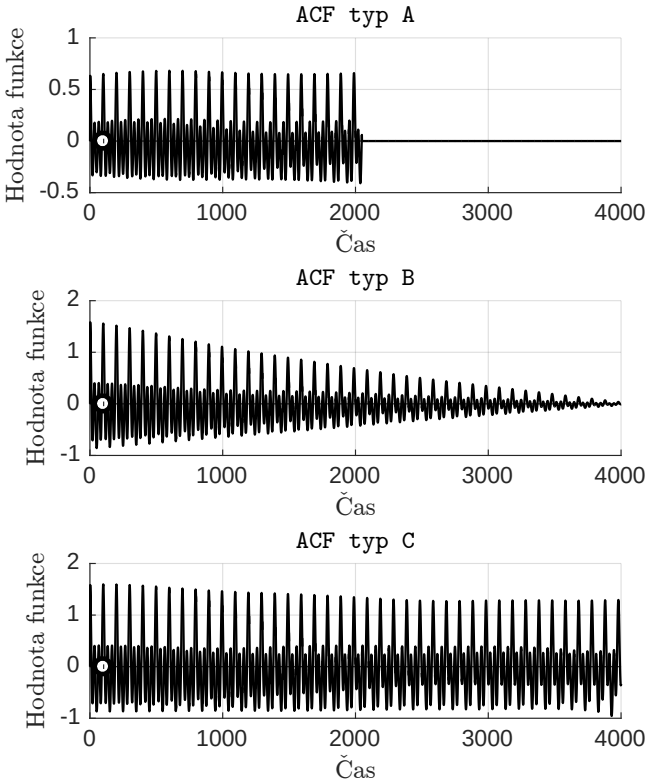
(b) Efektivní centrum algoritmu $\mathrm{ACF_B}$

Obrázek 3.22: Efektivní centrum různých verzí algoritmu ACF (Převzato z McLeod, 2009, str. 14)

Na obr. 3.21c je znázornění výpočtu autokorelační funkce. Ze vzorce 3.37 lze vidět, že pro výpočet hodnoty $\mathrm{ACF}_A(\tau)$ algoritmus využívá hodnoty signálů v bodech $[0, \frac{N}{2} + \tau]$. Velikost okna signálu braný v úvahu pro zjištění úrovně korelace pro periodu τ je tedy závislý na τ. Pro naprosto periodické signály toto nemusí být problém, u reálných signálů se však může na konci počítaného okna vyskytnout určitá „aperiodicita“, která ovlivní výpočet pouze pro vyšší hodnoty τ. Definujme si proto upravenou autokorelační funkci typu B:

$$\mathrm{ACF_B}(\tau) = \sum_{j=0}^{N-1-\tau} x_j x_{j+\tau} \qquad 0 \leq \tau < N \tag{3.38}$$

Změna spočívá v úpravě definičního oboru a především v tom, že korelace je prováděna pro libovolné τ v maximálním rozsahu okna. Výhodou tohoto přístupu je, že pro získání hodnoty $\mathrm{ACF_B}(\tau)$ jsme zohlednili okno celého signálu. To lze vidět na obr. 3.22, kde je znázorněno tzv. *efektivní centrum* obou funkcí. To definujeme jako průměr pozic vzorků, ze kterých algoritmus čte při výpočtu hodnoty $\mathrm{ACF}(\tau)$. Z obrázku můžeme vyčíst, že efektivní centrum typu A je závislé na hodnotě τ, zatímco pro typ B je vždy ve středu signálu.



Obrázek 3.23: Hodnoty autokorelační funkce na okně signálu noty A_4

Na obrázku 3.23 jsou hodnoty obou verzí ACF vypočítaných pro jedno okno signálu tónu A_4 hrané na flétnu. Bílou tečkou je zde znázorněna hodnota periody ($T = 100$). Můžeme zde sledovat další vlastnost funkce ACF typu B a tou je tzv. *zužující efekt*. Počet členů v sumě 3.38 totiž klesá v závislosti na rostoucí periodě τ. Abychom se zbavili tohoto efektu, definujme si tzv. *nevychýlenou autokorelační funkci*, jež nazveme autokorelační funkcí typu C. Ta vyvažuje zužující efekt vážícím koeficientem, jenž přímo úměrně roste v závislosti s rostoucím τ:

$$\mathrm{ACF_C}(\tau) = \frac{N}{N-\tau} \sum_{j=0}^{N-1-\tau} x_j x_{j+\tau} \qquad 0 \leq \tau < N \tag{3.39}$$

Výsledný odhad periody výšky pomocí autokorelační funkce typu $t \in \{A, B, C\}$ je pak dán vzorcem:

$$\widetilde{T} = \arg\max \mathrm{ACF}_t(\tau)$$

Každý z těchto typů autokorelace má své výhody. Typ A využívá pro každé τ konstantního počtu vzorků ze signálu. Typ B zase má konstantní efektivní centrum napříč různými hodnotami τ (Rinaldi et al., 2015). Typ C oproti metodě B má vyvažovanou závislost počtu členů na τ. Přínos této úpravy je však diskutabilní (Alain de Cheveigné, 2002). Pokud je totiž nějaký signál periodický v periodě T, je periodický i v periodě $kT, k \in \mathbb{N}$. U kvaziperiodických signálů se může stát, že pro takový signál je hodnota korelace v násobku periody vyšší, než pro hodnotu periody ($\mathrm{ACF_C}(kT) > \mathrm{ACF_C}(\tau), k \in \mathbb{N}$). Proto může být znevýhodnění vyšších hodnot autokorelační funkce (zužující efekt), pro detekci výšky žádoucí. Cheveigne (2002) však argumentuje, že využívání zužujícího efektu k vážení není vhodné, jelikož je závislé na velikosti okna, a že by toto vážení mělo být navržené explicitně vzhledem ke konkrétním hodnotám τ (viz algoritmus YIN).

Minimální velikost okna pro autokorelační funkci by měl být alespoň dvojnásobek nejdelší periody, tedy $2T_{max} = \frac{2}{f_{min}}$. Delší okna poskytují tu výhodu, že zašuměné či chybné vzorky nenaruší hodnotu korelace v takové míře, pro nestacionární signály však nejsou vhodné. Tato velikost okna bude totožná i pro následující metody založené na autokorelaci.

Časová složitost algoritmu přímého vyhodnocení všech typů autokorelace je rovna $\mathcal{O}(N^2)$. Existuje však postup, jak využít algoritmus rychlé Fourierovy transformace ke zrychlení tohoto výpočtu. K tomu nám pomůže *Wiener-Khintchinův teorém* (Weisstein, 2006). Dle tohoto teorému lze autokorelační funkci vyjádřit pomocí inverzní Fourierovy transformace výkonového spektra signálu (Klapuri, 2006, str. 24). *Výkonové spektrum* získáme ze spektra umocněním jednotlivých spektrálních komponent. Při použití Wiener-Khintchinova teorému tedy získáváme autokorelaci typu B následovně:

$$\mathrm{ACF_B}(\tau) = \mathrm{IDFT}\big(\big|\mathrm{DFT}(x_t)\big|^2\big), \tag{3.40}$$

kde K je zvolená délka transformace a ovlivňuje počet výstupních hodnot autokorelace. Jelikož nás zajímá pouze reálná část signálu, lze tuto rovnici zapsat:

$$\mathrm{ACF_B}(\tau) = \frac{1}{K}\sum_{k=0}^{K-1}\cos\Big(\frac{2\pi\tau k}{K}\big|X[k]\big|^2\Big). \tag{3.41}$$

Jak popisuje McLeod (2005, str. 26), abychom pomocí této metody získali $N/2$ hodnot autokorelační funkce a vyhnuli se efektu prosakování, je vhodné okno signálu doplnit $N/2$ nulami na celkovou délku $\frac{3}{2}N$. Pokud k výpočtu DFT a IDFT požijeme algoritmus FFT, bude časová složitost výpočtu $N/2$ hodnot $\mathrm{ACF_B}$ rovna $\mathcal{O}(N \log N)$. Jelikož se $\mathrm{ACF_C}$ se od typu B liší pouze převážením hodnot, jež lze provést v lineárním čase, jeho složitost je též $\mathcal{O}(N \log N)$. Pro rovnici $\mathrm{ACF_A}$ však tento vzorec nefunguje a proto je jeho složitost $\mathcal{O}(N^2)$

3.5.2.4 Kvadratická rozdílová funkce

Kvadratická rozdílová funkce (*Square Difference Function*, SDF) se dá označit za kombinaci AMDF a ACF a vychází též z tvarové podobnosti periodického signálu ve vzdálenosti jeho periody. Měření této podobnosti se zde provádí podobně jako v AMDF, pouze místo absolutní hodnoty je započtena druhá mocnina rozdílu:

$$\mathrm{SDF_A}(\tau) = \sum_{j=0}^{\frac{N}{2}-1} (x_j - x_{j+\tau})^2 \qquad 0 \leq \tau < \frac{N}{2}. \tag{3.42}$$

Podobně jako u autokorelační funkce navrhneme kromě verze A pro $N/2$ hodnot τ také variantu B, pro N jež dodržuje konstantní efektivní centrum napříč hodnotami τ:

$$\mathrm{SDF_B}(\tau) = \sum_{j=0}^{N-1-\tau} (x_j - x_{j+\tau})^2 \qquad 0 \leq \tau < N. \tag{3.43}$$

Některé zdroje (McLeod, 2009, str. 18) uvádějí též třetí *nevychýlenou verzi* SDF, kterou podobně jako nevychýlenou autokorelační funkci získáme pomocí vážení hodnot $\mathrm{SDF_B}$:

$$\mathrm{SDF_C}(\tau) = \frac{N}{N-\tau}\mathrm{SDF_B}(\tau) \qquad 0 \leq \tau < N. \tag{3.44}$$

Časová složitost výpočtu SDF dle vzorce je přirozeně také $\mathcal{O}(N^2)$. Tento výpočet lze však dle McLeoda (2005, str. 42) převést na výpočet autokorelační funkce, kterou dokážeme vyhodnotit v čase $\mathcal{O}(N \log N)$. Ukažme si to např. na SDF typu B, kde začneme rozložením vzorce:

$$\begin{aligned} \mathrm{SDF}_\mathrm{B}(\tau) &= \sum_{j=0}^{N-1-\tau} (x_j - x_{j+\tau})^2 \\ &= \sum_{j=0}^{N-1-\tau} (x_j^2 - x_{j+\tau}^2) - 2 \sum_{j=0}^{N-1-\tau} x_j x_{j+\tau} \end{aligned} \tag{3.45}$$

Definujme $m(\tau) = \sum_{j=0}^{N-1-\tau} (x_j - x_{j+\tau})^2$. Pak platí:

$$\mathrm{SDF}_\mathrm{B}(\tau) = m(\tau) - 2\,\mathrm{ACF}_\mathrm{B}(\tau),$$

kde $m(\tau)$ můžeme iterativně vypočítat jedním průchodem se zvyšujícím se τ. Jelikož $m(0) = 2 \sum_{j=0}^{N-1-\tau} x_j^2$ a lze snadno odvodit, že v každém dalším kroku z této sumy odečteme druhou mocninu ze začátku intervalu a druhou mocninu z konce intervalu, tedy pro $\tau > 0 : m(\tau) = m(\tau - 1) - x_{\tau-1}^2 - x_{N-\tau}^2$. Časová složitost výpočtu SDF je tedy součet složitosti výpočtu autokorelace a výpočtu funkce $m(\tau)$: $\mathcal{O}(N \log N + N) = \mathcal{O}(N \log N)$.

3.5.2.5 Normalizovaná korelační funkce

Talkin (1995) ve své práci popisuje metodu hledání výšky tónu pomocí upravené korelační funkce (Normalized cross-correlation function, NCCF). Ta byla využita v jeho systému RAPT (Robust algorithm for Pitch tracking) a také v sytému YAAPT (Yet Antoher Algorithm for Pitch Tracking, Kasi, Kavita et al., 2002). Tyto systémy jsou příliš výpočetně náročné pro využití v aplikaci v reálném čase. Jelikož se však funkce NCCF ukázala v těchto systémech užitečnou, podíváme se na její vlastnosti.

Její úprava spočívá v přidání normalizačního dělitele odvozeného z energie části signálu využívaného pro korelaci:

$$\mathrm{NCCF}(\tau) = \sum_{j=0}^{N-1-\tau} \frac{x_j x_{j+\tau}}{\sqrt{e_0 e_\tau}}, \qquad e_\tau = \sum_{j=\tau}^{N-1} x_j^2, \qquad 0 \leq \tau < N, \tag{3.46}$$

$$\widetilde{T} = \arg\max \mathrm{NCCF}(\tau). \tag{3.47}$$

Tato normalizace zaručí, že všechny hodnoty této funkce jsou z intervalu $[-1, 1]$. V hodnotách odpovídajícím periodě signálu se pak její hodnoty blíží jedné. Tato normalizace se ukáže praktickou při detekování lokálního maxima, jež popisujeme v kap. 3.5.3. Také zvyšuje odolnost algoritmu vůči modulacím amplitudy (Talkin, 1995) a měla by v posunu o periodu produkovat maxima, která jsou „prominentnější a tím snadněji detekovatelní“ (Kasi, Kavita et al., 2002).

Časová složitost je za využití FFT pro výpočet autokorelace rovna $\mathcal{O}(N \log N)$, jelikož hodnoty e_τ lze pro celý výpočet zjistit lineárně. To provedeme vyhodnocením e_0 a iterativním upravováním této hodnoty pro rostoucí τ, kdy v každém kroku spočteme $e_\tau = e_{\tau-1} - x_{\tau-1}^2$.

3.5.2.6 Rozdílová funkce normalizovaná průměrem

Cheveigné (2002) ve své práci o algoritmu *YIN* popisuje speciální normalizaci kvadratické rozdílové funkce, pomocí které se snaží vytvořit rovnováhu mezi výhodami rozdílové a autokorelační funkce (název vychází z konceptu jin–jang z orientální filosofie). Algoritmus YIN kromě této funkce spočívá ve speciálním přístupu k detekci maxima (popsáno v kap. 3.5.3) a zvyšování přesnosti interpolací parabolou (kap. 3.5.4). Rozdílovou funkci normalizovanou průměrem (Cumulative mean normalized difference function, CMNDF) definujeme následovně:

$$\text{CMNDF}(\tau) = \begin{cases} \dfrac{\text{SDF}_\text{B}(\tau)}{\frac{1}{\tau}\sum_{j=0}^{\tau}\text{SDF}_\text{B}(j)} & 1 \leq \tau < \frac{N}{2}. \\ 1 & \tau = 0 \end{cases} \tag{3.48}$$

$$\widetilde{T} = \arg\min \text{CMNDF}(\tau) \tag{3.49}$$

Oproti kvadratické rozdílové funkci tato funkce usnadňuje detekci lokálních minim, jelikož její hodnota klesne pod hodnotu jedné pouze tehdy, když je hodnota funkce $\text{SDF}(\tau)$ nižší než průměrná. Jelikož lze normalizační členy získat z vypočtené funkce $\text{SDF}_\text{A}(\tau)$ jedním průchodem, výpočetní složitost odpovídá složitosti výpočtu SDF_A, tedy je rovna $\mathcal{O}(N \log N)$.

3.5.2.7 Vážená autokorelační funkce

Shimamura a Kobayashi (2001) ve své práci zmiňují, že autokorelační a kvadratická rozdílová funkce má statisticky odlišné chování v šumném prostředí. Představují způsob, jak pomocí jejich kombinace docílit detekční funkce s větší odolností proti šumu. Vážená autokorelační funkce (Weighted Autocorrelation function, WACF) je definovaná následovně:

$$\text{WACF}(\tau) = \sum_{j=0}^{N-1-\tau} \frac{\text{ACF}_\text{B}}{\text{SDF}_\text{B} + k}, \qquad 0 \leq \tau < N/2, \tag{3.50}$$

Jelikož ACF dosahuje pro hodnoty $\tau = T$ maxima a SDF minima, můžeme určit odhad T též pomocí maxima:

$$\widetilde{T} = \arg\max \text{NCCF}(\tau). \tag{3.51}$$

Parametr k je zde především z důvodu zamezení divergence inverze SDF, a běžně se volí $k = 1$. Jelikož dokážeme výsledky ACF a SDF snadno zkombinovat v lineárním čase, je složitost výpočtu WACF rovná složitosti výpočtu ACF a SDF, tedy $\mathcal{O}(N \log N)$.

3.5.2.8 Speciálně normalizovaná autokorelační funkce

O další způsob, jak zkombinovat vlastnosti ACF a SDF, se pokusil McLeod (2009). Popisuje, že SDF má schopnost vypočítat vhodně odhad výšky, i když nemá celočíselný počet period v okně. ACF má zase výhodu, že její hodnoty jsou větší než nula pouze ve vyšších periodicitách, jež umožňuje snazší výběr maxima. Speciálně normalizovaná autokorelační funkce (Specially-normalised autocorrelation function, SNAC) je definovaná následovně:

$$\mathrm{SNAC}(\tau) = \frac{2\sum_{j=0}^{W-1-\tau} x_j x_{j+\tau}}{\sum_{j=0}^{W-1-\tau} (x_j^2 + x_{j+\tau}^2)}, \qquad 0 \leq \tau < N/2, \tag{3.52}$$

Za použití funkce $m(\tau)$ definované v kap. 3.5.2.4 lze tento výraz zapsat:

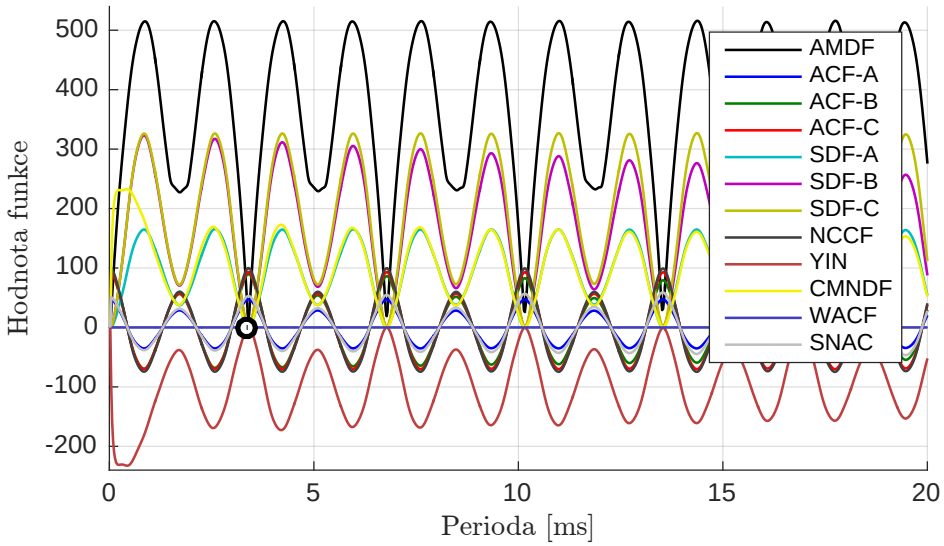
$$\mathrm{SNAC}(\tau) = \frac{2\mathrm{ACF_B}(\tau)}{m(\tau)} \tag{3.53}$$

Tato funkce zajišťuje normalizační vlastnost $-1 \leq \mathrm{SNAC}(\tau) \leq 1$, jež bude praktická při volbě lokálního maxima. Navíc když je autokorelace signálu v hodnotě τ nulová, platí i $\mathrm{SNAC}(\tau) = 0$. Naopak při perfektní korelaci, kdy se obě funkce shodují, dosahuje $\mathrm{SNAC}(\tau) = 1$.

Časová složitost výpočtu funkce SNAC se rovná složitosti výpočtu ACF a $m(\tau)$, tedy $\mathcal{O}(N \log N)$.

3.5.3 Volba vrcholu detekčních funkcí

Většina algoritmů zmíněných v předchozích dvou kapitolách využívali detekční funkce, jejichž maximální či minimální hodnota odpovídala odhadu periody či frekvence výšky tónu. V určitých případech však nemusí být nejlepší zvolit globální maximum. Jeden z důvodů je, že u mnohých funkcích se nalézá v nule (ACF, CEPST). Pro funkce detekující periodu jsou maxima na celočíselných



Obrázek 3.24: Detekční funkce algoritmů časové domény pro úsek noty H_3 ($T =$ 3,356 ms)

násobcích základní periody nutností. Hodnoty ve vyšších periodách pak můžou základní periodu přerůst ať už z důvodu výskytu šumu, kvantizační chyby, neceločíselného počtu period v okně či z důvodu numerických nepřesností při vyhodnocování dané funkce. V případech zmiňovaných v kap. 3.3 (poslední modely na obou obrázcích) může být detekce správného vrcholu obtížným problémem. McLeod (2009, str. 16) popisuje problém volby vrcholu (Peak picking) jako „černou magii“ napříč řešeními problému detekce výšky tónů, pravděpodobně z důvodu jejich heuristické povahy. Ukázalo se však, že návrh strategie k volbě maxima je klíčový ke správné detekci a řádově ovlivňuje úspěšnost výsledků. Proto si několik používaných strategií popíšeme a otestujeme jejich úspěšnost.

Pro sjednocení detekce vrcholů převedeme všechny detekční funkce na funkce periody $d(\tau)$ definovanou v bodech $0 \leq \tau < N$, takové, že odhad periody je definován jako $\widetilde{T} = \arg\max d(\tau)$. Detekční funkce, jež mají odhad periody v minimu ($d_{min}(\tau)$), převedeme jednoduše použitím jejich opačných hodnot:

$d(\tau) = d_{min}(\tau)$. Převod detekčních funkcí, které mají definiční obor udaný ve frekvenci $d_F(f)$ zase převedeme vztahem $d(\tau) = d_F(N/\tau)$. Tento postup nám však nedá definiční obor stejný jako u funkcích časové domény a tento problém budeme diskutovat v další kapitole.

3.5.3.1 Globální maximum nenulových vrcholů

Některé detekční funkce produkují v oblasti správné periody dostatečně silné vrcholy (ACF, CEPST) a jejich jediným problémem je vrchol v nule. Ten je např. u korelace přirozený, jelikož funkce se sebou nejlépe koreluje, pokud neprovedeme žádný posun. Nám však nepřináší žádnou informaci, proto jej musíme přeskočit. To provedeme následovně:

Předpoklad: ϵ volíme jako přibližně desetinu průměru x_t

```
1 function PeackPickGlob(d, N)
2     i ← 0
3     while i + 1 < N and d(τ) − d(τ + 1) > ϵ do
4         i ← i + 1
5     return arg max_{i≤τ<N} d(τ)
```

3.5.3.2 Heuristika pro oktávové chyby

Pro algoritmus HPS byla použita speciální heuristika pro řešení oktávových chyb. Snaží se opravit chyby, kdy algoritmus zvolil příliš vysoký tón a spočívá v opravě globálního maxima na základě hodnoty maxima v intervalu nižších tónů (De La Cuadra et al., 2001).

```
1 function OctaveHeuristic(d, N, H)
2     i ← arg max_{0≤τ<N} d(τ)
3     j ← arg max_{i<τ<N} d(τ)
4     if j − 2i < ϵ then
5         if d(i)/d(j) > H then          ▷ Autoři článku zvolili H = 0, 2
6             return j
7     return i
```

3.5.3.3 Prahování (Thresholding)

Prahování je obvyklá metoda v oborech zpracování signálu. V algoritmu YIN byla použita při volbě vrcholu detekční funkce CMNDF. Je možné ji však použít u libovolné detekční funkce, jež byla normalizovaná, a tudíž víme, v jakých hodnot by měly vrcholy dosahovat (NCCF, SNAC,...). Spočívá ve stanovení prahu θ a zvolení nejkratší periody, která má hodnotu vyšší:

$$\widetilde{T} = \arg\min\{\tau : d(\tau) > \theta\}.$$

Volba hodnoty prahu je závislá na použité detekční funkci. Např. u CMNDF ji lze interpretovat jako „maximální hodnota výkonu aperiodicity, jež jsem v periodickém signálu ochoten tolerovat“. V algoritmu YIN byla zvolena hodnota θ=0.1.

3.5.3.4 Inteligentní volba vrcholu

V systému pro sledování změn výšky tónu YAAPT byla navržena tzv. „inteligentní metoda volby vrcholu“ Kasi, Kavita et al., 2002. Po několika průchodech vrátí množinu kandidátů pro volbu vrcholu a u každého spočte jeho váhu:

1. Za vrchol označme prvek, jenž je větší než L hodnot vlevo i vpravo od něj. Tedy najdeme takovou množinu vrcholů P, že platí:

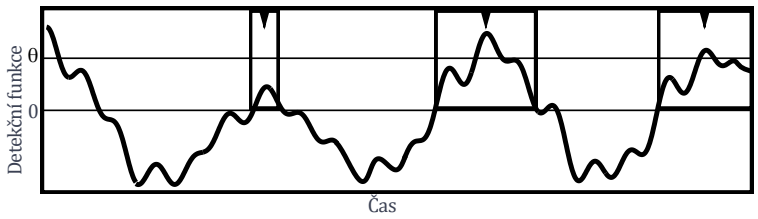
$$P = \{\, \tau : \tau = \underset{\tau - L \leq i \leq \tau + L}{\arg\max}\, d(i) \,\}.$$

2. Na této metodě provedeme prahování s argumentem θ. Z množiny tedy eliminujeme takové τ, že $d(\tau) < \theta$.

3. Z množiny P eliminujeme vrcholy, jejichž vzdálenost od vyššího vrcholu je menší než $D = 2$ ms.

4. Označíme váhu vrcholu P_i jako hodnotu detekční funkce: $w(P_i) = d(P_i)$.

5. Vrcholům P_i, pro které existuje vrchol v jejich dvojnásobku $2P_i$, navýší jejich váhu o hodnotu $w(P_i) = w(P_i) + M$, $M = 0,25$.



Obrázek 3.25: Znázornění McLeodova postupu volby vrcholu

3.5.3.5 McLeodova volba vrcholu

McLeodova metoda (McLeod, 2009, str. 97) využívá faktu, že hodnoty detekčních funkcí založených na autokorelaci se střídavě pohybují v kladných a záporných hodnotách. V kladných intervalech mezi překročením nuly nás pak většinou zajímá jen ta maximální hodnota. McLeodův detekuje tyto tzv. *hlavní vrcholy* a na nich pak provede prahování:

1. Mezi každým pozitivně jdoucím a negativně jdoucím překročením nuly najdi maximum a označ jej za *hlavní vrchol*.

2. Označ práh $\theta = c \cdot \max_{0 \leq \tau < N}(d(\tau))$, parametr $c \in (0, 1)$.

3. Zvol první hlavní vrchol τ, pro který platí $d(\tau) > \theta$.

Tento postup je znázorněn na obr. 3.25. Jsou zde ohraničeny intervaly, ve kterých hledáme maximum z prvního kroku, šipkami jsou pak ukázány hlavní vrcholy.

3.5.4 Zvýšení přesnosti detekce výšky

Ve výše zmíněných algoritmech jsme detekovali výšku v pouze omezené přesnosti. U spektrálních algoritmů jsme detekovali celočíselné frekvence v rámci okna, tedy periody o hodnotách $\frac{N}{f}$ pro $f = 2, 3, \ldots, N/2$. Jinými slovy, detekujeme pouze frekvence, jež jsou násobky vzorkovací frekvence f_s. U algoritmů časové domény jsme zase detekovali pouze periody, jež jsou celočíselným násobkem vzorkovací periody $T_s = \frac{1}{f_s}$. V prvním případě máme k dispozici

hodnoty frekvence v přesnosti f_s Hz a přesnost periody exponenciálně klesá, ve druhém případě zase máme údaje o periodách v přesnosti $\frac{1}{f_s}$ s a přesnost určení frekvence exponenciálně klesá. To nám však většinou nevadí, jelikož i člověk vnímá vzdálenosti ve frekvenci logaritmicky.

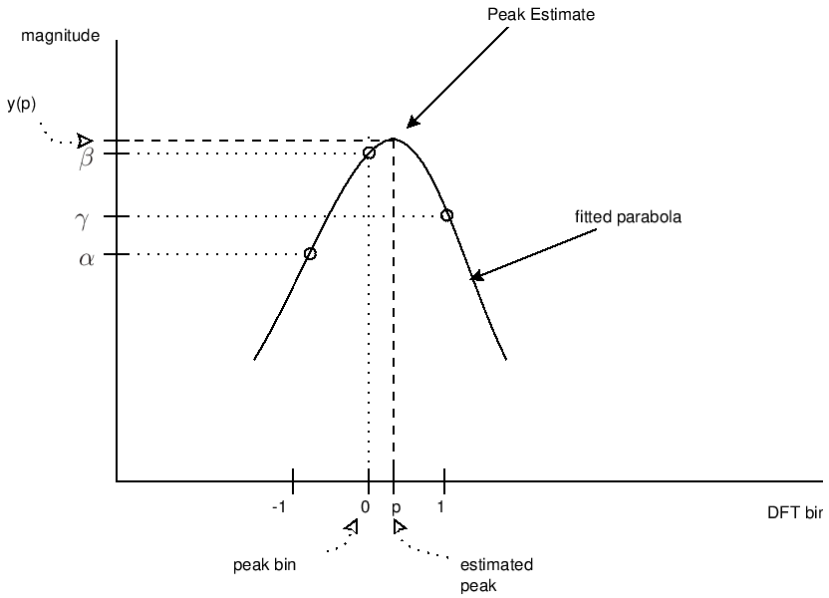
Pokud však bude frekvence v signálu mezi dvěma hodnotami, jež jsme schopni detekovat, bude nám informace o této frekvenci chybět. Většina detekčních funkcí však má tu vlastnost, že se jeho hodnota rozdělí mezi okolní body (ve frekvenční doméně tomu tak je z důvodu již zmíněného efektu prosakování).

Pro zjištění přesné hodnoty tohoto maxima pak většinou používáme interpolační metody. Mezi nejpoužívanější je prokládání parabolou, které vrací dobré výsledky pro časovou i frekvenční doménu (Alain de Cheveigné, 2002). To provedeme tak, že pro všechna lokální maxima použijeme tříbodovou interpolaci kvadratickou křivkou. Hodnoty tohoto bodu a jeho dvou sousedů tedy proložíme parabolou (viz obr. 3.26) a najdeme její maximum. To pak přidáme mezi body detekční funkce. Nalezení parametrů paraboly lze dosáhnout výpočtem jednoduché rovnice, jejíž odvození nalezneme ve Smithově publikaci (2011).

Kromě kvadratické interpolace lze pro zvýšení přesnosti použít též doplnění okna nulami, lineární interpolaci, kubickou B-spline křivkou (McLeod Ph.D. et al., 2003), Barycentrickou metodou, Quinnovými interpolátory či Jainyho metodou (Donadio, 1999). Z důvodu rozsahu této práce a oblíbenosti kvadratické metody však tyto způsoby zde popisovat nebudeme.

3.6 Shrnutí

V této kapitole jsme si popsali metody používané k detekci výšky tónu. Nejprve jsme si tento problém a jeho atributy formálně definovali. Poté jsme určili omezení na fyzikální atributy výšky tónu na základě jejich psychoakustických vlastností a obecné povahy vstupního signálu. Dále jsme si popsali percepční modely vnímání výšky, jež nám pomáhají při návrhu algoritmů. Pro snazší pochopení problému jsme nejprve demonstrovali jeho obtížnost pomocí protipříkladů při snaze navrhnout řešení naivním způsobem. Další kapitola byla věnována základnímu nástroji analýzy signálů – spektrální analýze. V poslední kapitole jsme pak představili sérii algoritmů pro detekci výšky. Jeho návrh



Obrázek 3.26: Znázornění kvadratické interpolace, převzato ze Smithovy publikace (2011)

jsme u většiny rozdělili na proces návrhu detekční funkce a volbu jejího vrcholu. U každé detekční funkce jsme zmínili časovou složitost jejího výpočtu, parametry algoritmu a její požadavek na minimální délku okna. Tyto informace shrnujeme v tab. 3.1. Jelikož je časová složitost všech metod volby vrcholu lineární, zmiňujeme u nich pouze jeho parametry a jejich přehled lze vidět v tab. 3.2.

Algoritmus	Parametry	Min. délka okna	Časová složitost
HPS	M	$2Mf_{max}$	$\mathcal{O}(N \log N + M)$
SHR	M, θ	$2Mf_{max}$	$\mathcal{O}(N \log N + M)$
CEPST	-	$16\,T_{max}$	$\mathcal{O}(N \log N)$
LPM	-	$2T_{max}$	$\mathcal{O}(N)$
AMDF	-	$2T_{max}$	$\mathcal{O}(N^2)$
ACF_A	-	$2T_{max}$	$\mathcal{O}(N^2)$
ACF_B	-	$2T_{max}$	$\mathcal{O}(N \log N)$
ACF_C	-	$2T_{max}$	$\mathcal{O}(N \log N)$
SDF_A	-	$2T_{max}$	$\mathcal{O}(N^2)$
SDF_B	-	$2T_{max}$	$\mathcal{O}(N \log N)$
SDF_C	-	$2T_{max}$	$\mathcal{O}(N \log N)$
NCCF	-	$2T_{max}$	$\mathcal{O}(N \log N)$
CMNDF	-	$2T_{max}$	$\mathcal{O}(N \log N)$
WACF	-	$2T_{max}$	$\mathcal{O}(N \log N)$
SNAC	-	$2T_{max}$	$\mathcal{O}(N \log N)$

Tabulka 3.1: Algoritmy detekce výšky

Algoritmus	Parametry
PeakPickGlob	-
OctaveHeuristic	H
Thresholding	θ
IntelligentPeakPicker	L, θ, D
McLeodMethod	c

Tabulka 3.2: Metody volby vrcholu detekčních funkcí

Kapitola 4

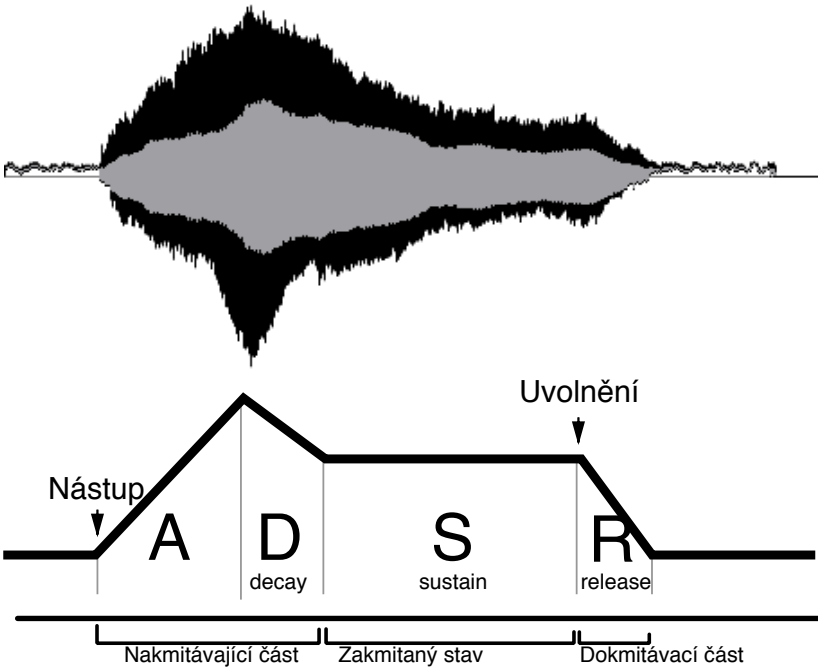
Detekce not

Problém detekce not má za úkol ze záznamu hudební skladby s_t, extrahovat informace o časech nástupu a délkách not, jež se v této skladbě vyskytují. Toto má provést stejným způsobem, jako by to udělal průměrný posluchač.

Pojmy hudební skladba, nota a tón jsme si definovali v kapitole 1.2.1. Zde si nejprve tyto pojmy rozvedeme a podíváme se na stavbu signálu noty a její kontext v hudební skladbě důkladněji. V další části této kapitoly si pak popíšeme problémy, se kterými se musíme při jejich detekci potýkat, a popíšeme si způsoby jejich řešení.

Problém detekce not lze rozdělit na podproblémy detekce nástupu noty a detekce uvolnění noty. Detekce nástupu not je v posledních letech velmi zkoumaný problém, jehož výzkum byl velmi motivován jeho aplikací v kompresi, indexování a ve vytěžování informací (Bello; Daudet et al., 2004, str. 1). Informace o nástupech not má také velké využití v zarovnávání hudby a notového zápisu (score alignment) či dvou hudebních záznamů stejné skladby a synchronizace hudby s počítačovým doprovodem (score following) (Coler et al., 2014).

Detekci času uvolnění noty či detekci její délky však v odborné literatuře téměř nenalezneme. Je to dané zejména nejasnou definicí pojmu uvolnění noty. Pokud se však podíváme na možné situace ukončení noty, je zřejmé, že možnosti jsou pouze dvě. V prvním případě nota nijak výrazně nekončí a přechází v nástup



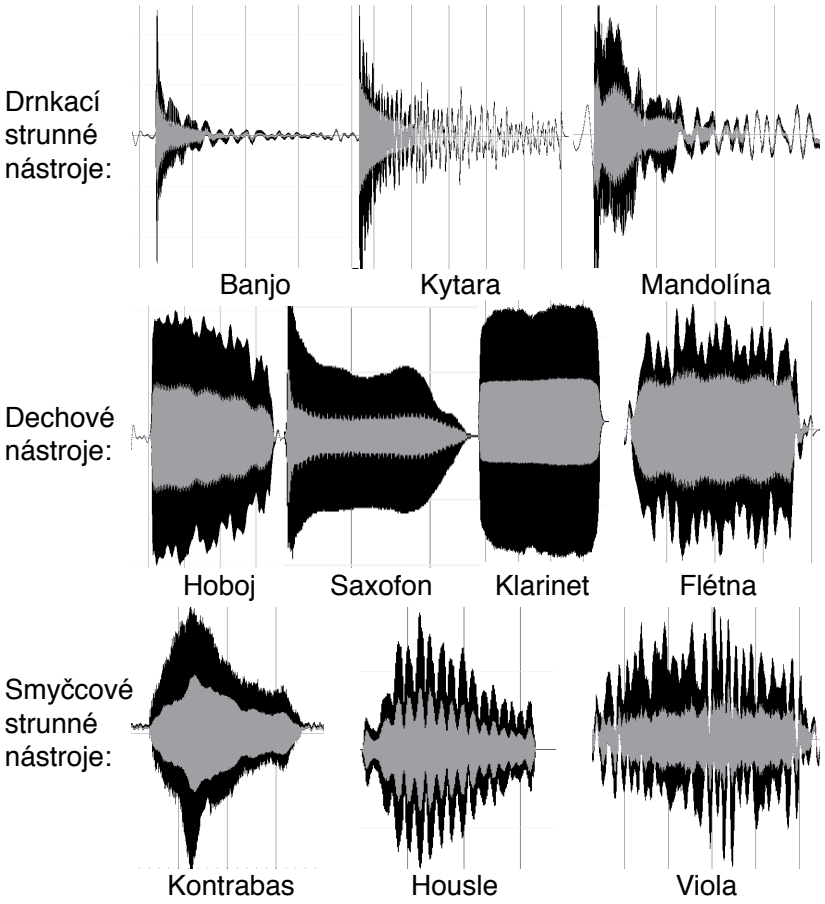
Obrázek 4.1: Znázornění ADSR modelu na záznamu noty G_4 hrané na violu

další noty, čímž je problém převeden na detekci dalšího nástupu. Ve druhém případě nota skončí samovolně a přechází v ticho. Naším úkolem je tedy detekovat tzv. znělé a neznělé úseky skladby (voiced/unvoiced decision), což je často řešený problém v analýze řeči.

4.1 Základní pojmy

Hudební událost Na hudební signál lze pohlížet jako na sekvenci hudebních událostí. Tyto události si definujeme jako libovolné změny v určitém parametru signálu, jež mají hudební význam. Za základní událost monofonních skladeb lze považovat notu, kterou je jednoznačně určena událostmi jejího nástupu (začátku) a uvolnění (konce).

Nota V kontextu hudebních událostí je nota takový segment, který je vnímán jako nepřerušovaný zvuk generovaný hudebním nástrojem. Stavbu noty lze obecně popsat ADSR modelem.



Obrázek 4.2: Tvary obálek signálu noty G_4 u různých hudebních nástrojích

ADSR Model K analýze stavby noty můžeme využít ADSR model (Bello; Daudet et al., 2004). Ten popisuje čtyři části ideální formy noty – nástup, útlum, podržení a uvolnění. Jednotlivé úseky by měly být doprovázeny specifickým tvarem obálky signálu (křivka jež prochází extrémy časového průběhu signálu). Tento tvar namapovaný na konkrétní signál noty G_4 hrané na violu lze vidět na obr. 4.1. Černá oblast časového průběhu odpovídá maximu a minimu signálu, šedá pak průměru hodnoty přes krátký časový úsek. Tvar ADSR křivky je však pouze orientační, z obr. 4.2, kde jsou znázorněny signály noty G_4 hrané na různé nástroje, můžeme vypozorovat, že se tvary obálek oproti modelu i vzájemně výrazně liší.

Mnohem obecnější je dělení z hlediska tvorby zvuku na *nakmitávací*, *zakmitanou* a *dokmitávací* část noty. V nakmitávací i dokmitávací části dochází k rychlým změnám ve velkém rozsahu. Proto ji označujeme jako *tranzientní* (přechodovou) část noty. V zakmitané části se oproti tomu dějí pouze malé změny v malé rychlosti, proto tento úsek označujeme za *kvazistacionární* část tónu. Perkusní nástroje zakmitanou část tónu neobsahují. (Syrový, 2013)

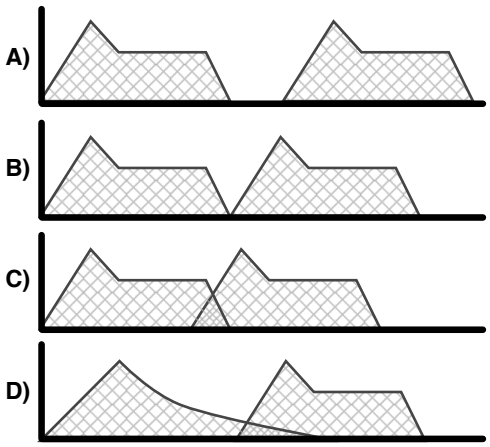
Nástup (*attack*) označuje část signálu, kdy je nástroji dodána první vlna energie k tvorbě zvuku (např. proud vzduchu u dechových nástrojů, či pohyb smyčce u smyčcových nástrojů). Nástup je často doprovázen prudkým vzrůstem obálky (zejména u perkusních nástrojů), u některých nástrojů (např. smyčcových) tomu však být nemusí (viz obr. 4.2).

Útlum (*decay*) pak značí část signálu, kdy obálka signálu klesá na nějakou hodnotu, kde se ustálí. Stále jde však o nakmitávající stav a ve spektrální doméně i v obálce dochází k velkým a výrazným změnám.

Podržení, neboli *sustain* je interval při kterém je nástroji dodáváno přibližně konstantní množství energie a je nositelem „statické“ informace o výšce, barvě a hlasitosti tónu. Přesto není tvar této části signálu naprosto statický. Promítají se zde tzv. artikulační efekty tónů, jež většinou ovládá samotný hráč. Mezi ně patří např. *tremolo*, což je periodická modulace amplitudy, či *vibrato*, které spočívá v periodické změně výšky tónu.

Uvolnění (*release*), neboli dokmitávací část, odpovídá časovému úseku od ukončení dodávání energie nástroji po čas, kdy přestane vydávat zvuk. Tento segment je závislý na způsobu tvorby zvuku. Např. dechové nástroje mají čas uvolnění téměř okamžitý, jelikož po zastavení proudu vzduchu není nic rozkmitáváno. Oproti tomu strunné nástroje vydrží být rozkmitané po mnohem delší dobu.

Délku noty pak v kontextu tohoto modelu považujeme za časový úsek mezi počátkem nástupu a počátkem uvolnění. Pojmy nástup (onset) a uvolnění (offset) pak můžeme použít též v kontextu konkrétního časového okamžiku a myslíme tím časy počátků těchto segmentů.



Obrázek 4.3: Vzájemné pozice dvou not (inspirováno prací Colera et al., 2014)

Znělost úseku signálu je pojem přenesený z fonetiky a analýzy řeči (Bachu et al., 2008). U hudebních signálů definujeme znělý úsek jako takový, který má definovanou výšku. Tato výška by navíc měla pocházet z harmonického hudebního nástroje a neměla by tedy být atributem okolního šumu. Neznělý úsek oproti tomu obsahuje neperiodické zvuky působící náhodně. Libovolný časový okamžik pak můžeme označit jako znělý či neznělý na základě toho, jakému úseku signálu přísluší.

Tranzient či přechod představuje výraznou změnu, jež se většinou odehrává, jak bylo výše zmíněno, při nástupu, útlumu či uvolnění noty. Tato percepčně rychlá a velká změna však může být souhrou malých změn v časové a frekvenční doméně a může být dána jejich vzájemným postavením, proto nemusí být intuitivně nikterak zřetelná.

4.2 Analýza problému

Vzájemné postavení not Pokud se podíváme na vzájemné pozice dvou not v monofonním signálu, Coler (2014) popisuje následující základní čtyři možnosti, které jsou znázorněny na obr. 4.3.

A) Noty oddělené pomlkou Detekce těchto not je snadná, jelikož spočívá pouze v binární klasifikaci úseku na znělý či neznělý, a i když by byly tranzienty not nevýrazné, jejich detekci nebude třeba řešit. Proto se pokusíme v první řadě detekovat tuto možnost.

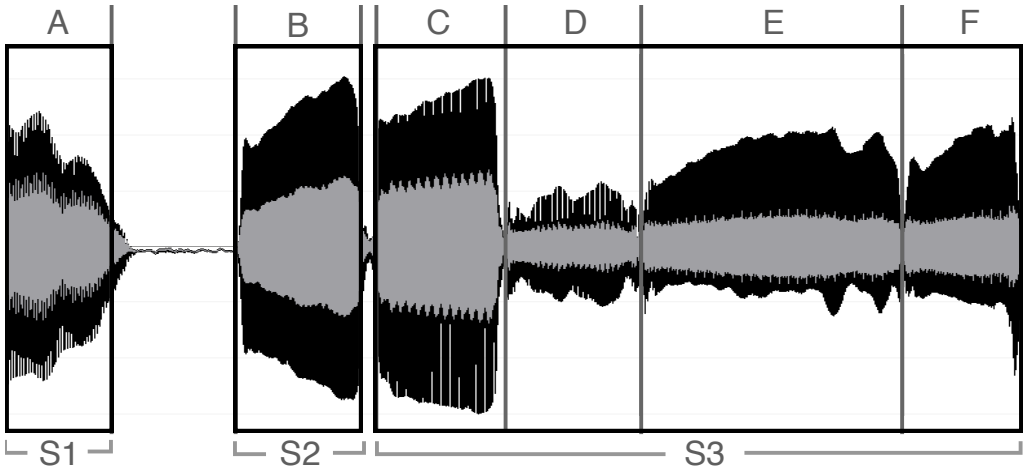
B) Oddělené noty Pokud nástup noty nastane až po výrazném tranzient uvolnění, většinou obálka má tvar výrazného poklesu a nástupu, jelikož dochází k výrazným změnám energie v signálu. Proto detekce těchto událostí není obtížná.

C) Legato, neboli vázaně, je technika hraní dvou not tak, aby jejich přechod zněl co nejvíce hladce. Hudebník toho dociluje například tím, že změnu tónu na dechovém nástroji provede v jednom dechu, či že na smyčcovém nástroji nepřeruší tah smyčce. Pro tuto techniku existuje speciální instrukce v hudební notaci a je velmi hodně využívána. Pro tuto techniku nemusí fungovat metoda detekce z obálky signálu, protože obzvláště na smyčcové nástroje lze dosáhnout přechodu tak hladkého, že je tvar obálky spojitý. Jelikož se však změna výšky tónu musí projevit ve spektru, je možné v něm tuto změnu detekovat.

D) Pozvolný tranzient Může nastat případ, že první notě chybí úsek podržení i uvolnění a po celou dobu setrvává v pozvolném útlumu. Pak nedochází k žádnému konečnému tranzientu a pro detekci jejího konce musí být detekován až tranzient nástupu navazující noty. Ten však může být nedokončeným útlumem částečně maskován.

Na základě výše uvedených superpozic not a také kvůli způsobu řešení těchto problému v odborné literatuře si rozdělíme problém detekce not na následující dva podproblémy: detekce nástupů not a rozhodnutí o znělosti signálu.

Rozhodnutí o znělosti signálu U jednodušších vstupů, kdy jednotlivé noty jsou oddělené úseky ticha, lze celý problém vyřešit za pomoci segmentace signálu na základě znělosti. Existuje mnoho způsobů, jak tuto analýzu provést a popíšeme si je v následující kapitole.

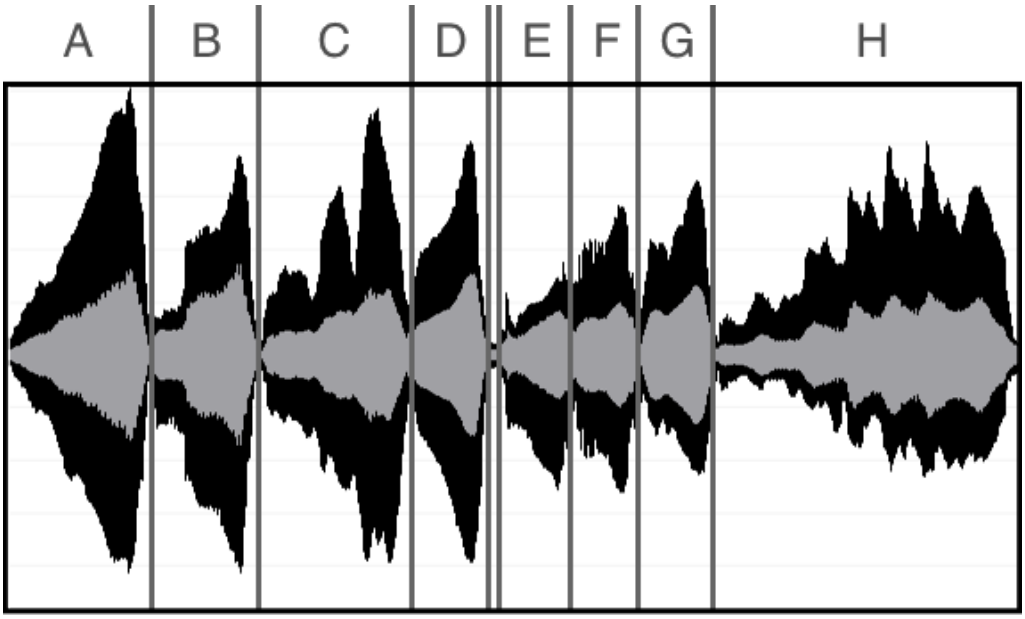


Obrázek 4.4: Detekce not na skladbě Ach Gott und Herr (Bach, 1893a) hrané na fagot

V první fázi tedy můžeme rozdělit signál na znělé a neznělé segmenty. U těch znělých však musíme detekovat, zda neobsahují více not, jež nejsou přerušeny úsekem neznělým. Ukázku tohoto postupu můžeme vidět na obr. 4.4 na úseku z nahrávky skladby Ach Gott und Herr (Bach, 1893a) hrané na fagot. Detektor znělosti nám detekoval tři znělé úseky (S1, S2 a S3). Úseky S1 a S2 odpovídají samostatným notám a detekce je zde tedy dokončena. Úsek S3 však obsahuje další čtyři noty.

Detekce nástupu not Detekce nástupu noty spočívá v hledání tranzientních událostí v hudebním signálu. Naším cílem je identifikovat čas události, kdy hráč začal dodávat nástroji energii. Způsob, jakým se taková událost projeví, je velmi závislý na nástroji (Bello; Daudet et al., 2004).

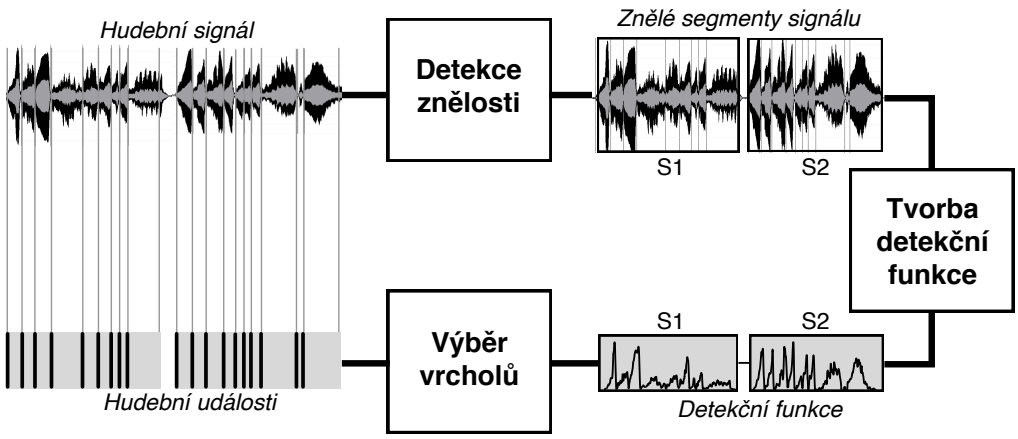
Určitě můžeme označit za nástup počátek segmentu, rozpoznání těch dalších však nemusí být tak snadné. Na obr. 4.4 lze hranici mezi notami C a D, či E a F snadno vyčíst i z tvaru obálky, která prokazuje výrazné rozdíly v maximální amplitudě. Hranice mezi notami D a E však tak snadno čitelná není. Na obr. 4.5 je tatáž skladba hraná na housle. Zde úsek, jež odpovídá notě C a úsek, jež odpovídá notám E a F je dle tvaru obálky velmi podobný. Jeden však odpovídá jedné notě, zatímco druhý obsahuje noty dvě.



Obrázek 4.5: Detekce not na skladbě Ach Gott und Herr (Bach, 1893a) hrané na housle

Z příkladů je proto zjevné, že nestačí hledat tranzientní úseky pouze v obálce signálu. Scheirer tento jev popsal ve své psychoakustické hypotéze vnímání rytmu, kdy tvrdí, že lidská sluchová soustava analyzuje různé frekvenčními pásma odděleně (Klapuri, 1997, str. 18). Protože je člověk schopen nástupy not bez obtíží rozpoznat, dle stávajících modelů vnímání zvuku se v harmonické či dynamické rovině tyto změny projevit musí (Bello; Daudet et al., 2004). Naším úkolem tedy bude vytvořit takový algoritmus, který zvolí vhodný kompromis při hledání tranzientních událostí tak, aby detekoval nástupy a uvolnění not a zároveň toleroval artikulační efekty v zakmitané fázi tónu.

Psychoakustické vlastnosti detekce not Klapuri (1997, str. 17) píše, že lidské ucho je schopné rozpoznat navazující nástupy not pouze tehdy, je li mezi nimi prodleva alespoň 20 ms. Pro kratší intervaly dochází k jejich tzv. maskování, kdy silnější nástup z hlediska *prominence* zakryje ten slabší. Prominencí je myšlen percepční vjem intenzity dané noty v kontextu okolních not a je závislá na barvě, čistotě tónu i časovém kontextu (Rosenthal, 1992). Limit 20 ms je



Obrázek 4.6: Schéma systému pro detekci not

však pouze dolní odhad. Rosenthal (1992) popisuje jev zvaný *segregace hudebních proudů* (stream segregation), který popisuje způsob vnímání zvukových událostí v závislosti na čase důkladněji. Na základě experimentu bylo zjištěno, že hranice vzdálenosti percepčního sloučení dvou tónů stoupá s rozdílem frekvence těchto zvuků.

Schéma detekčního systému Na základě předchozí diskuze tedy navrhneme detekční systém jako několikafázový proces. V první fázi detekujeme znělé segmenty. Počátek, resp. konec znělých segmentů označíme za hudební události nástupu, resp. uvolnění noty. Tyto segmenty budou vstupovat do druhé fáze, ve které vytvoříme systém detekce nástupů not v rámci daného segmentu. Nejčastěji se pro tento systém využívá koncept *detekční funkce* (Bello; Daudet et al., 2004), jenž jsme využili i u některých algoritmů detekce výšky. Jeho cílem bude ze vstupního signálu vytvořit funkci závislou na čase, jež bude v oblasti nástupů tónů dosahovat vysokých hodnot. Ve třetí fázi budeme tyto vysoké hodnoty vybírat na základě různých strategií volby lokálního maxima. Detekované časy pak označíme za časy uvolnění aktuální noty a nástupy noty následující.

Rozdíl mezi touto detekční funkcí a funkcí z detekce výšky je, že naším úkolem bude detekovat těchto lokálních maxim tolik, kolik je v signálu tónů. Proto ani detekční funkce ani metody volby maxima z detekce výšky nelze použít pro tento problém. Další rozdíl spočívá v časové přesnosti, oproti 2 ms ze systému detekce výšky nám tady postačí přesnost 20 ms. Aby byl systém

schopný pracovat v reálném čase, je třeba provádět detekci maxima průběžně. Tato úprava však tento systém výrazně komplikuje, protože nemůžeme při volbě maxima využít celý kontext. Jelikož jsme si v kap. 2.7 stanovili nejzazší čas odezvy detekovaného tónu na 0,5 s, rozdělíme si signál na takto dlouhé úseky.

Schéma tohoto systému lze vidět na obr. 4.6. Je zde znázorněn postup detekce hudebních událostí pro jeden krátký segment signálu. V následující kapitole si jednotlivé moduly popíšeme.

4.3 Algoritmy detekce not

V této kapitole se podíváme na přístupy k detekci znělosti i k detekci nástupů not. K oběma problémům přistoupíme metodou návrhu detekční funkce a prahování. Rozdíl mezi nimi bude spočívat ve způsobu prahování, zatímco u znělosti rozhodneme, že libovolné hodnoty nad nějaký stanovený práh jsou znělé, při detekci nástupů označíme hodnoty nad prahem jako nástup. To nám ve znělém segmentu umožní rozpoznat noty, jež nejsou odděleny segmenty neznělými.

4.3.1 Detekce znělosti

Úkolem modulu detekce znělosti je rozhodnout, zda v daném časovém úseku signálu je slyšet nějaký tón. Ideální by bylo detekovat hlasitost daného segmentu a zkombinovat ji s detekcí výšky. Vjem hlasitosti tónu je však vnímán relativně v kontextu okolních zvuků a člověk je většinou schopný pouze říci, zda je jeden segment hlasitější, než druhý. Odhad rozdílu je závislý na intenzitě, výšce a barvě daných tónů (Syrový, 2013, str. 72). Jedním z intuitivních způsobů, jak provést detekci znělosti, by také bylo využít zmíněné algoritmy detekce výšky a zjistit, zda byla výška detekována dostatečně jednoznačně. Tento systém však nemusí fungovat u tónů, jež jsou příliš tiché na to aby je člověk slyšel. Proto se tento problém většinou řeší kombinací detekce těchto dvou atributů hudebního signálu.

4.3.1.1 Základní metody využívající tvar obálky

Pro porovnání při testování zde nejprve popíšeme základní metody, které se snaží napodobit postup, jenž člověk provádí při snaze detekovat znělé úseky signálu ze křivky. Definujme si tyto dvě základní funkce:

$$\begin{aligned} \mathrm{ENV}_{max} &= \max_{0 \leq t < N} x_t, \\ \mathrm{ENV}_{rct} &= \frac{1}{N} \sum_{t=0}^{N} |x_t|, \\ \mathrm{ENV}_{sqr} &= \frac{1}{N} \sum_{t=0}^{N} (x_t)^2, \\ \mathrm{ENV}_{log} &= \frac{1}{N} \sum_{t=0}^{N} \log(|x_t|). \end{aligned} \tag{4.1}$$

Metoda ENV_{max} rozpoznává znělost na základě maximální hodnoty ve vstupním segmentu. Funkce ENV_{rct} a ENV_{sqr} pak vrací průměr absolutních hodnot resp. druhých mocnin signálu. Funkce ENV_{log} se pak snaží využívat vlastnosti, že hlasitost vnímáme logaritmicky a vrací průměr logaritmu absolutní hodnoty signálu. Ačkoliv tyto funkce fungují na čistých nahrávkách tónů bez zvuků v pozadí velmi dobře, stačí, aby šum překročil stanovený práh a jejich schopnost rozpoznat znělost se vytrácí.

4.3.1.2 Metoda využívající závislosti energie a frekvence překročení nuly

Bachuova metoda (2008) využívá dvou metrik, jež vykazují rozdílné chování pro znělé a neznělé úseky signálu. Jde o metriky *frekvence překročení nuly* (zero-crossing rate, ZCR) a *krátkodobá energie signálu* (short-time energy, STE). Frekvenci překročení nuly na okně signálu x_t délky N definujeme následovně:

$$\mathrm{ZCR}(x_t) = \frac{1}{N-1} \sum_{t=2}^{N-1} \mathbf{1}_{\mathbb{R}_{<0}}(x_t x_{t-1}), \tag{4.2}$$

Kde $\mathbf{1}_A(x)$ je charakteristická funkce (rovná jedné když $x \in A$, jinak nule). ZCR udává průměrný počet překročení na délku okna a lze jej využívat u jednoduchých signálů i pro detekci výšky tónu (A Cheveigné, 2008). Bylo vypozorováno, že ZCR bývá výrazně vyšší u znělých signálů, než u signálu neznělých (Bachu et al., 2008).

Druhou metrikou je krátkodobá energie signálu, která reflektuje změnu amplitudy v čase. Ta je definována následovně:

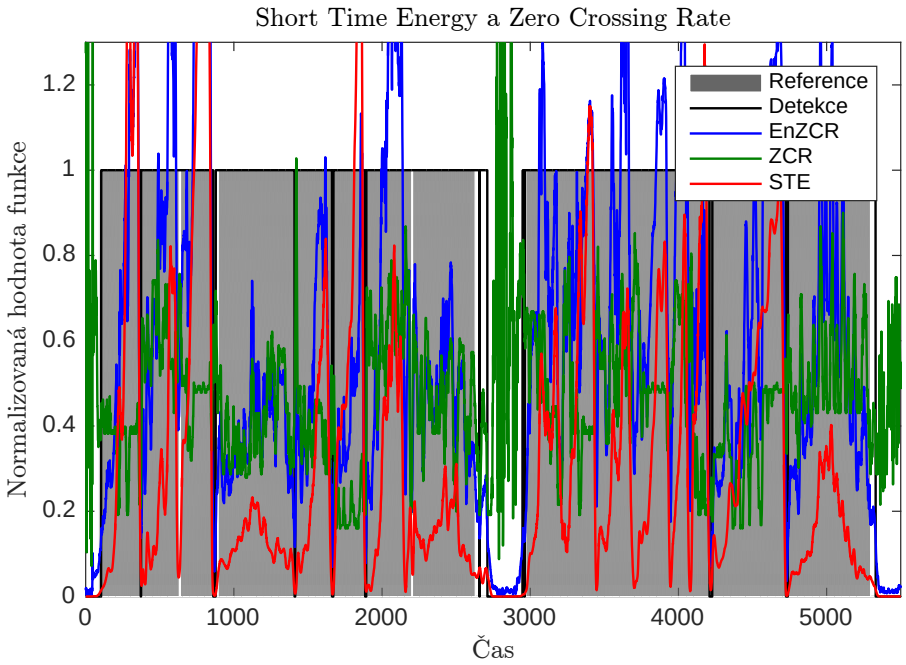
$$\mathrm{STE}(x_t) = \sum_{t=0}^{N-1} x_t^2 \tag{4.3}$$

Krátkodobá energie je přirozeně větší pro znělé signály než pro signály neznělé. Jelikož je však vnímání hlasitosti závislé na frekvencích, jež se v signálu vyskytují, není tato metrika postačující.

Autoři článku detektor postavený na těchto funkcích více nepopisují. Proto navrhujeme následující řešení. Definujme funkci závislosti krátkodobé energie a frekvence překročení nuly (EnZCR) následovně:

$$\mathrm{EnZCR}(x_t) = \mathrm{STE}(x_t) \cdot \sqrt{\mathrm{ZCR}(x_t)} \tag{4.4}$$

Tato funkce byla odvozena na základě pozorování chování metrik ZCR a STE na různých signálech. Na obr. 4.7 je znázorněn začátek skladby Ach Gott und Herr (Bach, 1893a) hraný na housle, kde šedé bloky představují úseky jednotlivých not. Dále zde vidíme funkce ZCR (zelená) a STE (červená) vygenerované při velikosti okna 1024 vzorků. Modrou barvou je pak znázorněna funkce EnZCR a je zde vidět, že vhodně přebírá správné hodnoty obou funkcí, aniž by se nechala vychýlit jejich chybami. Detektor znělosti pak pro každý úsek signálu vyhodnotí funkci EnZCR. Pokud hodnota přesáhne pevně stanovený práh θ, označíme tento úsek za znělý. Jinak jej budeme považovat za neznělý.



Obrázek 4.7: Znázornění závislosti krátkodobé energie a frekvence překročení nuly

Při testování na reálných signálech se ukázala vhodná hodnota $\theta = 10^{-3}$. Černá linka pak reprezentuje výsledek našeho detektoru a vidíme, že úseky většiny not správně identifikoval. Důkladnější otestování této metody naleznete v kap. 4.3.1.

4.3.1.3 Normalizovaný poměr energie v nízkých frekvencích

Funkce NLFER (v originále Normalized Low-frequency Energy Ratio) byla využívána v již zmiňovaném systému YAAPT k detekci znělosti Kasi, Kavita et al., 2002. Spočívá v měření energie pouze v nízkých frekvencích spektra a srovnávání těchto hodnot s hodnotami okolních časů. Její hodnota v čase t je definovaná následovně:

$$\mathrm{NLFER}(t) = \frac{\sum_{k=A}^{B} X_k(t)^2}{\frac{1}{N}\sum_{n=0}^{t}\sum_{k=A}^{B} |X_k(n)|^2}, \tag{4.5}$$

kde $X_k(t)$ odpovídá spektrálnímu koeficientu k v čase t a parametry A, B určují interval frekvencí, na kterých energii měříme. Volba intervalu $[A, B]$ však není jednoduchá. Autoři článku tuto funkci využívali pro detekci znělosti signálů řeči, jejichž výška se pohybuje v intervalu 40 Hz - 500 Hz (McLeod, 2009, str. 37). Pro hudební signály je však tento rozsah mnohem větší a pokud bychom jej obsáhli celý, počítáme energii v téměř celém spektru. Pokud však nezvolíme celý rozsah, hrozí nebezpečí, že označíme úseky vysokých tónů za neznělé.

4.3.1.4 Spektrální rozptyl

Jelikož spektrum znělého signálu vykazuje výrazné hodnoty v oblastech harmonických koeficientů, zatímco spektrum u neznělých signálů působí téměř náhodně (Syrový, 2013), nabízí se možnost tohoto faktu využít pro detekci znělosti. Definujme proto hodnotu SPECTVAR (spectral variance) následovně:

$$\text{SPECTVAR} = \frac{1}{N}\sum_{k=0}^{N/2}(X_k - \mu)^2, \text{ kde} \quad \mu = \frac{1}{N}\sum_{k=0}^{N/2}|X_k| \tag{4.6}$$

a kde X_k odpovídá k-tému spektrálnímu koeficientu.

4.3.2 Detekční funkce nástupů not

Naším úkolem při návrhu detekční funkce je vytvořit takovou funkci, jež dosahuje lokálního maxima v časech nástupů not. Měla by to dělat tak, aby existoval takový práh θ, že by všechna maxima nástupů not jej převyšovala. Zároveň by však mělo platit, že hodnota detekční funkce mimo oblasti nástupů not je menší než tento práh θ. Při detekci nástupů not hledáme tranzientní oblasti signálu. Tyto přechody mohou být jak v energii signálu, tak v jeho spektrální podobě. V této kapitole si představíme detekční funkce, které se k řešení tohoto problému běžně využívají.

4.3.2.1 Sledování tvaru obálky

Energii signálu, která určuje tvar obálky, jsme využívali u detekce znělosti. Jelikož je však nástup not často doprovázen prudkým vzrůstem energie, ukazuje se, že za použití dobré detekce volby maxima jí lze využít i pro detekci nástupu not. Proto pro detekci nástupu zkusíme použít i základní detekční funkce založené na tvaru obálky. Použijeme tedy funkce ENV_{max}, ENV_{rct}, ENV_{sqr} a ENV_{log} definované v předchozí kapitole.

4.3.2.2 Detekce změny tvaru obálky

Pokud se podíváme na tvary obálek na obr. 4.2, lze vidět, že u většiny nástrojů dochází k největší změně na začátku tónu. Derivace tvaru obálky by tedy měla dosahovat vysokých hodnot. Pro reprezentaci obálky použijeme funkci ENV_{log}, jelikož logaritmus sníží vliv změn ve tvaru obálku v zakmitané části noty. Definujme proto funkci derivace obálky (ENVDERIV) následovně:

$$\text{ENVDERIV} = \frac{d(log(\text{ENV}_{log}))}{dt}. \tag{4.7}$$

4.3.2.3 Funkce energie vysokých frekvencí

Jednou z nejpoužívanějších metod pro detekci nástupů not je nepochybně High Frequency Content (HFC, Bello; Daudet et al., 2004). Ta využívá faktu, že tranzienty jsou vnímány mnohem výrazněji u vyšších frekvencí. Proto tato metoda váží spektrální koeficienty jejich indexem:

$$\text{HFC} = \frac{1}{N}\sum_{k=1}^{N/2} k(X_k)^2, \tag{4.8}$$

kde X_k odpovídá k-tému spektrálnímu koeficientu.

4.3.2.4 Spektrální vzdálenost

Obecnější přístup hledá tranzienty pomocí detekování změn ve spektru. Tyto změny definuje na základě vzdálenosti dvou sousedních spekter (spectral difference, SD). K výpočtu vzdálenosti se většinou používá ℓ_1 norma či ℓ_2 norma (euklidovská vzdálenost).

$$\begin{aligned} \mathrm{SD}_{\ell_1}(n) &= \sum_{k=1}^{N/2} |H(X_k(n) - X_k(n-1))| \\ \mathrm{SD}_{\ell_2}(n) &= \sqrt{\sum_{k=1}^{N/2} H(X_k(n) - X_k(n-1))^2}, \\ &\text{kde } H(x) = \frac{x + |x|}{2}. \end{aligned} \tag{4.9}$$

Člen $H(x)$ (nulový pro záporné hodnoty) má zde ten efekt, že jsou ve vzdálenosti započítány pouze kladné změny, což opět využívá faktu, že energie při nástupu noty prudce stoupá.

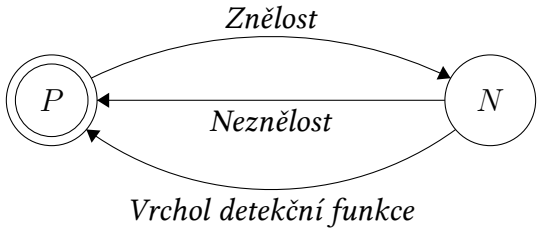
4.3.2.5 Vzdálenost ve fázi

Dosud jsme ze spektra využívali pouze informaci o amplitudě. V kap. 3.4 jsme si však popsali, že spektrum obsahuje též informace o fázi jednotlivých spektrálních komponent. Syrový (2013, str. 111) definuje *okamžitou frekvenci* jako změnu okamžité fáze signálu $\varphi(t)$ v čase t:

$$\omega(t) = \frac{d\varphi(t)}{dt} \tag{4.10}$$

Jelikož se předpokládá, že pro kvazistacionární signál by měla být okamžitá frekvence relativně konstantní (Bello; Daudet et al., 2004), můžeme předpokládat, že rozdíl fáze mezi spektry dvou sousedních oken kvazistacionárního signálu by měl být téměř konstantní:

$$\varphi_k(t) - \varphi_k(t-1) \simeq \varphi_k(t-1) - \varphi_k(t-2), \tag{4.11}$$



Obrázek 4.8: Konečný automat detektoru not

kde $\varphi_k(t)$ odpovídá fázi k-tého spektrálního koeficientu t-tého okna. Fázovou odchylku pak definujeme diferenciální rovnicí:

$$\Delta\varphi_k(t) = \varphi_k(t) - 2\varphi_k(t-1) + \varphi_k(t-2) \tag{4.12}$$

Jelikož během tranzientních regionů není okamžitá frekvence $\omega(t)$ dobře definovaná, fázová odchylka $\Delta\varphi_k(t)$ bude v těchto úsecích dosahovat vysokých hodnot. U kvazistacionárních úseků pak bude platit, že $\Delta\varphi_k(t) \simeq 0$. Fázová odchylka tak splňuje požadavky detekční funkce. Proto označíme hodnotu PD (Phase Deviation) jako součet fázových odchylek spektrálních koeficientů ve spektru t-tého okna:

$$\mathrm{PD} = \sum_{k=0}^{N/2} \Delta\varphi_k(t). \tag{4.13}$$

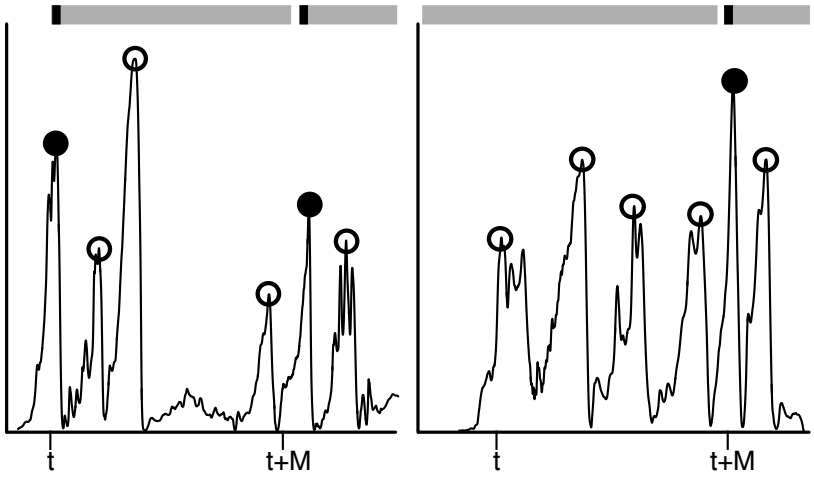
4.3.3 Detektor not

Detektor not má na vstupu informace o znělosti a neznělosti signálu, pro znělé segmenty pak má spočtenou detekční funkci. Jeho úkolem je s minimálním zpožděním detekovat nástupy a konce not. Postup detekce lze reprezentovat jednoduchým konečným automatem o dvou stavech *nota* (N) a *pomlka* (P). Přechody pak tvoří informace o změnách znělosti a o vrcholech v detekční funkci (viz obr. 4.8). Při každé aplikaci přechodu dojde k detekování události nástupu či uvolnění noty. Pro jednoduchost jsme z obrázku vynechali přechody do stejného stavu. Přechod vrcholem detekční funkce z noty do pomlky je z důvodu, že pro detekování nové noty musíme nejprve stávající notu přerušit. Pokud je následující úsek stále znělý, bude následovat okamžitý přesun do stavu nové noty.

Nyní se budeme věnovat samotné volbě vrcholů detekční funkce. Pokud jsou detekční funkce správně navrženy, měly by dosahovat lokálního maxima v časech nástupů not. Pokud tedy existuje takový práh θ, že detekční funkce je vyšší pouze v oblastech nástupů not, stačí nám tento práh nalézt. Jelikož však změnu hlasitosti a výšky, a tedy i sílu tranzientu člověk vnímá v závislosti na kontextu, je zapotřebí robustnějšího algoritmu volby vrcholů (Bello; Daudet et al., 2004).

V zakmitané fázi noty totiž také může dojít k určitým tranzientům, pokud je nota hrána s nějakým artikulačním efektem (např. vibrato či tremolo). Volba vrcholů by však tyto tranzienty měla ignorovat. Také může nastat situace, kdy během tranzientu nástupu či uvolnění detekční funkce vytvoří sérii lokálních maxim. Detektor vrcholů by měl při nástupu noty ignorovat všechna tyto maxima po dobu minimální délky noty M. To bude dělat následujícím způsobem. Pokud se nachází ve stavu pomlky, přijme první detekované maximum a po dobu M ignoruje všechna následující maxima. Pokud se nachází ve stavu noty, přijme až poslední maximum v úseku délky M, jelikož tato maxima můžou odpovídat tranzientu útlumu. Tento postup lze vidět na obr. 4.9, kde jsou v horní části znázorněné detekované noty (černá značí její nástup, šedá pak její délku). Pod nimi lze vidět výstup detekční funkce v těchto sporných situacích. Detekované vrcholy jsou znázorněny kruhy, kde vyplněný kruh značí vrchol zvolený jako nástup. V levém obrázku vidíme první situaci, kdy ze stavu pomlky byl zvolen první vrchol v časovém intervalu délky M. V pravém obrázku pak vidíme druhou situaci, kdy ze stavu noty byl jako nástup další noty zvolen až poslední vrchol ve vzdálenosti délky M od prvního vrcholu.

V následujících podkapitolách se budeme věnovat strategii rozpoznávání těchto vrcholů. Budeme popisovat algoritmy, které mají za vstup detekční funkci a se zpožděním kratším než 0,5 s vrací informace o nástupech a uvolněních not.



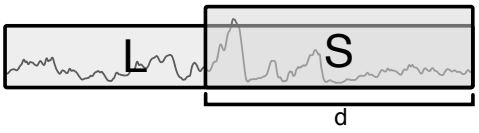
Obrázek 4.9: Problémy více vrcholů v detekční funkci

4.3.3.1 Pevně stanovený práh

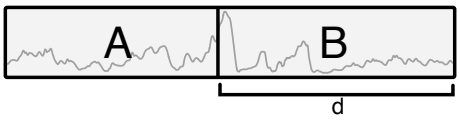
Pro dobře normované detekční funkce by mělo stačit zvolit nějaký práh θ, který by umožnil detekovat všechny nástupy not spolehlivě pro libovolný signál. Proto označme AbsoluteThreshold algoritmus, který společně s inteligentním stavovým automatem popsaným v předchozí kapitole vrací informace o nástupech a uvolněních not pouze na základě informace, zda hodnota detekční funkce přesahuje tento práh. Konkrétní pozici vzorku, jež označíme za nástup noty, určíme následovně. Nechť detekční funkce $d(t)$ přesáhla práh θ v čase t_0. Pak pro čas nástupu noty T_0 platí:

$$T_0 = \underset{t_0 \leq t < t_0 + M}{\arg\max}\ d(t) \tag{4.14}$$

Tuto metodu opravení pozice vrcholu budou používat i všechny následující detektory. Pokaždé, když detekují vrchol v čase t, počkají na hodnoty detekční funkce do času $t + M$ a zvolí její maximální vrchol. To nám umožní využívat detailní přesnosti detekční funkce a zároveň používat hrubější metody pro detekování tranzientu v širším kontextu.



(a) Adaptivní filtr dolních propustí



(b) Derivace klouzavého průměru

Obrázek 4.10: Znázornění detektorů vrcholů

4.3.3.2 Adaptivní filtr dolních propustí

Jelikož hudba obecně vykazuje výrazné změny hlasitosti v průběhu skladby a detekční funkce jsou téměř vždy na hlasitosti závislé, pevně stanovený práh pro tyto výchylky buďto mine vrcholy tišších úseků skladby, nebo detekuje nadbytek nástupů not v hlasitější části skladby. Proto je naším cílem vytvořit adaptivní práh, jenž se bude upravovat v závislosti na kontextu. Jednou z možností návrhu adaptivního prahu je využít dvou nezávislých filtrů dolní propusti (McLeod Ph.D. et al., 2003, str. 115). Tuto metodu můžeme generalizovat výpočtem průměru dlouhého a kratšího okna končícího na aktuální pozici. Nechť hodnota filtru $\mathcal{L}(t)$ délky L a filtru $\mathcal{S}(t)$ délky S je v čase t definována následovně:

$$\mathcal{L}(t) = \frac{1}{L} \sum_{i=t-L}^{t} x_i, \qquad \mathcal{S}(t) = \frac{1}{S} \sum_{i=t-S}^{t} x_i, \tag{4.15}$$

a platí, že $L > S$, jako na obr. **??**. Na základě těchto filtrů pak definujeme následující metody volby vrcholů. Algoritmus FilterDifference detekuje vrchol tehdy, nastane li situace, že $S(t) - L(t) \geq \theta$, kde θ je pevně stanovený práh. FilterRatio zase detekuje vrchol v situaci, když platí $S(t)/L(t) \geq \theta$. Tyto algoritmy se v ostatních aspektech neliší od předchozího algoritmu. Pokud detektor v čase t rozpozná vrchol, může o něm informovat se zpožděním $d+M$, kde d značí zpoždění kratšího filtru délky S a M je minimální délka noty.

4.3.3.3 Derivace klouzavých průměrů

Alternativní metoda inspirovaná jednoduchou derivací obálky, je derivace klouzavého průměru. Algoritmus si udržuje dvě hodnoty klouzavých průměrů $\mathcal{A}$ a $\mathcal{B}$ délek A a B přes disjunktní intervaly, jak lze vidět na obr. 4.10b. Formálně je definujeme následovně:

$$\mathcal{A}(t) = \frac{1}{A} \sum_{i=t-A-B+1}^{t-B} x_i, \qquad \mathcal{B}(t) = \frac{1}{B} \sum_{i=t-B+1}^{t} x_i. \tag{4.16}$$

Detektor AverageDerivative detekuje vrchol tehdy, když $\mathcal{B}(t) - \mathcal{A}(t) \geq \theta$. Pokud se inspirujeme metodou dolních propustí, můžeme definovat alternativní verzi tohoto detektoru, jenž namísto operace rozdílu počítá poměr těchto dvou hodnot. Detektor AverageRatio pak tedy detekuje vrchol tehdy, když $\frac{\mathcal{B}(t)}{\mathcal{A}(t)} \geq \theta$. Pokud jeden z těchto detektorů rozpozná vrchol v čase t, může o něm informovat se zpožděním $d + M$, kde d značí zpoždění bližšího klouzavého průměru délky B.

4.3.3.4 Adaptivní práh daný mediánem

Předchozí metody adaptivní detekce vrcholu, jež jsou založené na výpočtu průměru úseku detekční funkce, mají jeden velký nedostatek. Pokud se v detekční funkci objeví neúměrně vysoké či nízké hodnoty odlehlé od ostatních, výrazně tím ovlivní hodnotu průměru. Způsob, jak se tomuto nedostatku vyhnout, spočívá v nahrazení průměru mediánem. Nechť $\widetilde{x}_{t,A}$ je medián z hodnot na intervalu $[t - A, t]$. Pak detektor MedianThreshold detekuje vrchol v čase t , pokud $x_t \geq \widetilde{x}_{t,A} + \theta$. Zpoždění této metody je dané velikostí okna pro medián A a je rovno $A + M$.

4.4 Shrnutí

V této kapitole jsme si představili druhý úkol nízkoúrovňové analýzy, kterým je detekce not. Nejprve jsme si problém definovali na základě obecného ADSR modelu a popsali jsme si s ním související pojmy. Poté jsme provedli analýzu vzájemné pozice not, která nám pomohla identifikovat náročnější části

Algoritmus	Parametry	Časová složitost
ENV$_{max}$	N, θ	$\mathcal{O}(N)$
ENV$_{rect}$	N, θ	$\mathcal{O}(N)$
ENV$_{sqr}$	N, θ	$\mathcal{O}(N)$
ENV$_{log}$	N, θ	$\mathcal{O}(N)$
EnZCR	N, θ	$\mathcal{O}(N)$
NLFER	N, θ, A, B	$\mathcal{O}(N \log N)$
SPECTVAR	$N, \theta,$	$\mathcal{O}(N \log N)$

Tabulka 4.1: Algoritmy detekce znělosti

problému. Dále jsme si popsali, s jakou přesností detekuje noty člověk, jako základní měřítko pro náš systém. Na základě této analýzy jsme navrhli schéma detekčního systému, jenž se sestává ze tří kroků: detekce znělosti, tvorby detekční funkce a volby vrcholů.

Poslední podkapitolu jsme věnovali představení algoritmů, jež řeší tyto tři fáze problému. Některé algoritmy jsou převzaté z odborné literatury, jiné jsou upraveny tak, aby byly použitelné pro naši aplikaci. Pro využití testovacího frameworku jsme u chování určitých přístupů provedli jisté úpravy či kombinace různých přístupů, které vedli ke vzniku algoritmů nových.

Algoritmy jsme rozdělili dle jednotlivých fází systému. Tab. 4.1 popisuje metody detekce znělosti, v tab. 4.2. je seznam detekčních funkcí a tab. 4.3 poskytuje seznam zmíněných metod volby maxima. Jelikož je volba hodnot parametrů algoritmů většinou závislá na testovacích datech, je dobrým pravidlem se snažit u návrhu obecných algoritmů počet parametrů co nejvíce redukovat. Proto jednotlivé parametry v tabulce zmiňujeme, coby další měřítko jejich využitelnosti pro obecné aplikace.

Algoritmus	Parametry	Časová složitost
ENV$_{max}$	N	$\mathcal{O}(N)$
ENV$_{rect}$	N	$\mathcal{O}(N)$
ENV$_{sqr}$	N	$\mathcal{O}(N)$
ENV$_{log}$	N	$\mathcal{O}(N)$
ENVDERIV	N	$\mathcal{O}(N)$
HFC	N	$\mathcal{O}(N \log N)$
SD$_{\ell_1}$	N,	$\mathcal{O}(N \log N)$
SD$_{\ell_2}$	N,	$\mathcal{O}(N \log N)$

Tabulka 4.2: Detekční funkce nástupů not

Algoritmus	Parametry	Zpoždění
AbsoluteThreshold	M	M
FilterRatio	M, L, S	$M + S$
FilterDifference	M, L, S	$M + S$
AverageDerivative	M, A, B	$M + B$
AverageRatio	M, A, B	$M + B$
MedianThreshold	M, A	$M + A$

Tabulka 4.3: Metody volby vrcholu detekční funkce

Kapitola 5

Detekce rytmu

Úkolem detekce rytmu je rozpoznání rytmické struktury v hudební skladbě. Naším cílem je vytvořit analyzátor, na jehož vstupu je sekvence not, které jsou určeny svým časem nástupu, délkou, výškou a prominencí. Tuto sekvenci jsme získali na základě detekce výšky, nástupů a délek not v nízkoúrovňové analýze. Výstupem je datová struktura, jež popisuje hudební reprezentaci díla, kterou jsme popsali v kap. 2.5. Rytmická struktura hudební skladby popisuje hierarchické uspořádání notových událostí v čase a její obecný popis naleznete v kap.1.3.2.1.

Detekci rytmu řadíme mezi vysokoúrovňovou analýzu hudební skladby. Ta oproti nízkoúrovňové analýze nemodeluje pouze principy lidského vnímání, ale snaží se modelovat i kognitivní procesy založené na hudebních znalostech posluchače. Proto jsou řešení tohoto problému často mnohem rozsáhlejší než řešení problémů nízkoúrovňové analýzy. Protože jde o mnohem specializovanější problém, který má méně obecného využití, není v odborné literatuře řešen tak často.

V této kapitole se nejprve podíváme podrobněji na rytmickou analýzu skladby a popíšeme si základní pojmy s ní související. V analýze problému pak budeme diskutovat, jak detekci rytmu modelovat v počítači. Podíváme se na obecné přístupy k tomuto problému a ukážeme si, jaké komplikace mohou při automatické detekci rytmu nastat. Poslední kapitola bude věnována návrhu rytmického analyzátoru.

5.1 Základní pojmy

Rytmus je dle Brownové (1992) obecný pojem pro popisování časově závislých vlastností hudby. Zejména popisuje pravidelně se opakující vzory hudebních událostí v závislosti na čase z hlediska jejich délek a prominencí. Vlastnosti těchto vzorů jsou důkladně popsány v hudební teorii a udávají rytmická pravidla, která většina hudebních skladeb v určité míře dodržuje.

Analýza rytmu Analýzou rytmu rozumíme pochopení rytmické role každé jednotlivé noty. Různé rytmické role lze uspořádat z hlediska důležitosti, která pak často koreluje s prominencí dané noty.

Doba je základní časová jednotka analýzy rytmu. Představuje nejvýraznější časový úsek, ve kterém se opakují prominentní hudební události. Člověk je většinou schopen délku jedné doby „vyklepat“ či se tomu naučit (Hainsworth, 2004, str. 18).

Rytmická úroveň Každou notu hudební skladby lze charakterizovat příslušností k jedné či více rytmickým úrovním. Tyto úrovně odpovídají hierarchickému uspořádání délek not v moderní hudební notaci a popsali jsme si je v kap. 1.3 (obr. 1.11). Jak bylo zmíněno, každá rytmická úroveň je reprezentovaná svou periodou, fází a typem. Tři základní rytmické úrovně se nazývají *tatum*, *taktus* a *takt* (Hainsworth, 2004, str. 19).

Tatum je rytmická úroveň nejkratší periody. Dle Klapuriho (str. 13) je tato perioda rovna „nejkratší vzdálenosti mezi nástupy not, která se objevuje častěji než náhodně“. Kvůli zarovnanosti rytmických úrovní pak tatum jednoznačně definuje množinu not, která je v rámci rytmické analýzy uvažována. Tzv. *ozdobné noty* jsou tedy z další analýzy vyloučeny. Dalším důsledkem zarovnanosti úrovní je ten, že perioda každé vyšší úrovně odpovídá celočíselnému násobku periody tatumu.

Taktus je hudební úroveň, jejíž perioda odpovídá jedné době. Označuje tedy nejvýraznější rytmickou úroveň. Když si člověk klepe do hudby, interval tohoto klepání nejčastěji odpovídá periodě takta.

Takt je druhá nejvýraznější rytmická úroveň nad taktusem. Výraznost taktu je většinou umocněna zarovnáním melodií a harmonických změn vůči taktu. Jeho volba je však též ovlivněna konvencemi notace (Hainsworth, 2004, str. 19)

Tempo skladby je převrácená hodnota délky jedné doby a udává se v jednotkách počtu dob za minutu (BPM).

Metrum je základní charakteristika rytmu hudební skladby a je jednoznačně určené poměrem period taktu a taktusu a typem těchto úrovní. Jeho reprezentaci v hudební notaci jsme si popsali v kap. 1.3.2.1.

Na analýzu rytmu lze tedy nahlížet jako na přiřazování not k jednotlivým rytmickým úrovním. To můžeme udělat sekvenčně od tatumu přes jeho rodičovské úrovně, taktus až po takt. Tatum nám jednoznačně určuje množinu not, které mohou vyšší úrovně využívat, ty jsou pak definované jejich fází a typem (viz kap. 1.3, obr. 1.11).

5.2 Analýza problému

Jelikož hudební notace výrazně reflektuje rytmickou strukturu hudebního díla (kap. 1.3), pro věrnou transkripci je zapotřebí její důkladná analýza. Detekce rytmických informací však není triviální. To lze demonstrovat i na pouhé detekci samotného taktusu. Rosenthal (1992) popisuje, že pokud zaznamenáme doby, při kterých si člověk klepe do rytmu určité skladby, zjistíme, že časy klepání nijak výrazně nekorelují s ničím zjevným v časovém průběhu hudebního signálu skladby. Pokud však porovnáme časy klepání přímým poslechem, připadají nám tyto časy v pořádku.

Shrňme si zde několik problémů, jež dělají hledání správného rytmického výkladu tak složitým:

Časové nepřesnosti Ačkoliv je rytmická struktura, jež je daná hudební notací, definovaná na exaktní mřížce, do které by měly být časy událostí zarovnané, pozice not se mohou od hran této mřížky ve skutečnosti výrazně lišit. Člověk při poslechu rytmu toleruje výrazné časové nepřesnosti tudíž i velmi nepřesně zahraná skladba může zachovat informace o rytmu. V některých případech může rozdíl vůči zarovnání mřížky činit až dvojnásobek periody dané úrovně (Rosenthal, 1992).

Variace tempa Jednou z hlavních vlastností hudby je její dynamická povaha. Možnost vybočovat ze zavedeného tempa (*ritardando*, tedy zpomalování a *accelerando*, neboli zrychlování) je oblíbený nástroj pro ovlivňování vyznění skladby (Klapuri, 2004). Proto se musíme na rytmickou mřížku dívat pouze jako na vodítko, které se dynamicky zarovnává k tempu hudby, než jako na pevně danou časovou strukturu skladby.

Neúplnost rytmické mřížky Ačkoliv je rytmus daný periodicitou a pulzováním, ukazuje se, že toto pulzování může být velmi neúplné. Neúplnost je zde myšlena tak, že některé doby, jež mají mít na základě periodicity důraz, ho mít vždy nemusí, a přesto dochází ke vjemu stejného rytmu (Rosenthal, 1992).

Problém určení prominence I když jsou některé noty vnímané výrazněji než jiné, často to není proto, že by byly hlasitější (Klapuri, 1997, str. 69). Prominence noty je komplexní atribut daný tvarem jeho harmonického průběhu a kontextu, ve kterém se nachází. Jelikož je pro nás těžké odhalit prominenci noty, přicházíme tak o výrazné vodítko pro správnou interpretaci rytmu.

Mnohoznačnost interpretace V průběhu skladby může být její rytmická interpretace často nejasná a je možné, že nastane situace, kdy bychom k části skladby našli více odlišných interpretací. K jejich upřesnění dojde většinou až v pozdějších částech skladby, kdy některé interpretace přestanou dávat smysl. To však komplikuje implementaci systému pracujícího v reálném čase. Jelikož musí skladbu analyzovat sekvenčně, musí zvažovat možnost libovolné interpretace, která dosud nebyla vyloučena.

Expresivnost hudby U řešení problému detekce rytmu pro obecné hudební signály se potýkáme s rozmanitostí hudebních signálů. Ačkoliv hudební teorie poskytuje určitý systém definic a pravidel, nelze je brát doslova a v hudebních signálech jsou často porušovány. Např. užití tzv. *hemioly*, která označuje užití tří dob v místě, které by mělo být běžně obsazeno dvěma dobami, je naprosto běžné (Rosenthal, 1992). Porušení rytmických pravidel sloučením více rytmů dohromady, jež se nazývá *polyrytmus*, je dalším příkladem obtížnosti obecného řešení detekce rytmu.

Abychom napodobili schopnosti člověka detekovat rytmus, budeme se pravděpodobně muset inspirovat způsobem lidské interpretace rytmu. Podle Rosenthala (1992) člověk při vnímání rytmu analyzuje velké množství hudebních a akustických podnětů, jako jsou barva tónu, harmonický kontext, melodické vzory a relativní čas mezi nástupy not. Na jejich základě pak volí takovou rytmickou interpretaci, která těmto podnětům odpovídá nejlépe. Navíc má člověk schopnost plynule upravit aktuální interpretaci tak, aby lépe odpovídala změnám v hudbě. Ukazuje se, že jakmile si člověk stanoví nějakou rytmickou interpretaci, je ochoten tolerovat i velké narušení jejích pravidel.

5.2.1 Možnosti řešení

Zde si analyzujeme několik běžně používaných přístupů k řešení detekce rytmu. Tyto možnosti byly použity na základě rešerše, ze které byly vybrány pouze metody, jež se zdály být pro naši aplikaci relevantní. Rozsáhlejší seznam detektorů rytmu lze nalézt v Klapuriho práci (Klapuri, 2004, kap. 2).

Obecně se detektory rytmu v literatuře dělí dle typu vstupu, se kterým pracují. Některé zpracovávají přímo akustický signál, zatímco druhé provádějí analýzu rytmu na úrovni symbolické reprezentace hudebních událostí, nejčastěji MIDI. Další dělení lze provést na základě rozsahu rytmické analýzy, kterou jsou schopny provést. Některé algoritmy totiž považují za detekci rytmu pouze rozpoznání jedné úrovně, nejčastěji taktusu. Jiné se pak omezují pouze na detekci metra.

5.2.1.1 Detekce metra a tempa pomocí autokorelace

Brownová (1992) ve své práci využívá základního faktu, že analýzu rytmu lze vyložit jako hledání periodicity na časové ose nástupů not. Tatum by mělo odpovídat základní periodě této časové osy, libovolné vyšší rytmické úrovně pak celočíselným násobkům periody tatumu. Jak jsme již popsali v kap. 3, periodu daného signálu lze velmi dobře nalézt za pomoci autokorelační funkce.

Časovou osu pro hledání periodicity x_t vytvořila autorka (z) MIDI vstupu následovně. Jelikož analýza metra oproti detekci výšky nevyžaduje příliš vysokou časovou přesnost, byla zde zvolena vzorkovací frekvence $f_s = 200$. Pro každý interval nástupu noty označila hodnoty x_t délkou této noty. Poté na tomto signálu spočetla autokorelační funkci, která je definována v kap. 3.5.2.3 v rovnici 3.37.

Pokud platí podmínka, že v rámci jednoho taktu o periodě T je nejčastější pozice noty na jeho počátku, což je pro západní hudbu běžné, bude mít funkce $\mathrm{ACF}_{\mathrm{A}}(\tau)$ ze signálu x_t maximum v bodě T. Počet nižších maxim nám navíc umožní detekovat metrum. Pokud je např. skladba v metru $\frac{3}{4}$, bude nejvyšší periodicita detekována v časech T, $\frac{T}{2}$ a $\frac{T}{3}$.

Další variantu použití autokorelace pro detekci metra a tempa popsal Gainza (2009). Ve své práci popisuje systém pro detekci metra přímo z akustického signálu. Umožňuje tak přeskočit krok detekce not, zde popsaný v kap. 4. Tento systém je složen z komplexního nástroje pro sledování rytmu, který je založen na autokorelační a rozdílové funkci (viz kap. 3). Pomocí těchto funkcí rozpozná časy jednotlivých dob a spočte matici podobnosti signálu každé dvojice dob. Na této matici spočte průchod minimální ceny, což je známý problém řešený dynamickým programováním. Tento výsledek pak použije pro vytvoření matice podobnosti dob. Diagonála, která bude mít maximální součet hodnot, bude odpovídat časovému posunu, ve kterém na sebe doby navazují nejlépe, a tento časový posun tak odpovídá periodě taktu.

Tyto metody nám umožňují velmi dobře detekovat tempo a metrum skladby, což jsou informace velmi důležité pro transkripci. Neposkytují nám však informace o tom, jakou rytmickou roli mají jednotlivé noty, jak je takt zarovnaný (zda se ve skladbě vyskytuje předtaktí), ani o dalších úrovních rytmu.

5.2.1.2 Systém preferenčních pravidel

Kromě výše zmíněných přístupů, jež jsou založeny spíše na matematickém výpočtu, byla snaha vytvořit systém detekce rytmu, který by více využíval znalostí z teorie hudby. Nejčastější přístup těchto „sémantických“ systémů spočívá v tzv. *preferenčních pravidlech*. Tato preferenční pravidla slouží jako kritéria pro zvolení nejlepší rytmické interpretace z dané množiny možných interpretací. Každá interpretace je dána zarovnáním jednotlivých rytmických úrovní na určité notové události. Pro nalezení nejlepší interpretace jsou všechny možné interpretace ohodnoceny pomocí daných preferenčních pravidel a je zvolena ta nejlepší. Jelikož počet možných interpretací je při uvažování rytmických nepravidelností až exponenciální, je pro tento výpočet využíváno dynamické programování, které výrazně sníží prohledávaný prostor.

Tato preferenční pravidla byla nejprve formulována slovně, jejich implementace do funkčních systémů přišla až později (Klapuri, 2004, str. 16). Temperley a Sleator (1999) ve své práci popisují jednu z nejrozsáhlejších implementací těchto preferenčních systémů. Vstupem jejich systému je symbolická reprezentace MIDI a provádí detekci neomezeného počtu rytmických úrovní. Jednotlivá preferenční pravidla zní např. „pravidlo pravidelnosti“, které preferuje, aby na každé úrovni byly doby maximálně rovnoměrně rozmístěny. „Pravidlo délky“ zase preferuje rytmické interpretace, které mají doby vyšších rytmických úrovní co nejdelší.

Ačkoliv dynamické programování u rytmické analýzy pomocí preferenčních systémů výrazně snižuje výpočetní složitost, paměťová složitost je pro jeho využití pro analýzu v reálném čase příliš vysoká. Jelikož musí mít algoritmus v libovolném čase kompletní reprezentaci nejlepší rytmické interpretace pro každé ohodnocení poslední hudební události, roste paměťová složitost exponenciálně s počtem různých ohodnocení.

5.2.1.3 Systém paprskového prohledávání

Systém, který spojuje výhody obou výše zmíněných přístupů, využívá k nalezení nejlepší rytmické interpretace tzv. *paprskové prohledávání*. Jedná se o uspořádané prohledávání stavového prostoru, které v každém kroku po expandování všech uzlů vyhodnotí neuzavřené uzly dle určité heuristiky. Pokud počet neuzavřených uzlů přesáhne určitou mez, jsou uzly dosahující dle heuristiky nejmenšího skóre označeny za neperspektivní a z dalšího prohledávání jsou vyřazené.

Formálně lze paprskové prohledávání popsat následovně: Nechť $G = \{U, H\}$ je graf, $\mathcal{S} \subseteq U$ množina startovních uzlů a zobrazení $h : U \to \mathbb{N}$ je heuristická funkce pro ohodnocení uzlů. Pak definujeme algoritmus pro nalezení nejlépe ohodnoceného uzlu, pro který v libovolném kroku expanze nepřesáhne počet otevřených uzlů číslo M, následovně:

1. Všechny uzly z množiny $\mathcal{S}$ vložte do fronty q.

2. Expanduj všechny uzly uložené ve frontě q a expandované uzly ulož do množiny T. Uzly, které nemají žádné nenavštívené sousedy, vlož do množiny R.

3. Odstraň všechny až na M nejlepších uzlů z množiny T, dle ohodnocení heuristickou funkcí h.

4. Vlož všechny uzly množiny T do fronty q, vyprázdni množinu T a pokračuj krokem 2.

Algoritmus skončí tehdy, když bude fronta v kroku 2 prázdná. Díky kroku 3 bude platit, že v kroku 2 expandujeme zároveň maximálně M uzlů.

Aplikace paprskového vyhledávání pro hledání rytmické interpretace je děláno pomocí *rytmických hypotéz*. Rytmická hypotéza odpovídá určité interpretaci rytmické struktury a je reprezentována datovou strukturou popsanou v kap. 2.5. Na začátku skladby vygenerujeme startovní hypotézy za pomocí metody

podobné autokorelačním systémům. Tyto hypotézy pak pro paprskové prohledávání představují startovní množinu uzlů. Expandování těchto uzlů odpovídá různým variantám interpretace nadcházejících hudebních událostí skladby. Tímto způsobem prohledáváme stavový prostor všech hypotéz, který za pomocí paprskového prohledávání prořezáváme v neperspektivních větvích.

Použití paprskového prohledávání v analýze rytmu bylo nejprve navrženo pouze pro detekci taktusu (Allen et al., 1990), Temperley (1999) však tuto aplikaci rozvedl na rozsáhlý kognitivní model vnímání rytmu zvaný „Machine Rhythm“. Tento systém si klade za cíl provést úplnou analýzu rytmické struktury skladby přiřazením rytmické role každé notě v hudební skladbě.

5.3 Návrh řešení

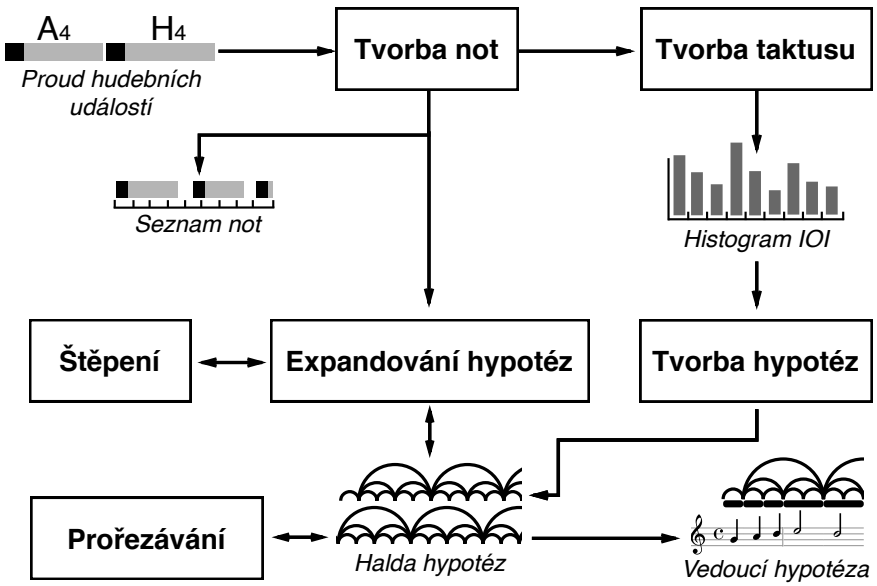
Jak popisuje Rosenthal (1992, str. 9), pro popsání tak složitého dynamického systému, jako je rytmická analýza, potřebujeme velmi rozsáhlé a precizní řešení. Oproti nízkoúrovňové analýze, kde jsme se snažili řešení co nejvíce zjednodušit a zobecnit, zde se zde nevyhneme detailnějšímu přístupu, který by popsal všechny aspekty rytmické analýzy.

Pro rozsah této práce bylo proto nerealistické porovnat výkonnost všech existujících analyzátorů rytmu. Proto jsme si zvolili ten, který jako jeden z mála dosahuje tak důkladné analýzy, aby na základě jejího výstupu bylo možné sestavit hudební notaci bez dalších heuristických pravidel a odhadů.

Zvolený systém *Machine Rhythm*, splňuje většinu našich požadavků. Jeho hlavní slabinou je, že není navržen pro práci v reálném čase. Proto jsme museli provést rozsáhlejší úpravy tohoto systému. Na konci své dizertační práce Rosenthal (1992, str. 101) popisuje další nedokonalosti systému a navrhuje určité úpravy pro novou verzi *Machine Rhythm 2*, které jsme se pokusili začlenit do našeho systému.

5.3.1 Schéma rytmického analyzátoru

V této části si popíšeme obecné schéma rytmického analyzátoru. Popíšeme si účel jeho jednotlivých komponent a datový tok mezi nimi.



Obrázek 5.1: Schéma systému pro analýzu rytmu

Schéma popisovaného systému lze vidět na obr. 5.1. Na vstupu je proud hudebních událostí, které představují nástupy a uvolnění not a údaje o jejich výšce a prominenci. Tento proud lze získat buď z nízkoúrovňové analýzy, kterou jsme popsali v předchozích kapitolách, nebo jej lze také obstarat z MIDI vstupu. Tyto informace sekvenčně vstupují do modulu *tvorby not*, který formuje objekty not a spravuje je v datové struktuře, která umožňuje efektivní přístup k *seznamu not* na časové ose. Vytvořené noty putují do modulu *tvorby taktusu*, který provádí detekci tempa podobně, jako je popsáno v kap. 5.2.1.1 za pomocí tzv. *histogramu IOI* (viz dále).

Informace o detekovaném tempu jsou pak poslány do modulu *tvorby hypotéz*, který na základě periody taktusu vytvoří různé startovní hypotézy rytmické interpretace. Tyto hypotézy jsou schraňovány v *haldě hypotéz*, tedy v struktuře napodobující funkci prioritní fronty, která slouží pro správnou funkci paprskového vyhledávání. Pokud počet hypotéz přesáhne stanovený limit, modul

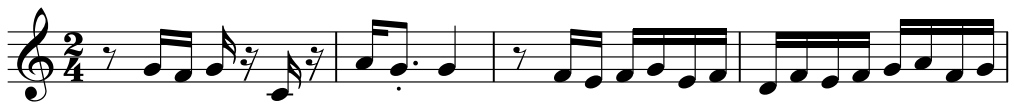
prořezávání se postará o to, aby byly neperspektivní hypotézy odstraněny. Zatímco vznikají nové hypotézy, vznikají i nové noty, které jsou posílány do modulu *expandování hypotéz*. Ten prochází haldu hypotéz a snaží se nové noty interpretovat za pomoci existujících hypotéz. Pokud je jejich interpretace víceznačná, je za pomoci modulu *štěpení* vytvořeno z dané hypotézy více hypotéz nových.

Činnost analyzátoru je rozdělena do dvou fází. Ve *startovní fázi*, kdy nemá dostatek dat pro provedení smysluplné analýzy, je rytmus reprezentován pouze v jedné rytmické úrovni. Po K přijatých událostech se analyzátor přepne do *analytické fáze*, ve které setrvá po celou dobu běhu. Jelikož je halda hypotéz neustále aktualizována, *vedoucí hypotéza*, jež je na vrcholu haldy, odpovídá aktuálně nejpřesnější rytmické interpretaci dle heuristik tohoto systému. Použijeme ji tedy pro vykreslení aktuální hudební notace. Protože jsou hypotézy reprezentovány datovou strukturou popsanou v kap. 2.5, bude přepis do hudební notace (až na detekci tóniny) triviální úkon. V následujících podkapitolách si popíšeme činnosti jednotlivých modulů.

5.3.2 Tvorba not

Do modulu tvorby not proudí informace o nástupech, uvolněních a výškách jednotlivých not. Jelikož informace o uvolnění noty může přijít po nástupu i se zpožděním několika sekund, abychom dosáhli požadované odezvy ne pozdější než půl sekundy, bude zapotřebí pracovat s notou ještě před jejím uvolněním. Proto jakmile modul obdrží informaci o nástupu noty, vytvoří novou notu a přidá ji do datové struktury seznamu not. Jakmile obdrží informaci o výšce a uvolnění noty, příslušné údaje na objektu noty upraví a případně o úpravě informuje závislé moduly.

Seznam not je datová struktura udržující informace o notách v chronologickém pořadí. Jelikož nejčastější operací bude přidání nové noty do seznamu, měla by být tato operace zpracována efektivně. Pro snadnou tvorbu hypotéz by tato struktura měla též efektivně zodpovídat intervalové dotazy. Ty po této



Obrázek 5.2: Začátek skladby „Preludium a fuga v C Dur, Dobře temperovaný klavír II, BWV 870“ od J. S. Bacha

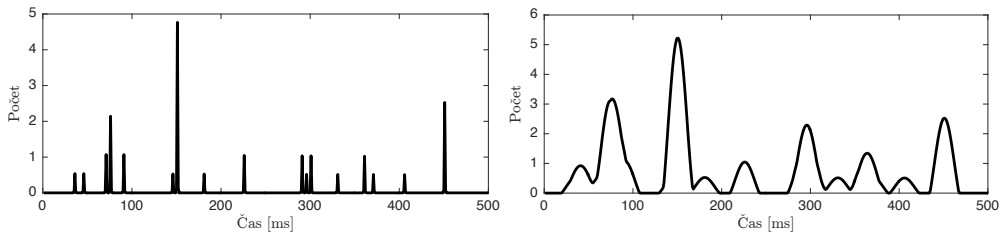
struktuře vyžadují seznam not na daném časovém intervalu. K logaritmické implementaci obou dotazů postačí libovolný automaticky vyvažovaný vyhledávací strom, např. *AVL*, *Treap* či *RB-tree*. Tyto stromy mají logaritmické vkládání a jelikož interval libovolného dotazu získáme pomocí dvou vyhledání not s časem spadajícím na hranice intervalu, je intervalový dotaz též logaritmický.

5.3.3 Tvorba taktusu

Úkolem tohoto modulu je aproximovat detekci tempa hudební skladby. Tempo, neboli frekvenci pulzování, lze detekovat podobně jako frekvenci výšky tónu pomocí autokorelační funkce (viz kap. 5.2.1.1). Rosenthal (1992) však navrhuje odlišný způsob, který se v principu autokorelačnímu způsobu velmi podobá, je však jednodušší pro implementaci v reálném čase. Využívá tzv. *histogram IOI*, kde IOI značí *inter-onset interval*, neboli časový interval mezi nástupy dvou tónů. Tento časový úsek je velmi důležitý pro vnímání rytmu. Brownová (1992) popisuje jeho důležitost na základě percepčních modelů a porovnává jeho důležitost s délkou not, která tak podstatná není. Intuitivně lze tento jev vysvětlit tím, že nástup noty bývá většinou mnohem výraznější než její uvolnění, a to především u nástrojů, které mají fázi útlumu noty obzvláště dlouhou (např. drnkací strunné nástroje).

Histogram IOI vytvoříme následujícím způsobem. Ve startovní fázi rytmického analyzátoru je vytvořen histogram vzdáleností mezi nástupy všech dvojic not. Pro danou vzdálenost τ je pak četnost jejího výskytu rovna $p(\tau)$. Harmonická četnost $p_h(\tau)$ je pak z histogramu vzdáleností vypočtena následovně:

$$p_h(\tau) = \sum_{i=1}^{Z} w_i p(i\tau), \tag{5.1}$$



(a) Histogram IOI **(b)** Vyhlazený histogram IOI

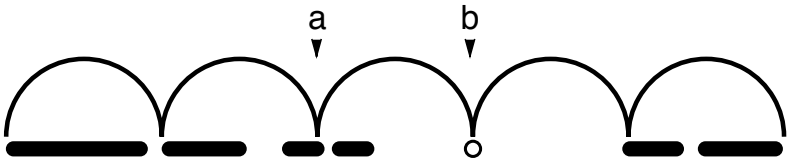
Obrázek 5.3: Histogram IOI úryvku skladby z obr. 5.2

kde Z je počet uvažovaných násobků dané délky a w_i je váha každého násobku. V původním programu bylo zvoleno $Z = 3$ a $w_1 = 1, w_2 = w_3 = 1/2$. Pokud máme např. tři noty vzdálené od sebe 50 ms, bude výpočet harmonické četnosti této vzdálenosti probíhat následovně: $p_h(50) = p(50) + \frac{1}{2}p(100) + \frac{1}{2}p(150) = 3 + \frac{2}{2} + \frac{1}{2} = 4\frac{1}{2}$. Na obr. 5.3a jsou vykresleny harmonické četnosti IOI ze začátku skladby „Preludium a fuga v C Dur“ od J. S. Bacha (viz obr. 5.2). Jelikož skladba na vstupu byla v tempu 100 BPM, doba čtvrťová odpovídá $\frac{60}{100}$ s. Protože jsou v této skladbě šestnáctinové noty nejvýraznější, je dle maxima histogramu doba periody taktusu odhadovaná na 150 ms. Jelikož výpočet harmonické četnosti spočívá v přesně zarovnaných vzdálenostech, aby tato metoda fungovala i při časových nepřesnostech zmiňované v kap. 5.2, rozostříme histogram Gaussovou křivkou šířky σ.

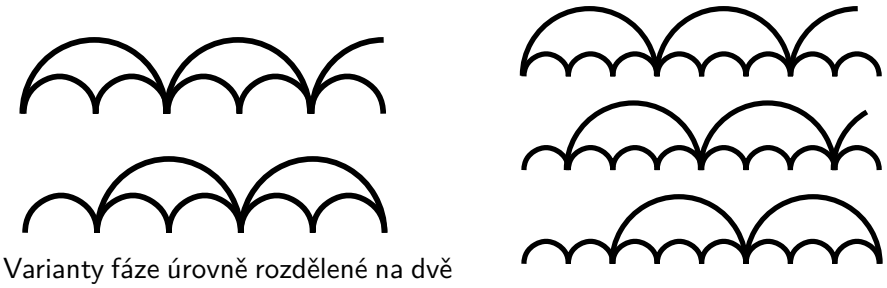
Finální volbu kandidátů pro periodu taktusu provedeme na konci startovní fáze tak, že zvolíme všechna lokální maxima nad prahem $c \cdot \arg\max p_h$, kde $c \in [0, 1]$. Tyto kandidáty zpracuje modul tvorby hypotéz. V průběhu analytické fáze je pak histogram aktualizován novými vzdálenostmi mezi nástupy sousedních tónů. Pokud aktualizovaný harmonický histogram přesáhne práh, noví kandidáti pro periodu rytmu jsou poslány do modulu tvorby hypotéz.

5.3.4 Tvorba hypotéz

Úkolem modulu tvorby hypotéz je vytvořit všechny smysluplné hypotézy rytmické interpretace pro dané tempo. To provede ve dvou krocích. Nejprve sestaví úroveň taktusu a pak k ní najde její rodičovské úrovně.



Obrázek 5.4: Ukázka problémů při tvorbě úrovně taktusu



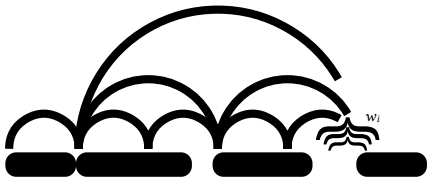
(a) Varianty fáze úrovně rozdělené na dvě doby

(b) Varianty fáze úrovně rozdělené na tři doby

Obrázek 5.5: Tvorba vyšších úrovní hypotézy

Tvorbu taktusu si lze představit jako speciální způsob prohledávání orientovaného grafu do hloubky, kde každý unikátní průchod reprezentuje nově vzniklou hypotézu. Ze všech not startovní fáze je spuštěn proces expanze úrovně, která se pro danou periodu této úrovně snaží sestavit takové propojení notových událostí, které se od periody taktusu liší co nejméně. Notu k propojení vybírá z časového intervalu, jehož volbu si blíže popíšeme v modulu expanze hypotéz. Někdy, (např. jako na obr. 5.4a) nelze jednoznačně zvolit notu k expanzi, a proto se hypotéza rozdělí na dvě. Pokud se ve vzdálenosti periody ani v jeho blízkosti žádný nástup noty nenachází, je vytvořena dočasná notová událost nazvaná *skrytá nota* 5.4b. Pokud se toto opakuje vícekrát za sebou, kontroluje se, aby počet po sobě jdoucích skrytých not nepřesáhl maximum sousedních skrytých not MaxNeighHiddenNotes.

Tvorba rodičovských úrovní je provedena vygenerováním všech možností rodičovských úrovní. Jak jsme popsali v kap. 1.3.2.1, hudební úrovně jsou zarovnané, a proto musí využívat pouze not používaných v úrovni pod sebou. Každá úroveň je definovaná svým typem a fází. Typ určuje, jak je relativně dlouhá jedna doba vůči své podúrovni. Fáze pak popisuje všechny varianty posunu této úrovně vůči té pod sebou. Jelikož pro většinu hudebních signálů



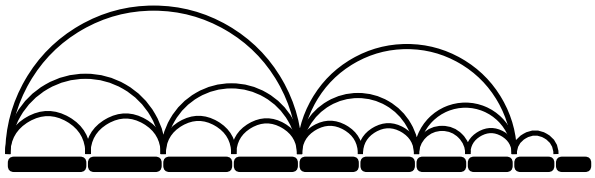
Obrázek 5.6: Průběh expanze hypotézy

postačí rozdělení úrovně na dvě a tři doby, omezíme se pouze na tyto dva typy. Všech pět variant vytvoření rodičovské úrovně lze vidět na obr. 5.5. Tímto způsobem vytváříme rekurzivně všechny možné úrovně až do maximálního počtu MaxNumOfLevels úrovní. Pokud náš program vytváří hypotézy až o k úrovních, vygeneruje se pro jeden taktus 5^k úrovní. Jelikož však postačí malý počet úrovní (pro většinu hudebních skladeb stačí čtyři), nebude tento počet nově vzniklých hypotéz ve fázi tvorby hypotéz pro náš program problém. Všechny nově vzniklé hypotézy pak vkládáme do haldy hypotéz.

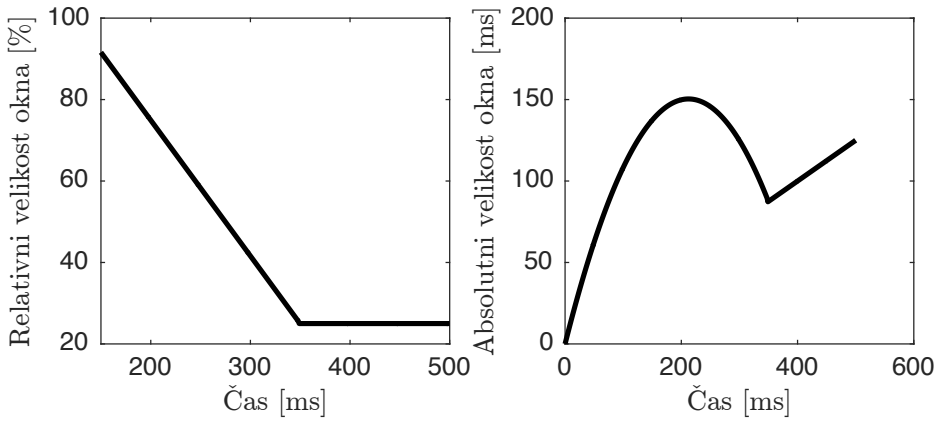
Informace o novém tempu do modulu tvorby hypotéz přichází z histogramu IOI, kde se může stát, že se určité tempo detekuje vícekrát. Proto je nutné kontrolovat, aby nevznikaly hypotézy, které již existují. Z hlediska tvorby hypotézy je každá vznikající hypotéza určena notou ze startovní fáze a jejím tempem. Proto si budeme udržovat počet aktivních hypotéz pro každé tempo. Jakmile celkový počet hypotéz daného tempa klesne na nulu (např. z důvodu prořezávání hypotéz) a z modulu tvorby taktusu dostaneme pokyn k tvorbě hypotéz tohoto tempa, můžeme jeho úroveň vytvořit.

5.3.5 Expanze hypotéz

Jak lze vidět na schématu 5.1, průběžně vznikající noty jsou souběžně posílány do dvou modulů. Zatímco napomáhají vzniku hypotéz nových, v modulu expanze hypotéz se pokoušíme smysl těchto nových not interpretovat z hlediska všech hypotéz v haldě hypotéz.



Obrázek 5.7: Ukázka přizpůsobování se hypotézy při změnám tempa



Obrázek 5.8: Velikost okna expanze

Při expanzi dané hypotézy pro danou notu děláme rozhodnutí, zda danou notu do hypotézy přijmout a na jaké úrovni. Na každé úrovni spočteme ideální pozici následující doby na základě průměru m posledních intervalů. Toto nám umožní upravovat rytmickou mřížku na základě změny tempa v hudební skladbě, jak je znázorněno např. na obr. 5.7, kde dochází ke zpomalování tempa, tzv. *ritardandu*. Zde je vidět, jak se takto délka periody taktusu snížila méně jak na polovinu.

Expanze dané hypotézy musí uvažovat všechny úrovně zároveň. Rosenthal (1992) popisuje, že na každé úrovni je dané jiné časové okno, ve kterém se následující nota může vyskytovat. Toto časové okno je definováno na základě faktu, že „posluchači tolerují větší relativní časové nepřesnosti u kratších intervalů (nižších úrovních)“. Konkrétní vztah je znázorněn na obr. 5.8, kde lze na levém grafu vidět klesající závislost relativní délky intervalu vůči délce intervalu. Pro intervaly delší než 350 ms pak již relativní délka zůstává neměnná. V absolutních délkách vidíme tento vztah na pravém grafu.

Štěpení hypotéz Jak lze vidět na grafu 5.8 vpravo, může nastat, že některá úroveň danou notu nepřijme, zatímco jiná ano. Proto je nutné zkusit expandovat notu na základě všech úrovní. Navíc musíme pro expanzi uvažovat možnost vložení skrytých not na ideální čas nejnižší úrovně a pokusit se expandovat stejným způsobem tuto novou hypotézu. Všechny varianty expanze jsou vyzkoušeny a jsou vloženy do haldy hypotéz. Ačkoliv tímto způsobem může vzniknout mnoho nesmyslných hypotéz, modul prořezávání hypotéz tyto nevhodné hypotézy odstraní.

5.3.6 Prořezávání hypotéz

Na základě výše zmíněného štěpení hypotéz a z důvodu neustálého přibývání nových hypotéz při detekci taktusu je pro výpočetní realizovatelnost nezbytné tento počet nějak redukovat. Jelikož je prohledávání stavového prostoru založené na principu paprskového prohledávání popsaného 5.2.1.3, tato redukce hypotéz bude fungovat na základě dané heuristiky h, která číselně ohodnotí vhodnost dané hypotézy z hlediska rytmické interpretace aktuálního úseku skladby.

Návrh této heuristiky je klíčovou částí celého rytmického analyzátoru. Zatímco předchozí moduly fungovaly spíše na základě hudebních pravidel rytmu, v tomto modulu je separován problém vnímání rytmu. Naším úkolem je rozpoznat, do jaké míry by člověk s danou rytmickou interpretací souhlasil při poslechu dané skladby. Jak jsme zmínili v kap. 5.2, člověk provádí analýzu rytmu na základě několika nezávislých podnětů, kterým věnuje větší či menší pozornost. Určité podněty mohou danou interpretaci potvrzovat, zatímco jiné se s ní zároveň mohou rozcházet.

Abychom se mohli věnovat jednotlivým podnětům zvlášť a případně na základě nových zjištění o vnímání rytmu snadno tyto závislosti do systému přidat, pokusíme se o co nejmodulárnější řešení. Každý podnět budeme vyhodnocovat nezávislou procedurou, tzv. *expertem*, který má za úkol číselně ohodnotit vhodnost této hypotézy pouze na základě jím analyzovaného podnětu. Od každého experta požadujeme, aby se věnoval pouze jednomu snadno popsatelnému a měřitelnému kriteriu, které je opodstatněné nějakým percepčním či hudebním faktem o vnímání rytmu.

Kombinace výsledků jednotlivých expertů provedeme na základě normalizace jejich výsledků pomocí průměru a směrodatné odchylky. Tyto normalizované hodnoty pak sloučíme na hodnotu zvanou *váha hypotézy* pomocí váženého součtu. Jednotlivé váhy expertů jsou parametrem programu, které budou pro výsledný program experimentálně zjištěny z testovacích dat. Pokud by byly všechny hypotézy z hlediska nějakého experta vyhodnoceny velmi podobně, normalizace pomocí směrodatné odchylky by vytvořila příliš velkou závislost i na velmi jemných rozdílech. Proto pro každého experta definujeme minimální směrodatnou odchylku, která určuje maximální detail rozdílů, které lze použít k porovnávání hypotéz.

V následujících podkapitolách si popíšeme jednotlivé experty, které jsme v našem analyzátoru implementovali.

5.3.6.1 Hustota událostí

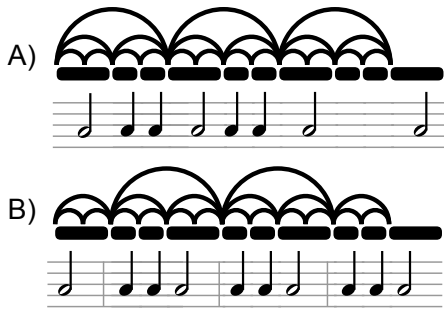
Fakt, že z hlediska vnímání rytmu by u správně hypotézy mělo na rytmicky důležité doby připadat více not, zejména těch percepčně prominentnějších, než na doby rytmicky méně důležité, se snaží využít expert, který vypočítává hustotu notových událostí. Ta je definována následovně:

$$\textsc{Density} = \sum_{e \in \mathcal{E}_h} \ell(e) r(e) / \mathrm{N}(h), \tag{5.2}$$

kde $\mathcal{E}_h$ je množina všech not použitých v hypotéze h, $\ell(e)$ je číslo nejvyšší úrovně, na které se nachází (číslované od taktusu odpovídajícímu jedné) a $r(e)$ je hlasitost dané noty. Normalizační člen $\mathrm{N}(h) = \sum_{e \in \mathcal{E}_h} \ell(e)$ je zde proto, abychom nezvýhodňovali hypotézy, které mají kratší periodu vyšších úrovní a na stejně velké množině not by automaticky získaly větší váhu.

5.3.6.2 Délka not událostí

Předchozí metrika popisovala prominenci hudebních událostí na základě jejich hlasitosti. Ukazuje se však, že dalším podstatným atributem prominence noty je její délka. Na základě práce Brownové (1992) namísto délky noty použijeme její vzdálenost od další noty, jelikož ta je vjemově podstatnější. Delších dob si přirozeně povšimneme snadněji než těch kratších. Proto definujeme metriku



Obrázek 5.9: Ukázka zdůraznění pomocí krátkých not

Duration takto:

$$\textsc{Duration}(h) = \sum_{e \in \mathcal{E}_h} \ell(e)\mathrm{IOI}(e)/\mathrm{N}(h), \tag{5.3}$$

kde $\mathrm{IOI}(e)$ je vzdálenost dané noty od noty následující, pro poslední notu pak její délka.

5.3.6.3 Zdůraznění pomocí sekvence krátkých not

Na základě faktu, že jsou noty vnímány důrazněji, pokud jsou předcházeny sekvencí kratších not (Rosenthal, 1992), můžeme zvýhodnit hypotézy, které mají výskyt takových situací. Příklad, využití této metriky je na obr. 5.9, kde hypotéza A získá výrazně vyšší ohodnocení, než hypotéza B, a to z důvodu zarovnání nejvyšší úrovně na noty půlové, kterým předchází dvě noty čtvrťové. Nota A je *kratší* než nota B právě tehdy, když je její délka $|A|$ menší než polovina její délky: $|A| < |B|/2$. Nechť $\textsc{pre}(e)$ je počet po sobě jdoucích not předcházející notě e, které jsou všechny kratší než nota e. Pak definujeme metriku hudební událostí ShortLong pro danou notu e následovně:

$$\textsc{ShortLong}(e) = \begin{cases} 0, & \text{kde } \textsc{pre}(e) = 0 \\ 1, & \text{kde } \textsc{pre}(e) = 1 \\ 2, & \text{kde } \textsc{pre}(e) \geq 2 \end{cases} \tag{5.4}$$

Na základě této vlastnosti každé noty vyhodnotíme metriku hypotézy ShortLongness jako:

$$\textsc{ShortLongness}(h) = \sum_{e \in \mathcal{E}_h} \ell(e)\textsc{ShortLong}(e)/\mathrm{N}(h), \tag{5.5}$$

5.3.6.4 Počet skrytých not

Pokud má nějaká hypotéza výrazně více skrytých not než hypotéza jiná, většinou to značí, že se snažíme hudebnímu modelu vycházet příliš vstříc tak, aby v něm daná rytmická interpretace byla možná. Proto zavedeme metriku Hiddenness následovně

$$\textsc{Hiddenness}(h) = \frac{\mathrm{HID}(\mathcal{E}_h)}{|\mathcal{E}_h|}, \tag{5.6}$$

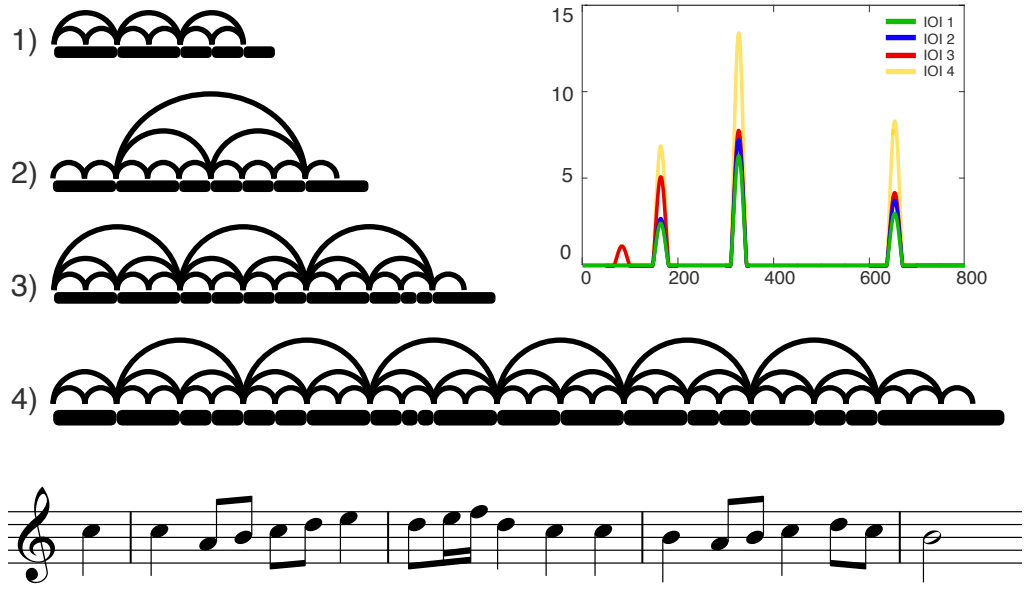
kde $\mathrm{HID}(\mathcal{E})$ je počet skrytých not v množině not $\mathcal{E}$.

5.3.6.5 Počet přeskočených not

Pokud některé noty nenáležejí žádným časovým oknům při expanzi hypotézy, může v našem analyzátoru snadno dojít k vynechání většího množství not. Tyto vynechané noty pak charakterizujeme jako *ozdobné* či trylky a nedáváme jim rytmický význam. Aby nedošlo k chybě této klasifikace, kde jsou jako ozdobné noty označeny i ty rytmicky důležité, je potřeba znevýhodnit hypotézy, které mají poměr ozdobných not vůči celkovému počtu not příliš velký. Na to použijeme metriku Skippiness:

$$\textsc{Skippiness}(h) = \frac{|\mathcal{E}_h|}{|\mathcal{E}|}, \tag{5.7}$$

kde $\mathcal{E}$ je množina všech not, jež analyzátor dosud přijal.



Obrázek 5.10: Znázornění průběhu analýzy rytmu na skladbě *Ach, lieben Christen, seid getrost* od J. S. Bacha (Bach, 1893b)

5.4 Shrnutí

V této kapitole jsme si nejprve vysvětlili základní pojmy z hudební teorie týkající se rytmu. Poté byla popsána problematika automatické detekce rytmu. U ní jsme provedli analýzu problémů, které při automatické detekci mohou nastat. Dále jsme zde popsali základní tři přístupy k detekci rytmu a tempa: autokorelační princip, systém preferenčních pravidel a systém paprskového vyhledávání. Na základě analýzy těchto přístupů jsme zvolili systém paprskového vyhledávání jako vhodný k použití v naší aplikaci. V poslední kapitole jsme popsali analyzátor rytmu založený na práci od Rosenthala (1992). V té bylo zapotřebí provést mnoho úprav, aby byl schopen provádět analýzu rytmu v reálném čase. Proto je zbytek kapitoly věnován popisu jednotlivých modulů tohoto systému.

Funkčnost tohoto analyzátoru znázorňujeme na obr. 5.10. Zde lze vidět vývoj analýzy začátku skladby *Ach, lieben Christen, seid getrost* od J. S. Bacha. Konkrétně zde vidíme vedoucí hypotézu v časech, kde došlo k její změně a ve skóre expertů převýšila nějakou jinou hypotézu. K nim je do grafu vložen histogram IOI v časech odpovídajících těmto změnám. První zobrazená hypotéza v čase 4

s již na základě prvních pěti not skladby správně odhadla čtvrťové a osminové noty. Druhá zobrazená hypotéza (5,2 s) zvolila správné předtaktí, typ druhé úrovně však neodpovídá skutečnosti. Ve třetí hypotéze (6,5 s) jsou již zvoleny správné typy úrovní, hypotéza s předtaktím však byla ve skóre předstižena touto hypotézou a to díky sedmé notě (E_2), která v tomto zarovnání zvyšuje skóre tím, že jí předcházejí krátké noty (metrika ShortLongness). Ve čtvrté hypotéze (11,5 s) je již rytmus správně interpretován a aktuální vedoucí hypotéza si udržela prvenství až do konce skladby.

Kapitola 6

Detekce tóniny

V tonální analýze nám podobně jako v analýze rytmu půjde o rozpoznání určité hudební struktury v hudební skladbě. Naším cílem bude detekovat, jakou roli hraje každá nota hudební skladby z hlediska tonality. Budeme tedy tvořit systém, který ze vstupu daných posloupností not, kde každá nota nese informaci o své délce, výšce a prominenci, rozdělí tuto posloupnost na segmenty stejné tóniny a zjistí, jaká tónina to je.

Tóninu lze definovat jako systém, pod kterým chápeme harmonické významy jednotlivých not (Temperley, 1999, str. 65). Tóninu, podobně jako u rytmus či výšku, vnímáme relativně vůči svému kontextu. Spíše než bychom tedy odvozovali tóninu konkrétní noty, musíme zjišťovat, jaký efekt má nota v rámci svého okolí.

Použitá tónina má velký význam ve vnímání hudební skladby. Potvrzují to kognitivní studie, které odhalily, že melodie se silným tonálním centrem se snáze zapamatují a zpětně rozpoznávají, a že i hudebně nevzdělaný člověk je citlivý ke změnám tóniny v průběhu skladby. Způsob, jakým člověk určuje tóninu skladby, nám není přesně znám. Byly však vytvořeny různé modely, které budou základem pro algoritmy, kterými se budeme snažit tóninu rozpoznat (detekovat). (Temperley, 1999, str. 66)

V této kapitole si nejprve vysvětlíme základní pojmy z hudební teorie, jež jsou nutné pro porozumění problému detekce tóniny. Pote se podíváme na komplikace, které při detekci mohou nastat a popíšeme si různé přístupy k řešení tohoto problému. V poslední kapitole si pak zvolíme jeden přístup, který bude vhodný pro naši aplikaci a důkladněji popíšeme jeho implementaci.

6.1 Základní pojmy

Tóninu jsme si definovali v kap. 1.2.1 jako označení sekvence not jako příslušící k nějaké stupnici na základě použitých not v tomto segmentu. Tato příslušnost je však definována na základě subjektivního sluchového porovnání jedince, a proto pro ní nemáme žádný exaktní vzorec. To výrazně komplikuje její detekci a podobně jako u ostatních problémů v této práci se ze též musíme inspirovat percepčními a hudebními modely.

Tonalita Tonalita je způsob organizace tónových výšek na základě hierarchického vztahu k jednomu centru (tzv. *tonální centrum*) (Beránková, 2017). Tyto vztahy se dějí jak na harmonické úrovni (v rámci souzvuku několika tónů, tzv. akordů), tak na úrovni melodické. Nás u monofonních skladeb budou zajímat pouze ty druhé.

Tonální centrum je jedna konkrétní výšková třída tónu, vůči které vztahujeme nějaký segment not. Je dáno použitými výškovými třídami v daném segmentu. Segment pevně určeného tonálního centra lze pak z hlediska tóniny považovat za stacionární.

Tonální hudba je kategorie hudby, která vzbuzuje dojem tonálního centra (Duckworth, 2012). Jelikož tonální centrum máme definované za použití klasických durových a molových stupnic, omezíme se při detekci pouze na ně.

Tonální analýza má za úkol rozpoznat tonální roli jednotlivých not vůči jejich tonálním centrům. Tuto roli charakterizujeme hudebním intervalem mezi výškovou třídou této noty a daným centrem. Nejdůležitější role se označují *tónika* (popisuje stejnou tonální třídu), *subdominanta* (vzdálenost sedmi půltónů) a *dominanta* (vzdálenost devíti půltónů). Tyto role jsou důležité, jelikož působí nejlibozvučněji, což zvyšuje dojem daného tonálního centra.

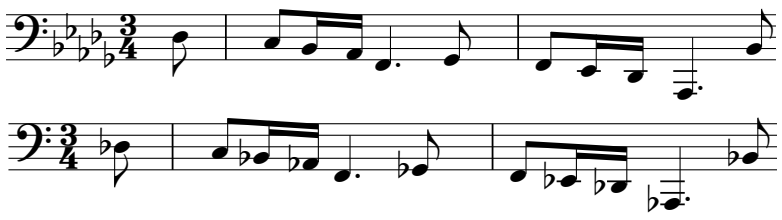
Modulace je krátkodobá změna tonálního centra. V hudebních skladbách je používána jako nástroj pro vytvoření napětí. Dělá hudbu zajímavější, jelikož vzbuzuje dojem konfliktu vyžadujícího rozuzlení, který tak budí očekávání. (Temperley, 1999).

Konsonance a disonance Hudební intervaly z hlediska libozvučnosti dělíme na konzonantní (libozvučné) a dizonantní (nelibozvučné). Pokud je tón dizonantní s aktuálním tonálním centrem, dochází k jeho vychýlení. Naopak pokud je t-on konzonantní, je toto centrum ještě více upevněno. Konzonance je proto důležitý aspekt ve vnímání tóniny a proto se jí budeme důkladněji věnovat v následující kapitole.

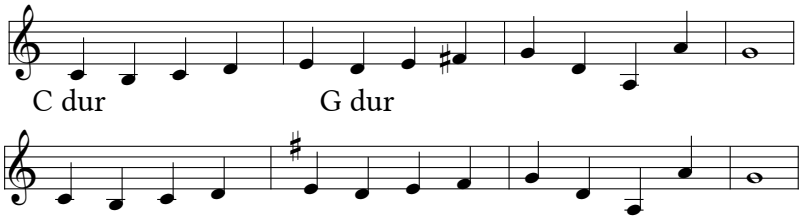
Chromatická stupnice je hudební stupnice, která obsahuje všech dvanáct výškových tříd temperovaného ladění uspořádaných od nejmenšího po největší ve vzdálenosti jednoho půltónu.

Diatonická stupnice je každá molová či durová stupnice popsaná v kap. 1.2.1. Každá se dá odvodit o své počáteční výškové třídy, proto jich je celkem $12 \cdot 2 = 24$.

Kadence je v hudební harmonii posloupnost akordů či not, kterou se běžně zakončuje hudební skladba. Tonálním centrem kadence bývá většinou tónika tonálního centra skladby, a tak by kadence poskytovala snadný přístup k detekci tóniny. Jelikož však v naší aplikaci vykreslujeme notaci v reálném čase, nemůžeme si dovolit na kadenci čekat. I člověk je při poslechu skladby schopen rozpoznat tóninu v průběhu skladby, předpokládáme tedy, že kadence není jediný způsob, jak tóninu zjistit.



Obrázek 6.1: Ukázka hudební notace s detekcí tóniny a bez ní

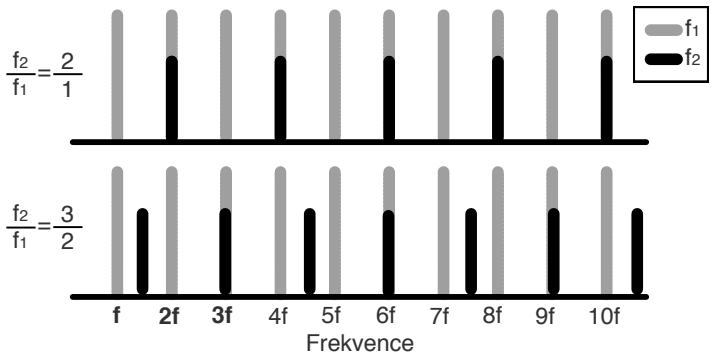


Obrázek 6.2: Ukázka modulace

6.2 Analýza problému

Ačkoliv je detekce tóniny z řešení hudební transkripce často vynechávána, o její důležitosti pro správnou hudební notaci hovořil již Moorer (1974, str. 113). Přestože by bylo možné tvrdit, že z hlediska naší definice hudební transkripce (kap. 1.1) není informace o tónině v notaci nutná, jelikož nám nepřidává žádné nové informace (zkušený hudebník dokáže tóninu odvodit na základě použitých not), je zde několik motivací tuto informaci přidat. Vynechání informace o tónině by v určitých situacích šlo označit z hlediska moderní hudební notace jako chybu v notaci, jelikož pokud tónina v předznamenání zmíněna není, implikuje to tóninu C dur či A mol. Navíc pokud zápis hudební skladby nereflektuje tóninu, ve které je skladba napsána, může být tato notace velmi matoucí pro interpreta skladby a ovlivnit tak jeho výkon. Ukázku tohoto jevu lze vidět na obr. 6.1, kde je začátek skladby v D^b mol ukázaný s předznamenáním a bez něj. V tomto případě to vede k tomu, že většinu not bylo nutné zapsat jako snížené, což výrazně znesnadňuje jejich čtení.

Jak jsme již zmínili, změny tonálního centra napříč skladbou jsou častým jevem. Kromě modulací, což jsou změny pouze dočasné a krátkodobé, které se často vrací zpět k tónině původní, může např. v polovině skladby dojít k trvalé změně tóniny v jinou. První změna je v notaci znázorněna pouze zvýšením



Obrázek 6.3: Ukázka souvislosti konzonance a zarovnání harmonických komponent

či snížením not z aktuální tóniny do modulované tóniny, druhý případ je pak znázorněn změnou předznamenání. Tyto dva typy jsou znázorněny na obr. 6.2. V implementaci detekci tóniny bude nutné zavést parametr citlivosti na tuto modulaci, který bude rozhodovat, jaká minimální doba modulace je zapotřebí, aby byl použit typ druhý. Tento parametr pak bude žádoucí nastavit jako dobrý kompromis mezi velmi častou změnou, která by byla rušivá, a příliš málo častou změnou, která by mohla mít za následek mnoho snížených či zvýšených not.

Konzonanci (resp. dizonanci) dvou tónů lze matematicky vysvětlit na základě zarovnání harmonických komponent tónu (viz definici harmonického tónu v kap. 3.1). Např. u hudebního intervalu oktávy je poměr frekvencí 2:1, a proto je každá druhá harmonická komponenta zarovnaná tak, jak je znázorněno na obr. 6.3 nahoře. Obecně se předpokládá, že čím více harmonických komponent je zarovnáno, tím spíše půjde o konzonantnější interval (Klapuri, 1997, str. 31). Jelikož neexistuje žádná kombinace harmonických tónů, které by byly komponenty zarovnané častěji než každá n-tá (kde n je přirozené číslo), jde o nejlibozvučnější tonální roli. Konzonance harmonických tónů, které mají roli tzv. *tóniky*, je tak výrazná, že ji nazýváme oktávovou ekvivalencí.

Obecně platí, že pokud je frekvence dvou tónů vzájemně v poměru, který lze napsat jako celočíselný zlomek, tedy pro tón o frekvenci f_1 lze druhý tón zapsat jako $f_2 = \frac{m}{n} \cdot f_1$, tak, že $m, n \in \mathbb{N}$, jde o konzonantní intervaly (Klapuri, 1997, str. 32). Čím je maximum těchto dvou čísel menší, tím konzonantnější intervaly to jsou (fakt známý již Pythagorovi, viz kap. 2.2). To je dáno tím, že každá m-tá

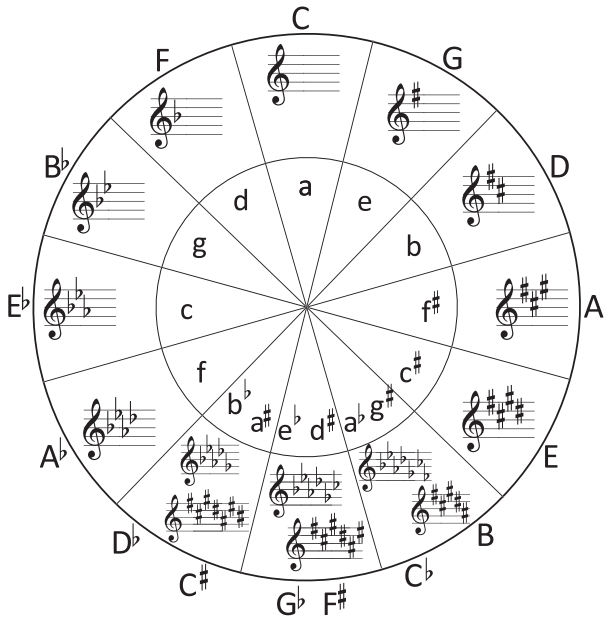
harmonická komponenta tónu f_1 je zarovnaná na každou n-tou harmonickou komponentu tónu f_2. Tento fakt lze odvodit následovně. Hledáme $i, j \in \mathbb{N}$ takové, že $i \cdot f_1 = j \cdot f_2$:

$$\begin{aligned} if_1 &= jf_2 \\ \frac{i}{j} f_1 &= \frac{m}{n} f_1 \\ \frac{i}{j} &= \frac{m}{n} \end{aligned} \tag{6.1}$$

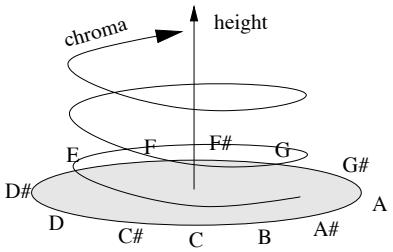
Ukázali jsme tedy, že i a j musí být v poměru $\frac{m}{n}$, což dokazuje předchozí tvrzení. Znázornění tohoto vztahu lze vidět na obr. 6.3. Na horním grafu jsou vidět frekvence v poměru $\frac{2}{1}$ a proto je zarovnaná každá druhá harmonická komponenta tónu f_1 na každou první harmonickou komponentu tónu f_2. Na dolním grafu je pak znázorněn poměr $\frac{3}{2}$ kde každá třetí komponenta tónu f_1 je zarovnaná na každou druhou harmonickou komponentu tónu f_2.

Na základě těchto vztahů lze odvodit základní role not vůči tonálnímu centru, jež jsou z hlediska tóniny považovány nejdůležitější. Kromě výše zmíněné tóniky (interval oktávy, 2:1) jsou dalšími důležitými pozicemi dominanta (interval *kvinty* 3:2) a subdominanta (interval *kvarty* 4:3). Při temperovaném ladění tyto poměry přesně dodržované nejsou, protože půltóny mají iracionální poměr $\sqrt[12]{2}$ (viz kap. 1.2.1). Jelikož je však člověk schopen tyto malé odchylky tolerovat, výše zmíněný efekt zarovnání je stále stěžejní pro vnímání hudby (Klapuri, 1997).

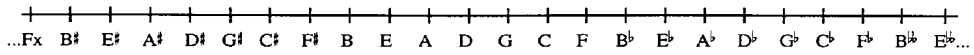
Pro snazší orientaci v těchto nelineárních vztazích podobnosti tónů bylo navrženo několik konceptů, které pomáhají převést podobnost výšek tónů na metriku vzdálenosti a vykreslit ji do prostoru. Asi nejznámější pomůckou k tomuto účelu je kvintová kružnice, kterou lze vidět na obr. 6.4. Vzdálenost znázorněných not odpovídá vnímané vzdálenosti z hlediska poměru jejich frekvencí. Po směru hodinových ručiček se nachází noty v intervalu kvinty, proti směru pak v intervalu kvarty. Neintuitivní vlastnost, že stejná vzdálenost na vizualizaci může být ve skutečnosti výškově rozdílná v závislosti na jejím směru, vychází z oktávové ekvivalence. Na této vizualizaci můžeme vidět znázorněny všechny durové a molové stupnice s křížky i s béčky, jejichž stavba je postavena na výše zmíněných vztazích.



Obrázek 6.4: Kvintová kružnice (převzato z Linkware-Graphics, 2015)

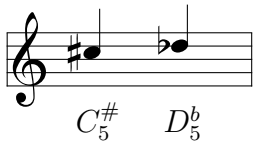


Obrázek 6.5: Šroubovice výšky (převzato z Walmsley, 2001, str. 38)



Obrázek 6.6: Kvintová přímka (převzato z Temperley, 2000, str. 290)

Na obr. 6.5 je další vizualizace vjemu výšky tónu, tentokrát v trojrozměrném prostoru. Zde je znázorněna závislost oktávy a výškové třídy tónu (neboli chroma). Poměry euklidovské vzdálenost mezi dvěma notami by měly odpovídat vnímaní výšky. Lze vidět, že každá úroveň šroubovice je tvořena kvintovou kružnicí.



Obrázek 6.7: Ukázka problému volby jména not

Kvintová přímka na obr. 6.6 je promítnutí kvintového kruhu do nekonečné přímky. Motivací k této reprezentaci je, že pojem výškové třídy tónu je absolutní, nezahrnuje tedy fakt, že výška tónu je vnímaná relativně a tedy ten samý tón může mít ve dvou různých kontextech úplně jiné vyznění (Temperley, 2000, str. 289). Každý tón má na kvintové přímce nekonečno různých pozic, kde každá pozice odpovídá jinému kontextu, ve kterém je použit. Konkrétně jde o tóny, které z hlediska výškové třídy vnímáme jako totožné, přesto mají více typů zápisu (např. $C_5^{\#}$ a $D_5^{\flat}$, viz obr. 6.7). Volba zápisu těchto not ovlivňuje jejich příslušnost k naprosto odlišnému tonálnímu centru. Proto jsou tyto noty na kvintové přímce umístěny daleko od sebe, přestože na kvintové kružnici jde o stejné pozice.

6.3 Algoritmy detekce tóniny

Pokud bychom řešili jednodušší problém detekce tóniny na hudební skladbě, která neobsahuje žádné modulace, intuitivní přístup by byl následující. Zjistili bychom jaké výšky tónů se ve skladbě vyskytují, a porovnali bychom je s každou diatonickou stupnicí. Pokud skladba využívá tóny pouze jedné tóniny, bude množina tónů této skladby podmnožinou množiny tónů dané stupnice, což nám umožní ji detekovat. Pochopitelně však nemusí využívat všechny její tóny, a proto tato detekce nemusí být jednoznačná.

Přestože je tento přístup naivní a na obecné skladby by nefungoval, je základem pro většinu známých algoritmů detekce výšky. Hlavní vadou tohoto přístupu je, že jakmile dojde ve skladbě k modulaci, může jednoduše nastat, že použité tóny nejsou podmnožinou žádné stupnice. Nyní si ukažme, jak různé algoritmy tento problém řeší.

6.3.1 Longuet-Higginsův a Steedmanův algoritmus (LHSA)

Jedním z nejstarších principů k automatické detekci tóniny je přístup Longuet-Higginse a Steedmana. Výše zmíněný problém je zde řešen předpokladem, že začátek skladby neobsahuje žádné modulace. Algoritmus je pak postupně eliminuje na základě ve skladbě, dokud nezbude pouze jedna nebo algoritmus nedosáhne konce skladby. Formálně je tento postup definován následovně:

1. Nechť $\mathcal{T}$ je množina všech 24 diatonických stupnic.

2. Nechť p je výšková třída aktuální noty na vstupu.

3. Eliminujme všechny tóniny z množiny $\mathcal{T}$, které neobsahují p. Pokud ji neobsahuje žádná, aniž bychom cokoliv odstraňovali, označíme celou množinu $\mathcal{T}$ jako výstup a ukončíme algoritmus.

4. Pokud $|\mathcal{T}| = 1$, označme tuto tóninu jako výstup a ukončíme algoritmus.

5. Posuneme se o notu dále a pokračujme krokem 2.

Jak ukazuje testování tohoto algoritmu, pro mnohé hudební skladby předpoklad vstupu platí a algoritmus opravdu zvolí správnou tóninu. Hlavní problém spočívá v tom, že průběh algoritmu většinou skončí v kroku 3, kdy kromě správné tóniny zvolí i několik jiných. Protože algoritmus nemá žádné prostředky k tomu, aby zjistil, která tónina je z těchto zbylých tou nejlepší, bylo by nutné ji volit náhodně. Jedním z přínosů tohoto algoritmu je dle Temperleye (1999, str. 74) princip prvenství: fakt, že tóny z počátku skladby ovlivňují její globální tonální vyznění výrazněji, než ty později. Pro více modulované skladby však toto zjištění nelze aplikovat.

Výše zmíněný algoritmus lze implementovat i za pomocí skalárního součinu vektorů. Každou tóninu budeme reprezentovat bitovým vektorem délky 12, kde každý bit odpovídá informaci, zda je jemu odpovídající výšková třída v tónině obsažena. Nazvěme si takový vektor *klíčový profil*. Pokud budeme číslovat výškové třídy od noty C, klíčový profil tóniny C dur vypadá následovně: $\mathrm{KP}_{(\text{C dur})} = [1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1]$. Pokud stejným způsobem spočteme vstupní výškový profil VP, počet společných not bude dán skalárním součinem

těchto vektorů:

$$\mathrm{KP}_{(\text{C dur})} \cdot \mathrm{VP} = \sum_{i=1}^{12} \mathrm{KP}_{(\text{C dur})_i} \mathrm{VP}_i \tag{6.2}$$

Označme si tuto notaci jako BinaryIn–BinaryKey. Na tento princip navážeme v následujícím algoritmu.

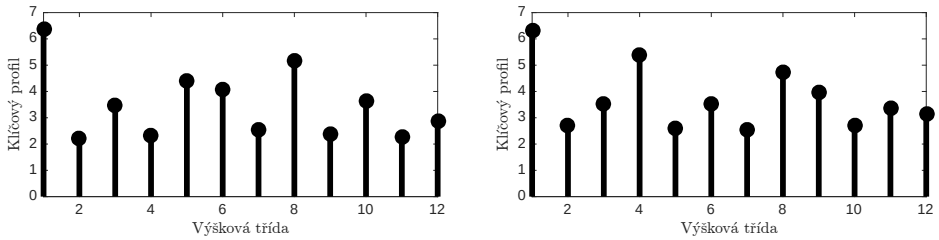
Temperley (1999) představuje úpravu algoritmu LHSA, která má podporovat modulaci. Vstupní sekvence not je rozdělena na několik segmentů, kde každý má přesně daný počet not (malá konstanta, např. 5). Na každém segmentu je pak nezávisle vyhodnocen LHSA. Volba velikosti segmentu zde však má příliš velký vliv na výsledek a je závislá na typu rytmu.

6.3.2 Huronův a Parncuttův algoritmus (HPA)

Další algoritmus pro detekci tóniny využívá tzv. exponenciální pokles . Při něm je pro detekování tóniny v libovolném čase používáno též porovnání not skladby s množinou not každé tóniny. Jednotlivé noty na vstupu jsou však váženy dle jejich aktuálnosti a tato váha s časem exponenciálně klesá. Konkrétně pokud vyhodnocujeme algoritmus v čase t, pak váha noty v čase a bude rovna $2^{-\frac{t-a}{\tau}}$, kde τ je parametr programu. Tato váha byla vynásobena normalizovanou prominencí noty z intervalu. (Huron et al., 1993)

Tento algoritmus by šel snadno upravit tak, aby se více podobal algoritmu v předchozí kapitole. Stačí, abychom namísto binárního vektoru vstupu použili vektor reálných čísel, kde hodnota pro každou výšku bude součet všech vah not této výšky na vstupu. Tento přístup nazveme WeightedIn–BinaryKey.

Modulaci bychom zde mohli řešit tak, že by byl algoritmus puštěn opakovaně v daných intervalech. Intervaly zvolíme stejně jako jsme volili segmenty, tedy v intervalu konstantního počtu not. Protože v každém kroku algoritmus provádí korelaci s T segmenty o výškových profilech délky M a protože hodnota předchozích segmentů se liší pouze v časovém posunu, který se na výpočtu projeví vynásobením této hodnoty výše zmíněnou rovnicí, bude časová složitost $\mathcal{O}(N \cdot T \cdot M)$.



(a) Klíčový profil durových stupnic **(b)** Klíčový profil molových stupnic

Obrázek 6.8: Klíčový profil pro KSA

6.3.3 Krumhansl-Schmucklerův algoritmus (KSA)

Jelikož skladby, které mají tonální centrum, obvykle modulace obsahují, Krumhanslová a Schmuckler přišli s následující myšlenkou. Když zjišťujeme, jak vhodná je daná tónina pro určitý úsek skladby, mohli bychom namísto binárního vážení na základě not dané tóniny použít pravděpodobnosti výskytu dané výšky v tónině. Jak jsme si ukázali v předchozí kapitole, některé tóny jsou od daného tonálního centra vzdálenější než jiné, a proto má smysl vyjádřit příslušnost dané výšky k tónině vyjádřit jinak než binárním číslem.

Získání těchto hodnot bylo provedeno v souladu s naší definicí tóniny na základě subjektivního sluchového porovnávání. Tento experiment probíhal následovně. Nejprve byla posluchači zahrána pasáž, která plně odpovídala nějaké tónině. Poté byl zahrán testovaný tón a posluchač měl ohodnotit, jak dobře tento tón pasuje k tonálnímu centru dané stupnice. Výsledky tohoto experimentu lze vidět ve výsledném klíčovém profilu KP na obr. 6.9.

Vstupní výškový profil byl v algoritmu KSA vážen podle délek not tak, že každá nota přidá k výškovému profilu VP pro danou výšku hodnotu své délky v sekundách. Aby bylo dosaženo normalizace obou metrik, byl namísto skalárního součinu popsaného v algoritmu LHSA použit vztah pro korelaci vektorů délky M (počet not v tónině):

$$\mathrm{KSA}(key) = \frac{\sum_{i=1}^{M}(\mathrm{KP}_{key_i} - \overline{\mathrm{KP}_{key}})(\mathrm{VP}_i - \overline{\mathrm{VP}})}{\sqrt{\sum_{i=1}^{M}(\mathrm{KP}_{key_i} - \overline{\mathrm{KP}_{key}})^2 \sum_{i=1}^{M}(\mathrm{VP}_i - \overline{\mathrm{VP}})^2}}, \tag{6.3}$$

kde KP_{key} je vektor vážených klíčových profilů a VP je vážený výškový profil vstupu.

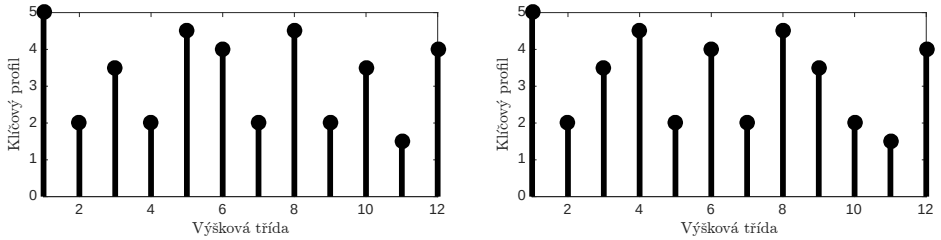
Temperley (1999, str. 78) zde kritizuje zejména globální přístup k detekci tóniny, a přímo tvrdí, že „použití vážených klíčových profilů k detekci globálního tonálního centra je chybné“. Proto i pro tento algoritmus můžeme vstup rozdělit na segmenty a vyhodnotit tyto segmenty algoritmem zvlášť. Pro tóninu každého segmentu zvolíme tu, která se vstupním výškovým profilem koreluje nejlépe. Protože v tomto případě používáme vážený vstup i klíčový profil, budeme tento přístup nazývat WeightedIn–WeightedKey. Jelikož pro každý segment musíme provést korelaci s každou tóninou, je časová složitost algoritmu rovna $\mathcal{O}(N \cdot T \cdot M)$.

6.3.4 Temperleyův upravený KSA (TKSA)

Temperley ve své práci (1999) provádí několik výrazných úprav algoritmu KSA. Na základě experimentů zjišťuje, že použití započítávání všech not segmentu do výškového profilu není vhodné řešení. Ukazuje to na několika protipříkladech, kdy se v segmentu vyskytuje opakovaně nota, která není tónikou. Ačkoliv tento segment není vůbec modulovaný, algoritmus se opakováním stejné noty čím dál více vzdaluje své tónině ve prospěch tóniny opakované noty.

Proto navrhuje poslední variantu použití výškových a klíčových profilů a tou je BinaryIn–WeightedKey, tedy vstup daný binárním vektorem. Pokud jsou segmenty dostatečně krátké, vede tato úprava na opravu výše zmíněného nedostatku, aniž by přitom celkový výsledek nějak zhoršil.

Dále ve své práci navrhuje úpravu klíčových profilů (KP) na základě znalostí z hudební teorie a experimentálních pokusů. Tyto nové profily jsou vidět na obr. 6.9. Hlavními změnami bylo zvýšení váhy pro diatonické stupně, obzvláště pak pro tonálně důležité pozice, a celková normalizace hodnot. Díky té již není nutné vypočítávat korelaci z rovnice 6.3, jelikož ani normalizace, ani centrování průměrem se neprojeví na poměrech hodnot napříč tóninami, které jsou jediné podstatné pro volbu maxima.



(a) Klíčový profil durových stupnic **(b)** Klíčový profil molových stupnic

Obrázek 6.9: Temperleyův upravený klíčový profil pro TKSA

Další úprava spočívá ve vyhodnocování modulace. Ačkoliv výše zmiňované úpravy algoritmů detekci modulovaných úseků do jisté míry podporovaly, měly jednu zásadní vadu. Lidské vnímání tóniny, podobně jako je tomu u rytmu či u výšky, má tendenci tolerovat nepřesnosti vůči již zavedenému výkladu okolností a setrvávat ve stejném stavu. Aby algoritmus detekce tóniny tento jev modeloval, je zaveden parametr *penalty* P, která je započítána při změně tóniny mezi dvěma sousedními segmenty. Algoritmus pak hledá takové ohodnocení segmentů tóninami, aby výsledná hodnota i se započtením penalt byla co nejvyšší. Formálně je hodnota nejlepší kombinace ohodnocení prvních n segmentů definována následovně:

$$\mathrm{TKSA}(n, key) = \begin{cases} \mathrm{VP}_n \cdot \mathrm{KP}_{key} + \max\limits_{k \in \mathcal{T}} \Big(\mathrm{TKSA}(n-1, k) & \\ \qquad\qquad - P \cdot \mathbf{1}_{\mathcal{T} \setminus key}(k) \Big) & n > 0 \\ 0 & n = 0 \end{cases} \tag{6.4}$$

kde VP_n je výškový profil segmentu, KP_{key} je klíčový profil tóniny key a $\mathcal{T}$ je množina všech tónin. Nejlepší dosažitelná hodnota pro N segmentů je tedy rovna:

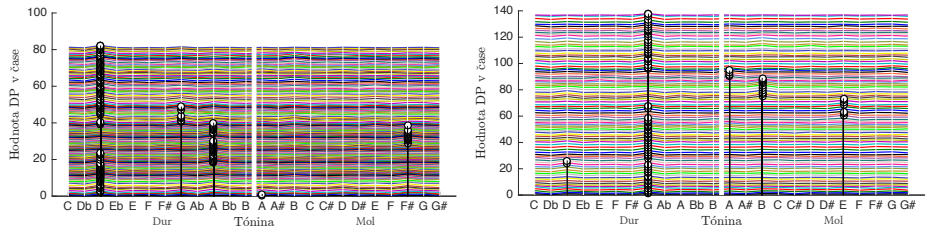
$$\mathrm{TKSA}(N) = \max_{k \in \mathcal{T}} \mathrm{TKSA}(N, k) \tag{6.5}$$

Abychom zjistili přes jaké tóniny lze této nejlepší hodnoty dosáhnout, stačí následovat výpočet až do $n = 1$ a v každém n-tém segmentu zvolit takovou tóninu k, pro kterou je hodnota výpočtu nejvyšší.

Tuto rovnici můžeme efektivně vypočíst za pomoci dynamického programování. Pro počet tónin T, počet tónů v segmentu M a počet segmentů N máme $N \cdot T$ různých stavů a v každém stavu provádíme $M + T$ operací (korelace a volba maximálního předchůdce), proto je výsledná časová složitost rovna $\mathcal{O}(N \cdot T(M + T))$. Pro rekonstrukci nejlepšího ohodnocení segmentů potřebujeme pro každý segment znát jeho nejlepší předcházející tóninu pro každé ohodnocení tohoto segmentu. Paměťová složitost algoritmu je proto rovna $\mathcal{O}(N \cdot T)$.

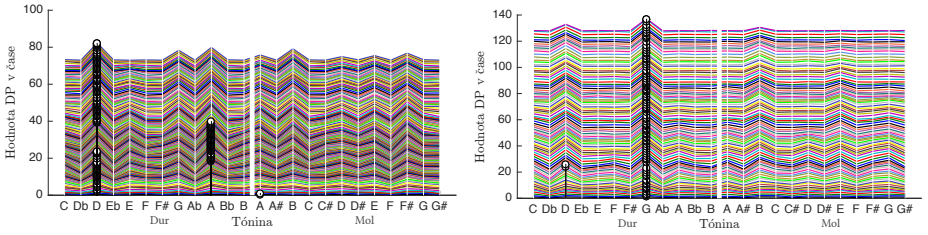
Pro snadnější pochopení je zde znázorněn úsek běhu algoritmu na dvou preludiách z Bachova *Dobře temperovaného klavíru* (1685). Na obr. 6.10 jsou vidět hodnoty funkce TKSA(n, key) pro všech 24 tónin v každém segmentu shora dolů od nejstarších po nejnovější. Horní obrázky odpovídají nastavení $P = 1$, dolní pak $P = 9$. Pro každý segment je znázorněna maximální hodnota Jak lze vidět, parametr penalty má opravdu vliv na frekvenci modulací. Na horních obrázcích lze vidět, že příliš nízká penalta modulace způsobuje velké nežádoucí skoky v tóninách. Zejména na spodních obrázcích pak můžeme vidět, že všechny vrcholy odpovídají tóninám, jež si jsou navzájem podobné (v durových *D*, *G* a *A* a v molových *B*, *E* a $F^{\#}$) jsou vedle sebe na kvintové kružnici, viz 6.4. Tato podobnost způsobila i dočasné překlenutí ze stupnice *D dur* do *A dur* na levém spodním obrázku.

Tato úprava zásadně odlišuje algoritmus TKSA od ostatních algoritmů. Zatímco výše zmíněné algoritmy bychom mohli nazvat „procedurální“, jelikož udávají přesný proces, jak detekci provést, na tento algoritmus lze nahlížet jako na „preferenční systém“, jelikož, podobně jako u analyzátorů rytmu, řeší problém popisem požadovaného výsledku. U percepčních modelů je tento přístup velmi častý, jelikož řešíme návrh algoritmu na základě známého chování systému. Jak zmiňuje Temperley (Temperley, 2002), preferenční systémy lze často modelovat jako Bayesovské modely, což lze vnímat jako další motivaci pro jejich správnost. Jelikož nevíme, jakým způsobem u člověka vnímání funguje, použít algoritmus, který se bude chovat podle požadovaného výsledku, se zdá být rozumné. Jelikož však tyto systémy nemají příliš velkou znalost o problému, musí vyzkoušet všechny varianty jeho chování, což je výpočetně mnohem náročnější.



(a) Průběh TKSA na preludiu v *D dur* (BWV 874)

(b) Průběh TKSA na preludiu v *G mol* (BWV 884)



(c) Průběh TKSA na preludiu v *D dur* (BWV 874)

(d) Průběh TKSA na preludiu v *G mol* (BWV 884)

Obrázek 6.10: Ukázka běhu algoritmu TKSA na preludiu v *D dur* (BWV 874, vlevo) a preludiu v *G mol* (BWV 884, vpravo)

6.4 Shrnutí

V této kapitole jsme si nejprve obecně popsali problém detekce tóniny a odůvodnili její význam. Poté jsme si popsali základní pojmy z hudební teorie, které jsou nutné pro pochopení způsobu řešení tohoto problému. Vysvětlili jsme si především tonální centrum a modulaci tóniny. V následující podkapitole jsme pak analyzovali detailnější aspekty tonálního centra a odůvodnili si jeho chování. Dále jsme si představili si několik reprezentací, které usnadňují jeho pochopení.

V poslední podkapitole jsme popsali několik algoritmů, které slouží k detekci tóniny. Na tyto algoritmy lze též pohlížet jako na percepční modely vnímání tóniny, jelikož se snaží napodobit lidský vjem tonality. Jelikož Temperley (1999) již provedl testování popsaných algoritmů a na jejich základě provedl mnohé úpravy a protože se nám již nepodařilo nalézt algoritmus, který by řešil tento problém v podobném rozsahu, můžeme zvolit Temperleyův algoritmus

pro naši aplikaci a pouze otestovat jeho implementaci. Proto jsme tento algoritmus popsali důkladněji. Tabulka 6.1 zmiňuje popsané algoritmy, jejich typ přístupu z hlediska hodnot výškových profilů vstupu a klíčových profilů tóniny, parametry algoritmů a jejich časovou složitost.

Algoritmus	Typ přístupu	Param.	Čas. složitost
LHSA	BinaryIn–BinaryKey	N	$\mathcal{O}(N)$
HPA	WeightedIn–BinaryKey	N, τ	$\mathcal{O}(N \cdot T \cdot M)$
KSA	WeightedIn–WeightedKey	N	$\mathcal{O}(N \cdot T \cdot M)$
TKSA	BinaryIn–WeightedKey	N, P	$\mathcal{O}(N \cdot T(M + T))$

Tabulka 6.1: Algoritmy detekce tóniny

Kapitola 7

Volba algoritmů a testování

V předchozích kapitolách jsme popsali algoritmy, které, jak bylo navrženo v kap. 2.6, společně řeší problém hudební transkripce. Před samotnou tvorbou transkripčního systému musíme nejprve z popsaných algoritmů zvolit pro každý problém ten nejlepší. Otázka, který je nejlepší, však není tak jednoduchá. Úspěšnost zejména nízkoúrovňových algoritmů je totiž silně závislá na typech vstupu, kterým jsou vystaveny. Proto volbu těchto algoritmů provedeme na základě důkladného testování na co nejrozmanitějších vstupech.

Mnoho z těchto algoritmů je závislých na vstupních parametrech, pro jejichž volbu neexistuje jasný návod. Proto bude jejich volba vycházet z výsledků testování na různých hodnotách vstupních parametrů. Při implementaci algoritmů je nutné dělat mnoho dalších rozhodnutí, která nejsou z obecného popisu algoritmu zřejmá. Návrh algoritmů pro modelování lidského vnímání je proces velmi náchylný k chybám, zejména u algoritmů volby vrcholů (Muller et al., 2011). Abychom tento proces co nejvíce usnadnili, vytvořili jsme *testovací framework* pro vývoj algoritmů hudební transkripce. Cílem frameworku je poskytnout prostředí pro pohodlný vývoj algoritmů pro všechny zmíněné podproblémy hudební transkripce. Snažíme se v něm unifikovat přístup k návrhu algoritmů a sjednotit formát různých datasetů určených k testování. Při vývoji algoritmů pro libovolný podproblém pak člověk může využívat stejné rozhraní a sdílet prostředky, kdykoliv je to možné. Dalším cílem bylo vytvořit framework co nejmodulárnější, aby bylo co nejjednodušší do něj přidávat nové algoritmy

a postupy, které se nevešly do rozsahu této práce. Tento framework se skládá z části určené pro nízkoúrovňovou analýzu, jejíž vstupem jsou datasety v audio formátu, a z části určené pro vysokoúrovňovou analýzu, která testuje algoritmy na datasetech ve formátu MIDI.

V této kapitole si nejprve popíšeme samotný testovací framework a některé jeho nástroje, které jsou užitečné pro návrh algoritmů hudební transkripce. Poté si definujeme vhodné metriky pro volbu algoritmů tak, aby byly použitelné v mobilní aplikaci pro operační systém iOS a dosáhly nejlepší kvality transkripce. Nástroje k měření těchto metrik byly společně se všemi testovanými algoritmy implementovány v testovacím frameworku. Při implementaci algoritmů byl dán důraz na jejich modularitu a snadnou přenositelnost do mobilní aplikace. Např. pro nízkoúrovňové operace byl použit framework *Accelerate*, jehož algoritmy mají velké využití díky jeho efektivní implementaci různých matematických operací.

7.1 Testovací framework

V této kapitole si stručně popíšeme návrh frameworku určeného k testování algoritmů hudení transkripce. Ten je rozdělen do dvou částí – nízkoúrovňové a vysokoúrovňové analýzy, a to dle návrhu dekompozice problému v kap. 2.6. Jeho implementaci v jazyce C++ lze nalézt v příloze A.

7.1.1 Framework pro nízkoúrovňovou analýzu

Vstupem nízkoúrovňové analýzy jsou audio soubory či popisy synteticky generovaných signálů. Výstupem je pak pro detekci výšky melodická linka a pro detekci not speciální formát pro reprezentaci hudebních událostí. Základní proceduru nízkoúrovňové analýzy lze popsat následujícím pseudokódem:

```
1  procedure PerformAnalysis(datasety, algoritmy)
2      for all d ∈ datasety do
3          for all s ∈ d → skladby do
4              for all a ∈ algoritmy do
5                  Připrav algoritmus a na skladbu s.
6                  while Není konec skladby s do
7                      s → Předej další okno algoritmu a
8                      a → Proveď analýzu okna
9                      Porovnej výsledek analýzy s referenčními údaji a ulož.
10                 Zaznamenej výsledky analýzy algoritmu a na skladbě s.
```

Jak lze v tomto pseudokódu vidět, vstupem nízkoúrovňové analýzy je množina datasetů a algoritmů, které jsou pro analýzu použity. Nyní si popíšeme, jak takový algoritmus a dataset obecně vypadá.

7.1.1.1 Rozhraní algoritmů nízkoúrovňové analýzy

Každý algoritmus musí implementovat následující rozhraní:

- *void* **prepareBufferForScore**(*TScore** score)
  tato funkce provede přípravu na výpočet skladby *score*. Struktura *TScore** nese informace o délce okna a vzorkovací frekvenci, na základě kterých by mohl algoritmus chtít přizpůsobit své datové struktury.
- *string* **getDescription**()
  Tato metoda vrací jednoznačný a stručný identifikátor algoritmu (např. ACF A).

Algoritmy detekce výšky pak vyžadují metodu:

- *double* **compute**()
  Pro aktuální okno signálu uložené v poli *buffer* spočte MIDI hodnotu jeho výšky.

Přístup k provádění detekce nástupů not se od analýzy výšky trochu liší. Jelikož je detekce nástupů prováděna s jistou latencí, algoritmus pro každé okno spočte hodnotu detekční funkce, která je poslána do série filtrů zakončené detektorem vrcholů. Proto je zapotřebí implementovat následující metody:

- *Konstruktor*
  Za pomocí metod **addFilter** a **setPeakPicker** definuje průběh algoritmu detekce výšky. Filtry pouze transformují detekční funkci a příkladem může být mediánový či derivační filtr. Algoritmus volby maxima (*peakPicker*) přijímá hodnoty detekční funkce a na jejich základě vrací pomocí metody **returnEvent** detekovanou hudební událost. Filtry i algoritmy volby maxima mají jednoduché rozhraní:
  - *void* **addData**(*double* val)
    Zpracuje aktuální hodnotu (*val*) detekční funkce a pomocí metody **returnData**(*double* val) a pošle ji v hierarchii procedur dále.
- *void* **compute**()
  Pro aktuální okno signálu uložené v poli *buffer* spočte hodnotu detekční funkce d, která bude za pomocí volání **pushDetectionFunction**(d) poslána do výše definované sekvence procedur.

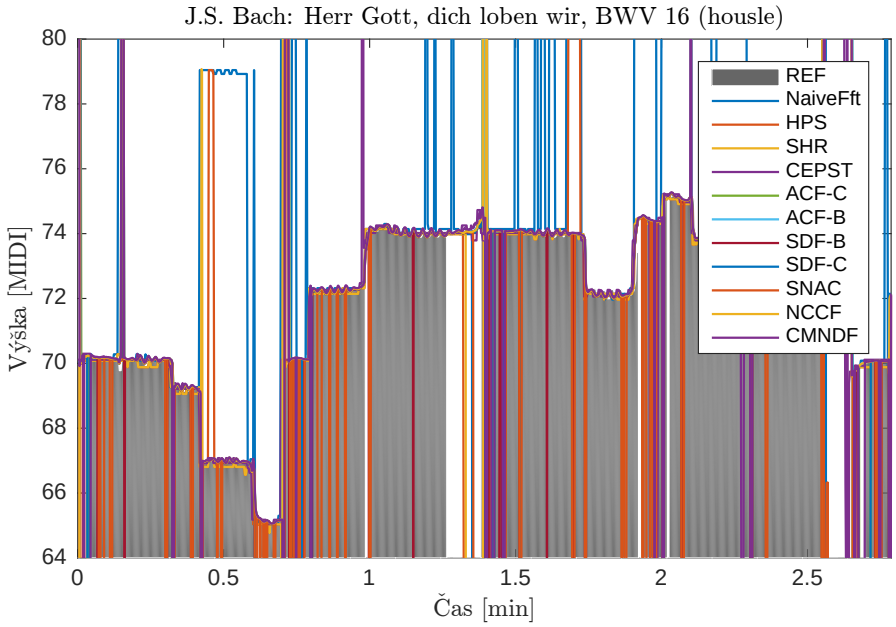
7.1.1.2 Rozhraní datasetů nízkoúrovňové analýzy

Každý dataset musí implementovat následující rozhraní:

- *Konstruktor*
  Připraví informace o jednotlivých skladbách datasetu a uloží je do pole *scores*. Na těchto skladbách bude sekvenčně puštěno testování.
- *string* **fillNextBuffer**(*TScore&* score, *Float32** buffer, *int* bufferLength)
  Vloží aktuální okno signálu skladby $score$ do pole *buffer* o velikosti *bufferLength*.
- *string* **calcRefForNextBuffer**(*TScore&* score)
  Vypočte či zjistí referenční informaci o aktuálním okně dané skladby.

7.1.1.3 Nástroje pro vývoj algoritmů

Kromě základních tříd pro vývoj algoritmů a přípravu datasetů obsahuje framework ještě několik nástrojů a pomocných tříd k usnadnění vývoje. Patří mezi ně třídy datasetů, jež mají vyřešený vstup z audio souborů a poskytují je svým dceřiným třídám. Framework také nabízí mnoho filtrů



Obrázek 7.1: Vizualizace výstupů analýzy výšky tónu na různých algoritmech.

pro návrh detekčních funkcí nástupu not. Dále framework obsahuje třídy *Logger*, který poskytuje rozhraní pro přehledné zaznamenávání zpráv, a *Plotter*, který umožňuje snadné ukládání výsledků testování. Tyto výsledky lze pak jednoduše vizualizovat za pomocí skriptů v jazyce *MATLAB*.

Příklad jedné z těchto vizualizací lze vidět na obr. 7.1, kde je vidět výstup z testování několika různých algoritmů detekce výšky. Šedá plocha na pozadí znázorňuje referenční výšku, zatímco barevné čáry ukazují výstupy jednotlivých algoritmů. Z této vizualizace lze například vyčíst frekventovanost oktávových chyb algoritmů spektrální analýzy, nepřesné zachycení nástupů not a velmi přesné sledování vibrata houslí kepstrální analýzou.

7.1.2 Framework pro vysokoúrovňovou analýzu

Na vyhodnocování algoritmů vysokoúrovňové analýzy by bylo obtížné definovat přesné rozhraní provádění testů, aniž by to omezovalo jejich návrh, a to z důvodů výrazné odlišnosti jejich úkolů. Můžeme však alespoň definovat rozhraní vkládání dat, aby získané datasety bylo možné použít pro vstup všech úkolů vysokoúrovňové analýzy. Jak již bylo zmíněno, pro vstup použijeme MIDI formát, pro který budeme mít speciální reprezentaci, aby jej bylo možné případně obohatit o další informace (např. způsob artikulace noty).

7.1.2.1 Rozhraní algoritmů vysokoúrovňové analýzy

Každý algoritmus musí implementovat následující rozhraní:

- *void* **addEvent**(*MEvent* mEvent)
  Zpracuje hudební událost $mEvent$. Tou je většinou nástup či uvolnění tónu. Datová struktura $MEvent$ s sebou nese informace o výšce tónu, času události a její prominenci (číslo z intervalu $[0, 1]$).

- *void* **finish**()
  Algoritmus je touto metodou informován o konci skladby. Na základě této informace tak může přehodnotit poslední hudební události, např. tedy může zohlednit vliv kadence na analýzu tóniny.

- *string* **getDescription**()
  Tato metoda vrací jednoznačný a stručný identifikátor algoritmu (např. $\mathrm{MachineRhythm}$).

7.1.2.2 Rozhraní datasetů vysokoúrovňové analýzy

Každý dataset musí implementovat následující rozhraní:

- *unsigned* **getScoreCount**()
  Vrátí počet skladeb v datasetu.

- *unsigned* **getTrackCount**(*int* s)
  Tento je dataset určený především pro monofonní analýzu, a tak pokud dataset obsahuje vícehlasé melodie, musíme analyzovat každou melodii zvlášť. Tímto přístupem získáme možnost využít mnohem více dostupných datasetů. Proto tato metoda vrací počet hlasů v s-té skladbě datasetu.

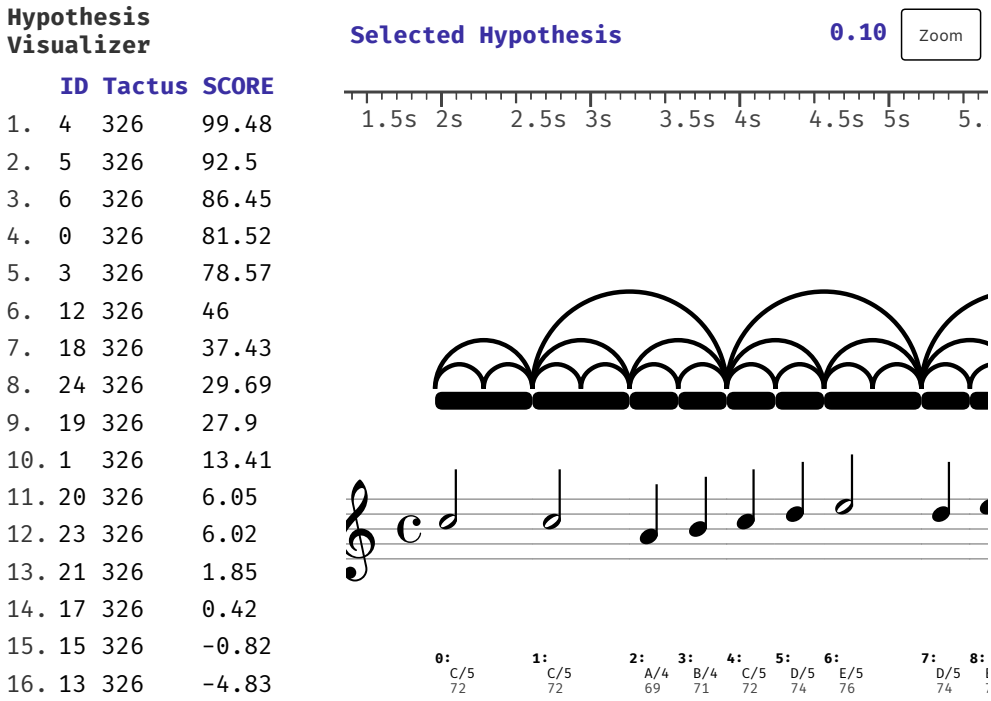
- *vector<MEvent>* **getTrack**(*int* s, *int* t)
  Vrací jednotlivé hudební události t-tého hlasu s-té skladby v datasetu.

7.1.2.3 Nástroje pro vývoj algoritmů

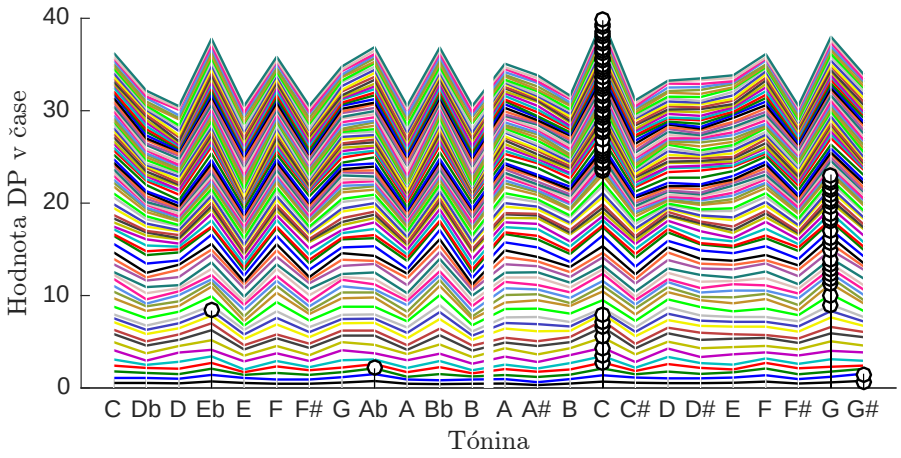
Podobně jako u nízkoúrovňové analýzy, i vysokoúrovňová část frameworku obsahuje pomocné nástroje pro vývoj algoritmů. Jde opět především o třídy umožňující jednodušší vývoj algoritmů, snadnější sledování jejich úspěšnosti a zobrazování dat. Pro detekci rytmu pak framework poskytuje vizualizační nástroj jednotlivých rytmických hypotéz (viz obr. 7.2). Ten umožňuje interaktivně procházet jednotlivé hypotézy a porovnávat jejich skóre od jednotlivých expertů. Pro analýzu detektoru tóniny pak framework pomocí MATLAB skriptů vizualizuje výpočet funkce dynamického programování (viz obr. 7.3).

7.2 Metriky pro volbu algoritmů

V této kapitole si definujeme metriky pro výběr algoritmů do výsledné aplikace. Hlavním cílem těchto metrik je, aby pro každý podproblém hudební transkripce popsal požadavky na algoritmy k jeho řešení tak, aby byly snadno měřitelné a pomohly nám tak algoritmus pro daný problém jednoznačně zvolit. Dalším cílem těchto metrik je formulace informací, které by mohli usnadnit předpověď chování těchto algoritmů na obecných vstupech.



Obrázek 7.2: Vizualizér rytmických hypotéz



Obrázek 7.3: Vizualizace detekce tóniny

7.2.1 Metriky detekce výšky

Hrubá chyba detekce výšky, neboli *Gross-Pitch-Error* (GPE), je nejpoužívanější metrika pro algoritmy detekce výšky (Babacan et al., 2013). Je definovaná jako poměr vzorků, pro které byla detekce výšky vyšší, než určitý práh. Pro analýzu výšky tónu je rozumné použít práh velikosti 50 centů (poloviny půltónu). Tento práh volíme proto, že pro hudební notaci je postačující, je-li výška určena na přesnost půltónu. Pokud bychom zaokrouhlovali na celočíselnou hodnotu MIDI, libovolná hodnota ve vzdálenosti 50 centů od referenční celočíselné hodnoty nebude na výsledku znát. Proto je tato hodnota nejdůležitější ze všech metrik. Formálně je GPE detekované výšky d vůči referenční výšce r definována následovně:

$$\mathrm{GPE}(d, r) = \begin{cases} 1, & \text{pokud } \lfloor d \rceil = \lfloor r \rceil \\ 0, & \text{jinak} \end{cases}, \tag{7.1}$$

kde $\lfloor x \rceil = \lfloor x + 0.5 \rfloor$ je celočíselně zaokrouhlená hodnota x.

Jemná chyba detekce výšky, neboli *Fine-Pitch-Error* (FPE), nám naopak umožňuje měřit jemné chyby detektoru výšky. Je definována jako směrodatná odchylka distribuce relativní chyby hodnot udávaná v centech pro vzorky, které nemají GPE chybu (Babacan et al., 2013). Formálně jí lze pro sekvenci detekovaných výšek d_n a sekvenci referenčních výšek r_n délky n, kde žádný ze vzorků nemá chybu GPE, definovat následovně:

$$\begin{aligned} \mathrm{FPE}(d_n, r_n) = \mathrm{SD}(|d_i - r_i|) = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(|d_i - r_i| - \mu)^2}, \\ \text{kde } \mu = \frac{1}{n}\sum_{i=0}^{n}(|d_i - r_i|) \end{aligned} \tag{7.2}$$

Oktávová chyba (*Octave-Error*, OCT) vznikne tehdy, když detektor odhalí správnou výškovou třídu noty, zvolená oktáva je však chybná. Tato chyba je kvůli efektu oktávové ekvivalence u detektorů výšky velmi častá, a proto pro ní zavedeme speciální metriku. Formálně má detekovaná výška d vůči referenční

výšce r oktávovou chybu právě tehdy, když platí:

$$\text{OCT}(d, r) = \begin{cases} 1, & \text{pokud } GPE(d \bmod 12, r \bmod 12) = 0 \\ 0, & \text{jinak} \end{cases} \tag{7.3}$$

Stejná oktáva Chyba stejné oktávy (*In-Octave*, INO) nám umožňuje odlišit případy, kdy detektor není schopen přesně určit výšku tónu na přesnost půltónu, detekovaná výška je však ve stejné oktávě. Pokud by byl nějaký algoritmus velmi přesný v detekci tónu, ale měl by vysoké oktávové chyby, a druhý algoritmus naopak, jejich kombinace by umožnila využít přednosti obou algoritmů. Formálně je chyba stejné oktávy definovaná následovně:

$$\text{INO}(d, r) = \begin{cases} 1, & \text{pokud } GPE(d, r) = 1 \ \wedge \ |d - r| < 12 \\ 0, & \text{jinak} \end{cases}, \tag{7.4}$$

Počet vzorků, pro které nastala chyba stejné oktávy, je udáván v procentech relativně vůči počtu GPE. Jelikož je chyba ve stejné oktávě více žádoucí než chyba v jiné, snažíme se tuto hodnotu maximalizovat.

Příliš nízko, příliš vysoko, neboli *Too-Low* (LOW), *Too-Hight* (HIG). Tyto chyby rozdělují hrubé chyby detekce výšky (GPE) do dvou kategorií na základě toho, zda k chybě došlo kvůli detekci vyšší či nižší výšky vůči výšce referenční. Pokud jedna tato metrika výrazně převyšuje druhou, je možné, že je způsobena pouze špatně nastavenými parametry algoritmu. To nastane např. pokud je práh v detektoru vrcholů detekční funkce periody zvolen příliš malý, hodnota HIG bude výrazně převyšovat LOW. Číslo LOW, resp. HIG, je udáváno v procentech relativně vůči počtu GPE.

7.2.2 Metriky detekce not

7.2.2.1 Metriky detekce znělosti

Na detekci znělosti lze pohlížet jako na binární klasifikační problém. Každý vzorek můžeme ohodnotit buďto jako znělý, či jako neznělý. Pro tento typ klasifikace jsou již zavedeny mnohé statistické metriky. My z nich budeme využívat následující. Přesnost, neboli *accuracy* definujeme jako:

$$\mathrm{ACC} = \frac{\textit{počet správně klasifikovaných}}{\textit{celkový počet vzorků}}. \tag{7.5}$$

Jelikož bude naší motivací pokud možno neklasifikovat znělý úsek jako neznělý, další metrikou bude tzv. *míra chyb*, neboli *False Negative Rate* (FNR):

$$\mathrm{FNR} = \frac{\textit{počet znělých vzorků označené jako neznělé}}{\textit{celkový počet znělých vzorků}}. \tag{7.6}$$

Pro usnadnění analýzy algoritmů pak ještě budeme používat poměr správně klasifikovaných znělých (VOICED) a správně klasifikovaných neznělých (UNVOICED) vzorků vůči jejich celkovému počtu.

7.2.2.2 Metriky detekce nástupů a uvolnění not

I na detekci not se lze zjednodušeně dívat jako na dva klasifikační problémy, detekce nástupů a detekce uvolnění. Pro každou z těchto hudebních událostí pak můžeme určit, zda je správně umístěna. To uděláme na základě její vzdálenosti od nejbližší referenční události. Pro každou z referenčních hudebních událostí zase můžeme určit, zda je reprezentována nějakou detekovanou událostí. Na základě těchto zjištění pak pro problém detekce nástupu i uvolnění definujeme citlivost (*True Positive Rate*, TPR) následovně:

$$\mathrm{TPR} = \frac{\textit{počet správně umístěných událostí}}{\textit{celkový počet událostí}}. \tag{7.7}$$

Pokud bychom měřili algoritmus pouze na této metrice, nejlépe ohodnocený by byl algoritmus který ohodnotí každý vzorek jako nástup, resp. uvolnění. Proto definujeme přesnost algoritmu (*Precision*, PREC) následovně:

$$\text{PREC} = \frac{\textit{počet správně umístěných událostí}}{\textit{celkový počet umístěných událostí}}. \tag{7.8}$$

Pro zjištění, kolik událostí jsme minuli, definujeme chybějící události (*False Negative Rate*, FNR) následovně:

$$\text{FNR} = \frac{\textit{počet chybějících událostí}}{\textit{celkový počet událostí}}. \tag{7.9}$$

Tyto metriky jsou z hlediska globálních informací o detekci postačující, jelikož ostatní (např. počet chybně umístěných událostí z počtu umístěných událostí) lze již dopočítat.

7.2.3 Metriky detekce rytmu

Pro otestování rytmické analýzy budeme porovnávat jednotlivé rytmické hypotézy. Algoritmus detekce rytmu k tomu poskytuje seznam rytmických hypotéz, seřazený dle toho, jak dle heuristik algoritmu předpokládají skutečné rytmické interpretaci skladby. Tuto hodnotu udává v bezrozměrné jednotce, pro potřeby měření ji však můžeme normalizovat.

K měření úspěšnosti algoritmů detekce rytmu použijeme následující metriky.

- **Pořadí správné hypotézy** (POS)
  Tato metrika udává pozici správné hypotézy v seřazeném seznamu. Pokud se tam nevyskytuje, je tato hodnota rovná nule.

- **Hodnota správné hypotézy** (VAL)
  Po „min-max“ normalizaci můžeme správné hypotéze přiřadit její hodnoty z intervalu $[0, 1]$ dle toho, jak algoritmus předpokládá její správnost v kontextu jiných.

- **Pořadí nejlepšího hlasu skladby** (MINPOS)
  Jelikož budeme algoritmus testovat na datasetu, který obsahuje polyfonní záznamy, musíme je vyhodnocovat separátně. Tímto však může dojít k jistému vychýlení, jelikož skladba může mít nějakou rytmickou interpretace pouze na základě jejího nejprominentnějšího hlasu. Proto definujeme POSSCORE jako maximální hodnota metriky POS napříč jednotlivými hlasy skladby.

- **Průměr pozic skladby** (MAXVAL) Maximální hodnota metriky VAL napříč jednotlivými hlasy skladby.

- **Průměr pozic skladby** (AVGPOS) Průměrná hodnota metriky POS napříč jednotlivými hlasy skladby.

- **Průměr pozic skladby** (AVGVAL) Průměrná hodnota metriky VAL napříč jednotlivými hlasy skladby.

7.2.4 Metriky detekce tóniny

Pro testování algoritmů detekce tóniny u skladeb, které jsou celé napsané v jedné tónině, definujeme následující metriky. Počet chybně detekovaných tónin datasetu značíme ERR, správě detekované tóniny pak značíme OK. Chybu detekce tóniny měříme její vzdáleností na kvintové kružnici. Pokud je tónina navíc v jiném módu (dur/mol), je tato hodnota záporná. Počet tónin, které jsou detekované správně, nebo mají chybu jen v módu, značíme MODOK Pro dataset pak spočteme metriku DISTSD, které udává směrodatnou odchylku absolutní hodnoty této vzdálenosti. Test globální tóniny je dělán dle poslední detekované tóniny skladby. Počet změn napříč segmenty je pak udán hodnotou CHANGES.

7.3 Výsledky testování

Tato kapitola prezentuje výsledky testování algoritmů detekce výšky, tóniny, rytmu, nástupu not a detekce znělosti, jež byly popsány v kapitolách 3-6 pomocí testovacího frameworku popsaného v kap. 7.1.

7.3.1 Algoritmy detekce výšky

Z testování algoritmů detekce výšky jsme vyřadili algoritmy ACF_A, SDF_A a AMDF, jelikož přesahují maximální požadavek na paměťovou složitost určený v kap. 2.7.

Dále zde z důvodu rozsahu práce nepopisujeme způsob volby vrcholu detekční funkce ani nastavování jejich parametrů. Každý z algoritmů byl testován a laděn na různých algoritmech volby vrcholu, až byl zvolen jeden nejlepší. Metoda globálního testování není pro volbu algoritmu pro tento podproblém příliš vhodná, jelikož každý algoritmus vyžaduje speciální péči pro nastavení běhu tohoto algoritmu. Jak píše McLeod (2009, str. 16), volba vrcholu detekční funkce byla vždy „černou magií“ mezi algoritmy detekce výšky a vyžaduje výrazně heuristický přístup. Jelikož se nám pro detekční funkci WACF nepodařilo nalézt vhodný algoritmus, který by dosahoval lepších, než průměrných výsledku, byl též ze srovnání vyřazen.

Pro srovnání jsme přidali jednoduchý algoritmus (*NaiveFFT*), který pouze provede spektrální analýzu a zvolí její maximum jako detekci výšky. Tento algoritmus je příliš naivní a nemá ambice na volbu pro výslednou aplikaci, v testování jej však zmiňujeme pro demonstraci chybného výkladu Ohmova akustického zákon, který je stále velmi zakořeněn.

7.3.1.1 Dataset Bach10

Dataset Bach10 od Zhiyao Duana a Bryan Parda je menší, volně dostupný dataset, sestávající z 40 hudebních nahrávek deseti čtyřhlasých chorálů od J. S. Bacha (Duan et al., 2010). Kromě audio a MIDI souborů obsahuje přesnou melodickou linku pro každou hudební nahrávku, která umožňuje přesné otestování algoritmů detekce výšky. Melodická linka je udaná v MIDI hodnotách pomocí reálných hodnot. Reprezentuje tak i drobné změny výšky a obsahuje i tedy údaje o výšce „na pomezí“ dvou MIDI hodnot. Tento dataset je tedy ideální pro otestování našich algoritmů v kontextu skladby, což na rozdíl od testování na izolovaných notách přináší možnost zjistit, jak hranice not ovlivňuje úspěšnost algoritmu.

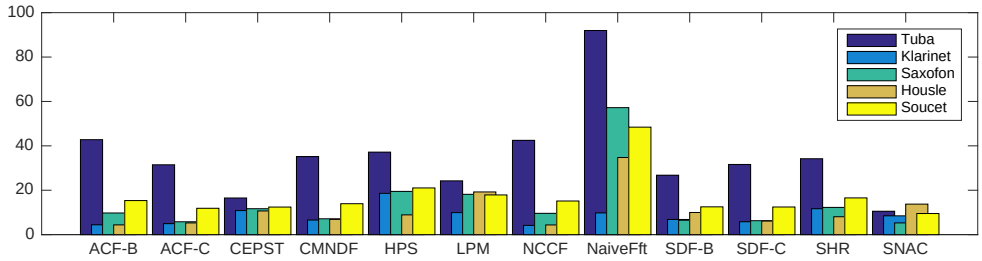
Jméno	GPE (%)	FPE (cent)	OCT (%)	INO (%)	LOW (%)	HIG (%)
ACF-B	15,346	5,288	34,99	31,874	13,177	86,823
ACF-C	11,876	5,104	34,252	44,273	23,528	76,472
CEPST	12,426	6,346	13,773	**78,799**	55,382	44,618
CMNDF	13,935	4,814	28,143	26,562	28,350	71,650
HPS	21,037	11,485	14,247	66,062	74,070	25,930
LPM	17,891	8,021	20,178	32,438	83,735	16,265
NCCF	15,154	5,292	35,217	32,118	11,73	88,270
NaiveFft	48,434	7,638	53,354	35,173	6,886	93,114
SDF-B	12,524	**4,627**	33,716	41,596	45,377	54,623
SDF-C	12,455	4,837	34,839	37,832	30,918	69,082
SHR	16,551	11,877	**4,6786**	58,762	65,416	34,584
SNAC	**9,511**	4,793	36,638	32,761	80,260	19,740

Tabulka 7.1: Výsledky testování detekce výšky na datasetu Bach10

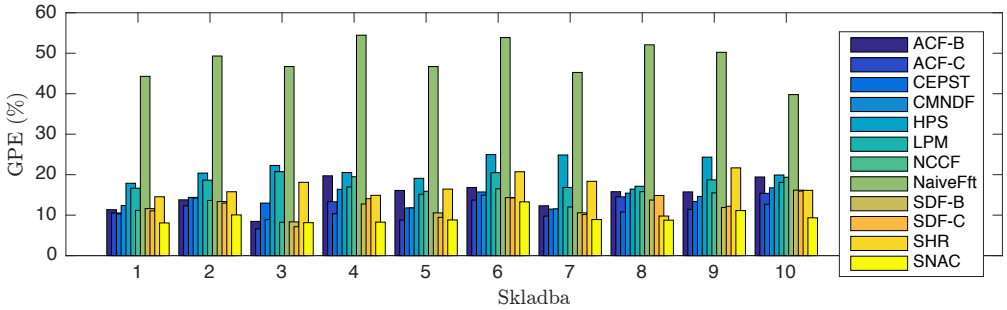
Hudební nahrávky těchto děl pochází ze čtyř různých hudebních nástrojů: housle, klarinet, saxofon a tuba, tedy dva dřevěné dechové, jeden žesťový dechový a jeden smyčcový strunný nástroj. Referenční informace o výšce jsou dány v intervalech 10 ms, což je pro naši aplikaci postačující.

Výsledky základních metrik na celém datasetu jsou pro každý algoritmus znázorněny v tabulce 7.2. Nejlépe zde fungoval algoritmus SNAC, který jako jediný dosáhl na celém datasetu chyby menší než 10 %. Při hlubší anaýze těchto chyb vyšlo najevo, že ani referenční data nejsou naprosto bezchybná, a proto některé z těchto hodnot mohou být lehce vychýlené. Z těchto chyb je 36,6 % oktávových a 32,7 % chyb bylo ve stejné oktávě. Nepoměr mezi LOW a HIG (8:2) se zdá být výrazným, důkladnější nastavování parametrů však tyto výsledky nezlepšilo a zdá se že pochází z návrhu algoritmu samotného. I na jemných chybách (FPE) si tento algoritmus vedl velice dobře, kdy se 4,7 centy směrodatné odchylky dosahuje druhého nejlepšího výsledku a ukazuje se tedy ve svých odhadech velmi přesný.

Druhým nejlepším algoritmem z hlediska hrubé chyby je ACF-C, jehož výsledek se však neliší příliš od třetího, kterým je algoritmus CEPS. Nejlepší odolnost proti oktávových chybám ukázal algoritmus SHR, kde se svými 4 % z celkového počtu GPE chyb překonává druhý nejlepší výsledek (CEPST) téměř trojnásobně.



Obrázek 7.4: Porovnání GPE na datasetu Bach10 dle nástrojů



Obrázek 7.5: Porovnání GPE na datasetu Bach10 dle skladeb

To není překvapivé, jelikož byl algoritmus SHR navržen přesně pro tento účel. V metrice INO je jednoznačně nejlepší opět kepstrální analýza a s druhým nejlepším výsledkem v oktávových chybách se z něj stává nejodolnější algoritmus pro volbu oktávy.

Na obr. 7.4 jsou znázorněny hodnoty výsledků testování jednotlivých algoritmů v závislosti na nástroji skladby. Je zde vidět, že většina algoritmů má problémy především s tubou, což může být způsobené tím, že jde o jeden z nejhlouběji znějících nástrojů. Na obr. 7.5 je vidět úspěšnost algoritmů dle skladby. Zde si můžeme povšimnout výraznějšího rozdílu úspěšnosti autokorelačních algoritmů vůči různým skladbám, jinak se zde příliš velké odchylky neobjevují.

Z výrazného neúspěchu tuby na mnoha algoritmech lze usuzovat, že typ nástroje může mít velký vliv na úspěšnost detekce výšky. Jelikož dataset Bach10 nabízí pouze tři rodiny nástrojů, je žádoucí nalézt nějaký dataset, který by tuto mezeru doplnil.

7.3.1.2 Dataset Philharmonia

Philharmonia Orchestra je britský symfonický orchestr, který na svém webu nabízí tisíce zvukových nahrávek tónů různých nástrojů (Philharmonia Orchestra, 2017). Pro jejich rozsáhlost a množství jsme se rozhodli tento dataset použít.

Dataset obsahuje 13 521 nahrávek, kde na každé z nich je jediná nota hraná na jeden z 19 různých hudebních nástrojů. Každá nahrávka je identifikovatelná nástrojem, délkou noty, její dynamikou a artikulací. Pro každý nástroj existuje nahrávka téměř pro každou notu z jeho rozsahu.

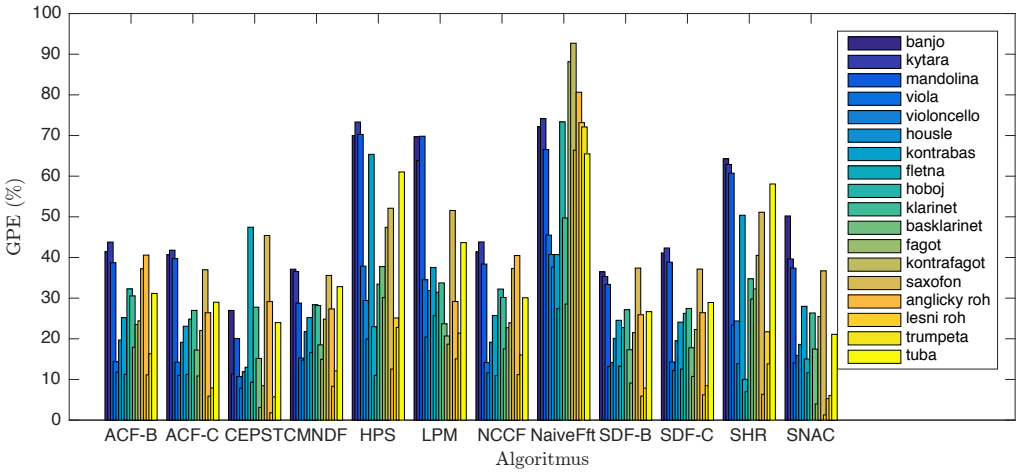
Při testování tohoto algoritmu se však vyskytla následující komplikace. Jelikož naše algoritmy produkují melodickou linku, která není definovaná v neznělých oblastech nahrávky, byly by výsledky testování na jejím celém rozsahu velmi vychýlené. Proto jsme museli použít detektor znělých úseků, na což jsme použili algoritmus EnZCR (viz kap. 4.3.1.2). Hodnotu jeho prahu jsme nastavili obzvláště vysoko, abychom měli jistotu, že v daných úsecích tón opravdu zní a výsledky nebyly vychýlené.

Abychom si ověřili, že závěry z výsledků na základě předchozího testování platí i ve větším rozsahu, otestovali jsme algoritmy stabilnější podmnožině datasetu, která neobsahovala neobvyklé artikulační ani dynamické efekty. Výsledky testování na těchto 7 936 záznamech lze vidět v tab. 7.2. Algoritmus SNAC zde stále vykazuje nejlepší úspěšnost a algoritmus CEPST zase nejlépe odhaduje oktávu. Hodnoty chyb jsou zde pro všechny algoritmy vyšší, což je způsobené tím, že některé nahrávky nejsou té nejlepší kvality. Obzvláště u hodně vysokých či nízkých tónů je i pro člověka velmi těžké výšku na těchto nahrávkách rozpoznat. To však nijak výrazně nemusí ovlivňovat výsledky testování, jelikož jde stále o validní vstup a podmínky mají všechny algoritmy stejné.

Na obr. 7.6 lze vidět výsledky algoritmů na jednotlivých nástrojích. Na agregovaném pohledu na obr. 7.7 vidíme, že převážně drnkací nástroje (banjo, kytara a mandolína) dělají algoritmům problémy. To je dané zejména stavbou jejich tónu, která se vyznačuje velmi krátkou zakmitanou částí. Tento problém lze

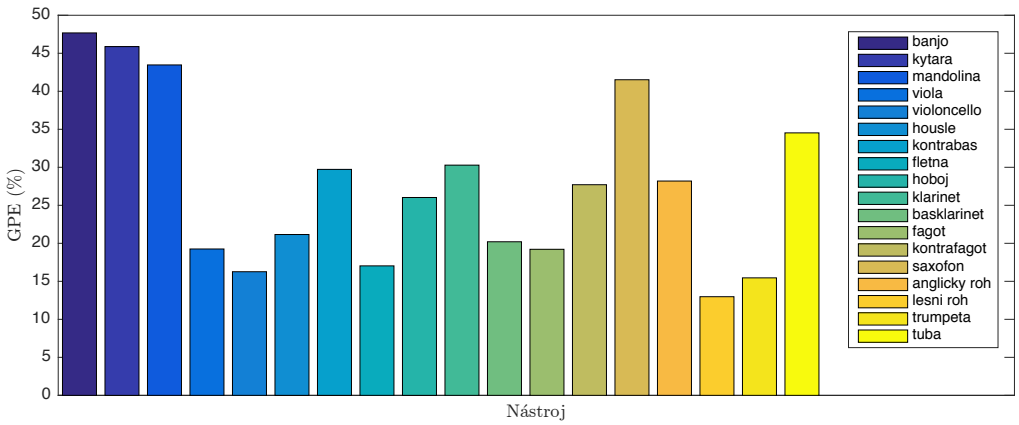
Jméno	GPE (%)	FPE (cent)	OCT (%)	INO (%)	LOW (%)	HIG (%)
ACF-B	29,887	6,1059	54,921	42,415	30,148	69,852
ACF-C	24,987	6,1391	52,428	51,148	37,354	62,646
CEPST	21,208	5,7856	58,381	**54,088**	88,166	11,834
CMNDF	26,553	6,4759	50,737	41,194	37,786	62,214
HPS	37,584	6,1437	20,587	53,382	75,947	24,053
LPM	34,003	8,6305	24,921	30,399	76,866	23,134
NCCF	29,639	6,0478	55,404	42,542	30,08	69,92
NaiveFft	66,871	**3,9215**	52,265	20,283	13,426	86,574
SDF-B	24,344	6,1849	51,469	52,884	45,162	54,838
SDF-C	25,391	6,1259	52,025	50,089	40,184	59,816
SHR	32,685	5,6577	**11,117**	37,298	80,848	19,152
SNAC	**21,067**	6,1712	38,222	48,942	77,167	22,833

Tabulka 7.2: Výsledky testování detekce výšky na datasetu Philharmonia



Obrázek 7.6: Porovnání GPE na datasetu Philharmonia dle algoritmů a nástrojů

řešit v pomocí silného algoritmu pro detekci not, který tuto zakmitanou část noty identifikuje a bude odsud přebírat informace o výšce. Jelikož jsme zde prováděli pouze detekci znělosti, nakmitávací i dokmitávací část tónu mohla silně ovlivnit výsledek testování.

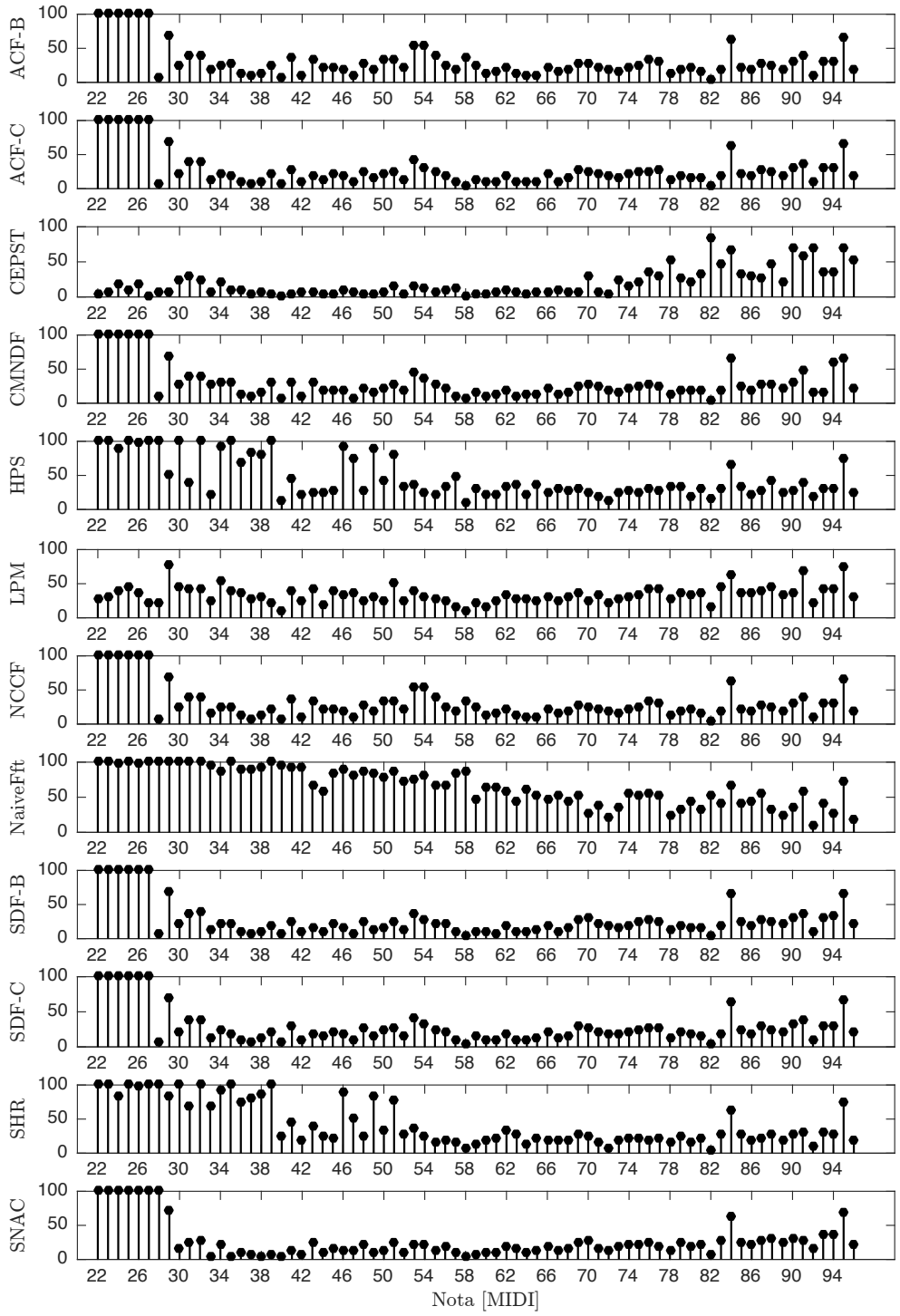


Obrázek 7.7: Porovnání GPE na datasetu PHILHARMONIA dle nástrojů

Jelikož každý záznam datasetu PHILHARMONIA obsahuje pouze jednu notu, můžeme se podívat na závislost úspěšnosti testování na jednotlivých výškách. Zde si povšimněme, že dataset obsahuje i noty nižší, než je náš stanovený rozsah (viz kapitola 3.2). Noty v datasetu jsou v rozsahu A_0^b–C_7 zatímco my detekujeme pouze rozsah E_1–C_8. Podívejme se však, jak se algoritmy chovají, pokud se setkají s nižší notou.

Na obr. 7.8 je znázorněná GPE v závislosti na výšce noty pro jednotlivé algoritmy. Vidíme zde, že málokteré algoritmy detekovali správně libovolné noty nižší než E_1 (MIDI 28). Na jednotlivých grafech lze vidět, že úspěšnost skoro periodicky klesá a stoupá. To je dáno tím, že se střídavě přibližuje a oddaluje celému počtu period v okně. Dále vidíme, že úspěšnost algoritmu SNAC klesá s rostoucí výškou. To je způsobeno tím, že vyšší noty mají více period v okně a kvazistacionarita signálu se projevuje výrazněji.

Algoritmus LPM projevuje stabilitu napříč celým rozsahem a detekuje dokonce i velmi nízké noty. Jeho průměrná GPE je však příliš vysoká. Kepstrum též detekuje i nejnižší noty, a jeho úspěšnost klesá s rostoucí výškou ze stejného důvodu, jako u algoritmu SNAC.



Obrázek 7.8: Porovnání GPE na datasetu Philharmonia dle algoritmů a výšky tónu

7.3.1.3 Kombinace algoritmů

V závěru této kapitoly uvádíme výsledky experimentu, který jsme provedli na základě předchozích výsledků. Algoritmus SNAC působí nejstabilněji a pokud bychom měli vybrat jeden algoritmus, rozhodně bychom zvolili tento. Na základě výsledků testování bychom se mohli snažit vylepšit nedostatky algoritmu, zejména hledat způsoby a nové heuristiky pro odstranění *Too-Low* chyb. O určité úpravy (zejména v algoritmu volby maxima) jsme se pokusili, nicméně po otestování těchto změn na jiných datech nepoužitých pro nastavení těchto heuristik se celková chyba nezlepšila. Z tohoto hlediska se zdá, že algoritmus již nemá dostatek způsobů pro další zlepšování pomocí drobných změn parametrů, aniž by došlo k přeučení. Novou snahou tedy bylo přinést do algoritmu nezávislé znalosti z algoritmů jiných, proto jsme v testovacím frameworku implementovali funkci kombinace algoritmů.

Při kombinaci algoritmů v testovacím frameworku je naším cílem najít takovou podmnožinu testovacích algoritmů, která dává společně lepší výsledky než každý algoritmus samostatně. Proto jsme pro každý algoritmus implementovali metodu volby vrcholů, která nevrací pouze jednoho, ale až K nejlepších kandidátů pro výšku. Pro každou podmnožinu algoritmů jsou pak tito kandidáti váženi dle hodnoty detekční funkce a normalizovány do rozsahu $[0, 1]$. Z těchto kandidátů je pak vytvořena nová detekční funkce, na které opět hledáme vrchol.

Jelikož by se kombinace více jak tří algoritmů již mohla výrazně projevit na časové i paměťové složitosti, uvažovali jsme pouze podmnožiny algoritmů velikosti tři a menší. Výsledky tohoto přístupu lze vidět v tabulkách 7.3 (Bach10) a 7.4 (Philharmonia). Pro efektivní implementaci tento přístup měří pouze metriku GPE a metriky s ní související, a proto sloupec FPE v tabulkách chybí. Tyto výsledky se však nezdají být příliš vypovídající, jelikož se vzájemně liší velmi málo a na různých vstupech je výsledek pro různé kombinace odlišný. Proto je na místě informovanější přístup ke kombinaci algoritmů, který prozatím necháme jako otevřený problém.

Jméno	GPE (%)	OCT (%)	INO (%)	LOW (%)	HIG (%)
CEPST,NCCF,SDF-C	5,2189	21,438	55,659	68,668	31,332
CEPST,NCCF,ACF-C	5,385	20,506	55,827	64,679	35,321
SNAC,CEPST,NCCF	5,3919	20,338	54,994	65,537	34,463
CEPST,NCCF	5,4962	19,155	61,639	52,396	47,604
CEPST,ACF-C,SDF-C	5,6616	20,695	47,649	72,493	27,507
CEPST,ACF-C	5,7457	19,461	58,587	54,886	45,114
SNAC,CEPST,SDF-C	5,7707	20,388	44,593	74,338	25,662
SNAC,CEPST	5,8054	19,188	57,757	56,416	43,584
SNAC,CEPST,ACF-C	5,8499	20,005	46,591	70,005	29,995
CEPST,SDF-C	5,8652	19,325	55,995	61,552	38,448
NCCF	5,9819	20,713	48,699	61,757	38,243
NCCF,ACF-C	6,3072	20,284	43,323	67,993	32,007
SNAC,NCCF	6,3586	20,557	41,333	69,497	30,503
SNAC,NCCF,ACF-C	6,4844	20,566	38,999	71,536	28,464
NCCF,ACF-C,SDF-C	6,6873	22,841	35,218	73,044	26,956

Tabulka 7.3: Výsledky testování kombinace algoritmů detekce výšky na datasetu Bach10

7.3.2 Algoritmy detekce not

7.3.2.1 Algoritmy detekce znělosti

Pro otestování detekce znělosti můžeme též použít melodickou linku z datasetu Bach10. Ta má tu vlastnost, že v neznělých částech skladby je nulová. Otestování algoritmů detekce znělosti jsme provedli ve dvou krocích. V prvním kroku jsme zjistili ideální hodnoty parametrů na menší části datasetu pomocí ternárního vyhledávání. Tyto hodnoty jsme pak otestovali na zbytku datasetu a výsledky tohoto testování lze nalézt v tabulce 7.5. Jelikož je toto testování provedeno na stejném datasetu, je vychýlené hlasitostí nahrávky a mírou šumu. Nicméně všechny zmíněné metody se jeví jako velmi schopné provádět detekci znělosti, jakmile jsou správně nastaveny. Závislost na nastavování hodnot však není praktická a je tu tedy výrazný prostor pro zlepšování.

Jméno	GPE (%)	OCT (%)	INO (%)	LOW (%)	HIG (%)
CEPST,SHR,SDF-C	36,183	25,949	30,629	52,643	47,357
SNAC,CEPST,SHR	37,412	25,007	34,055	55,675	44,325
CEPST,NCCF,SHR	37,57	25,569	34,632	54,91	45,09
CEPST,SHR,ACF-C	37,709	24,814	33,717	56,083	43,917
CEPST,NCCF	38,418	32,485	23,524	73,418	26,582
SNAC,CEPST	39,564	31,98	20,939	74,629	25,371
CEPST,SDF-C	39,667	31,124	22,595	65,502	34,498
CEPST,ACF-C	39,873	31,687	20,755	74,823	25,177
CEPST	40,183	30,8	26,921	68,668	31,332
CEPST,CMNDF,SHR	41,438	21,992	24,236	60,069	39,931
CEPST,NCCF,SDF-C	42,013	28,255	19,658	70,236	29,764
SNAC,CEPST,NCCF	42,446	29,035	17,409	78,375	21,625
CEPST,NCCF,ACF-C	42,731	28,87	17,275	78,548	21,452
SNAC,CEPST,SDF-C	43,116	27,782	19,041	71,4	28,6
CEPST,ACF-C,SDF-C	43,363	27,681	18,91	71,572	28,428

Tabulka 7.4: Výsledky testování kombinace algoritmů detekce výšky na datasetu Philharmonia

Jméno	ACC (%)	FNR (%)	VOICED (%)	UNVOICED (%)
MAX	95,3807	1,1793	98,6741	68,8820
AVG_Rct	95,2636	1,6279	98,1692	71,9509
AVG_Sqr	95,3268	1,4431	98,3771	70,8529
AVG_Log	95,1946	1,7976	97,9784	72,8589
EZCR	94,7138	2,0745	97,6669	71,0191
NLFER	94,3939	2,5648	97,1096	73,0037
SpectralVariance	94,3191	2,7112	96,9463	73,5225

Tabulka 7.5: Výsledky testování detekce znělosti na datasetu Bach10

7.3.2.2 Algoritmy detekce nástupů not

K otestování algoritmů detekce not můžeme též použít již zmiňovaný dataset Bach10. Ten kromě melodické linky obsahuje i MIDI soubory skladeb a mapovací soubor, který přiřazuje jednotlivým MIDI událostem jejich pozici v nahrávce. Z těchto dvou souborů máme dostatečné informace o nástupech a uvolněních not a máme tak referenční údaje pro otestování algoritmů.

Jméno	onTPR (%)	onPREC (%)	onFNR (%)
ENVDERIV	55,5383	**90,0168**	44,46
ENV_{sqr}	71,5839	83,0631	28,42
ENV_{rect}	69,6687	88,5526	30,33
HFC	**71,9979**	83,2436	**28,00**
SD_{ℓ_1}	68,5818	70,8935	30,43
SD_{ℓ_2}	67,9607	69,3975	31,42
$\text{SD}_{\cos}$	55,1242	54,6154	32,04
PD	69,5652	73,6842	44,88

Tabulka 7.6: Výsledky testování detekce **nástupu** not na datasetu Bach10

Jméno	offTPR (%)	offPREC (%)	offFNR (%)
ENVDERIV	48,1253	78,6074	87,05
ENV_{sqr}	66,3585	77,5976	33,64
ENV_{rect}	60,8115	77,8947	39,19
HFC	66,6153	77,6182	**33,38**
SD_{ℓ_1}	76,1685	**79,3472**	83,92
SD_{ℓ_2}	75,7062	77,9070	91,78
$\text{SD}_{\cos}$	**77,6579**	77,5385	92,04
PD	16,0760	17,1601	47,20

Tabulka 7.7: Výsledky testování detekce **uvolnění** not na datasetu Bach10

V tabulce 7.7 jsou výsledky tohoto testování. Jednotlivé algoritmy volby maxima a filtry pro úpravu a čištění detekční funkce byly zvoleny na základě jejich testování a vizualizací dat. Vítězná metoda HFC generuje detekční funkce z oken délky 1024 na základě metody popsané v kap. 4.3.2.3. Tuto detekční funkci vyhladíme mediánovým filtrem délky 17 vzorků. Poté ji normalizuje klouzavým mediánem a použije mediánem vážené prahování pro volbu maxima.

Ačkoliv tento algoritmus dosahuje z hlediska našich metrik nejlepších výsledků, přibližně 16 % z oznámených nástupů neexistovalo a 28 % nástupů algoritmus nezaregistroval, což je pro transkripční systém příliš mnoho. Proto musíme problém detekce nástupů not označit za nevyřešený.

7.3.3 Algoritmy detekce rytmu

Pro testování detekce rytmu jsme též použili dataset Bach10. Data k testování jsme získali na základě zarovnaných MIDI souborů, které obsahovaly informace o fázi jednotlivých rytmických úrovních. Typy rytmů jsme převzali z originální notace.

Algoritmus popsaný v kapitole 5.3 jsme otestovali na všech deseti skladbách tohoto datasetu, a to na každém hlasu zvlášť. Výsledky metrik z kapitoly 7.2.3 jsou vidět v tabulce 7.8 vlevo. Každý řádek zde odpovídá jedné skladbě z datasetu. Při analýze chyb jsme zjistili, že mnohé jsou způsobené fenoménem tzv. *půlení* či *zdvojnásobení* rytmu (Gouyon et al., 2006). Podobně jako člověk se při poslechu hudby může rozhodnout, že bude klepat dvakrát (či třikrát, dle typu úrovně) rychlejí a stále bude klepat správný rytmus, i náš algoritmus má tyto sklony. Abychom neoznačili tyto detekce za chybné, upravíme testování tak, porovnával rytmické úrovně při libovolném možném zarovnání úrovní. Výsledky této úpravy lze vidět v tab. 7.8 vpravo. Vidíme, že s touto úpravou algoritmus přiřadil správnou hypotézu pro každou skladbu alespoň v jednom hlasu.

MINPOS	MXVAL	AVGPOS	AVGVAL
1	1	10	0,76
14	0,838	20	0,61
7	0,943	23	0,50
1	1	7,5	0,72
1	1	1	1
1	0,962	9,75	0,80
3	0,957	7,75	0,67
1	1	1	1
1	1	1	1
1	1	7,75	0,73

MINPOS	MXVAL	AVGPOS	AVGVAL
1	1	1,75	0,95
1	1	2,00	0,99
1	1	1,5	0,995
1	1	1,25	0,992
1	1	1	1
1	1	1,5	0,992
1	1	2,25	0,957
1	1	1	1
1	1	1	1
1	1	1	1

Tabulka 7.8: Výsledky testování detekce rytmu na datasetu Bach10

P	OK	modOK	Changes	distSD
0	0	0	168	2,2
0,02	0	0	168	2,2
0,04	0	0	168	2,2
0,06	0	0	156	2,2
0,08	0	0	60	2,2
0,1	0	0	24	2,2
0,12	24	24	0	0
0,14	24	24	0	0

Tabulka 7.9: Výsledky testování detekce tóniny na diatonických tóninách

Výše zmíněná úprava testování však nemusí být vždy validní. Algoritmus totiž pro toto porovnání nemusí mít vygenerováno dostatek rytmických úrovní a výsledek pak bude lehce vychýlený. To je dáno tím, že nevíme předem, kolik úrovní bude zapotřebí. Jelikož počet úrovní ovlivňuje pořadí hypotéz, máme zde podobný problém, se kterým jsme se setkali při volbě velikosti okna u detekce rytmu.

7.3.4 Algoritmy detekce tóniny

Testování algoritmu pro detekci tóniny jsme provedli ve dvou krocích. V prvním kroku jsme pouze pro kontrolu správnosti implementace algoritmus otestovali na všech 24 diatonických stupnicích. Algoritmus detekoval pro hodnoty parametru *penalty* (P) rovné 0,12 a vyšší všechny správně (viz obr. 7.9).

Pro rozsáhlejší testování na reálných hudebních skladbách jsme po vzoru Temperleye (1999) použili fugy ze sbírky *Dobře temperovaný klavír* od J. S. Bacha (1685). Tato sbírka čítá 48 MIDI souborů a protože obsahuje dvě skladby pro každou tóninu, je vhodná pro testování detekce tóniny. Výsledky testování na všech 48 souborech lze vidět v závislosti na parametru P v tabulce 7.10. Od hodnoty P=17 již algoritmus s přehledem rozpozná pro 96/102 skladeb správnou tóninu ve všech segmentech skladby.

P	OK	modOK	Changes	distSD
0	37	37	16599	1,9
1	96	97	377	1,2
2	99	100	106	0,94
3	100	101	51	1
4	96	99	19	1
5	96	99	8	1
6	96	99	6	1
7	96	99	3	1
8	96	99	3	1
9	96	99	3	1
10	96	99	3	1
11	96	99	2	1
13	96	99	1	1
15	96	99	1	1
17	96	99	0	1
19	96	99	0	1

Tabulka 7.10: Výsledky testování detekce tóniny na datasetu WTCII

7.4 Shrnutí

V této kapitole byl popsán způsob, jak při tvorbě transkripčního systému zvolit vhodné algoritmy. Řešení v této práci využívá speciálně navržený testovací framework, pomocí kterého je možné algoritmy odladit a otestovat na různých vstupech dle požadavků výsledného systému. Pro usnadnění návrhu nových algoritmů a přidání nových datasetů do frameworku zde bylo stručně popsáno rozhraní pomocných tříd z naší implementace. Dále jsme zde popsali pomocné nástroje, které jsou součástí testovacího frameworku. V další části této kapitoly jsme si definovali vhodné metriky pro volby algoritmů do cílové aplikace. Algoritmy i způsob testování těchto metrik byl poté v testovacím frameworku implementován.

V poslední části kapitoly se věnujeme výsledkům testování algoritmů. Na jeho základě lze celkem jednoznačně vybrat algoritmy pro všechny podproblémy hudební transkripce. Při detekci výšky dopadl na testovaných metrikách nejlépe algoritmus SNAC. Kromě nejmenší hrubé chyby výšky dosahuje velmi vyrovnaných výsledků při rozpoznávání jemných změn a při volbě správné oktávy.

Pokud bychom chtěli při implementaci zvolit doplňující algoritmus např. pro usnadnění volby oktávy, který má jiné statistické parametry, kepstrální analýza se jeví jako nejlepší volba. Disponuje nejlepší přesností volby oktávy a dosahuje velmi vyrovnaných výsledků v celém detekovaném výškovém rozsahu.

Pro algoritmy detekce znělosti není volba tak snadná. Úspěšnost se zdá být velmi závislá na parametrech algoritmu i na typu nahrávky. Napříč metrikami poskytuje nejvyrovnanější výsledky algoritmus EZCR či NLFER. Při detekci nástupů not je jednoznačně nejlepší detekční funkcí HFC. Dosahuje vyrovnaných výsledků pro nástupy i uvolnění not. Jelikož je však detekce not naprosto klíčová funkce pro hudební transkripci, u které se těžko toleruje i malá chyba, považujeme výsledek 72 % u detekce nástupů not na metrice TPR silně nedostatečnou. Bohužel již není v časových ani rozsahových možnostech této práce tento problém vyřešit, a proto jej necháváme otevřený.

Algoritmů vysokoúrovňové analýzy jsme netestovali více ze dvou důvodů. Jednak jsme při rešerši neobjevili mnoho kandidátů, kteří by mohli výrazně konkurovat námi zvoleným a to z důvodu, že je vysokoúrovňová analýza transkripce velmi specializovaný problém, který není v odborné literatuře příliš řešen. Druhým důvodem je příliš velká rozsáhlost těchto řešení. Pro problém detekce rytmu jsme proto implementovali Rosenthalův (1992) algoritmus, který jsme výrazně upravili tak, aby mohl fungovat v reálném čase. Při testování úspěšně detekoval téměř všechny typy rytmů na testovaném datasetu.

Pro detekci tóniny byl zvolen algoritmus Temperleye (1999), který využívá dynamické programování pro ohodnocení segmentů skladby typem tóniny. Toto řešení jsme otestovali na rozsáhlé sbírce J. S. Bacha *Dobře temperovaný klavír*. Algoritmus zde úspěšně detekoval tóninu téměř všech skladeb v datasetu.

Závěr

Tato práce mapuje problematiku automatické hudební transkripce. Jejím cílem bylo nalézt, implementovat a otestovat algoritmy použitelné pro přepis monofonního hudebního signálu do notového zápisu v reálném čase. Pro rozsáhlost tohoto problému jsme jej rozdělili na čtyři podproblémy a věnovali se každému odděleně.

Nejvíce prostoru věnujeme problému detekce výšky. Pro jeho řešení zde bylo popsáno a implementováno jedenáct různých algoritmů pro generování detekčních funkcí a dalších pět jednoduchých algoritmů pro volbu jejích vrcholů. Hlavní problémy, se kterými jsme se potýkali, byly volba kompromisu mezi přesností v časové a frekvenční doméně a oktávové chyby. Tyto problémy jsme se mimo jiné pokoušeli řešit heuristickými metodami volby vrcholů detekčních funkcí.

Problém detekce not jsme si rozdělili na tři fáze. V první fázi byla detekována znělost signálu, pro kterou byly nalezeny čtyři různé algoritmy. Druhá fáze sestává z generování detekčních funkcí, kterých jsme nalezly též čtyři různé typy. V poslední fázi je naším úkolem volba maxima detekční funkce a pro kterou jsme implementovali šest různých algoritmů. Hlavní komplikace, které nastaly při řešení problému detekce not, vychází z toho, že sluchové vjemy přijímáme pouze relativně vůči ostatním vjemům, zatímco algoritmy pracují s absolutními hodnotami. Řešením této komplikace byl návrh adaptivních metod volby maxima detekčních funkcí.

Vysokoúrovňovou hudební analýzu v této práci rozdělujeme na detekci rytmu a tóniny. Pro detekci rytmu jsme implementovali rozsáhlejší systém, který funguje na principu paprskového vyhledávání, pomocí kterého prohledává prostor různých rytmických interpretací. Při návrhu tohoto systému jsme se potýkali především s problémem *půlení* či *zdvojnásobení* rytmu a obtížnostmi, které plynou z tolerance lidského vnímání vůči časovým nepřesnostem. Tato tolerance se těžko modeluje, problém však byl řešen metodou histogramu vzdáleností mezi nástupy jednotlivých not.

Pro detekci tóniny jsme implementovali řešení založené na preferenčních pravidlech a implementovali jej pomocí dynamického programování. Během řešení tohoto problému se objevila komplikace při rozpoznávání mezi krátkodobou změnou tonálního centra a trvalou změnou tóniny. Řešením bylo zavedení parametru penalty, jež byla udělována za časté změny tóniny při interpretaci skladby.

Pro otestování vybraných algoritmů bylo naším úkolem navrhnout metriky, které by pomohly z navrhovaných algoritmů zvolit ty, které by byly nejvhodnější pro použití v mobilní aplikaci pro iOS, provádějící monofonní hudební transkripci. Tyto metriky, kromě důrazu na obecnou úspěšnost algoritmů, omezovaly výběr algoritmů z hlediska jejich časové složitosti. Toto omezení bylo stanoveno z důvodu aplikace algoritmů v reálném čase.

Pro snazší návrh, ladění a testování algoritmů jsme vytvořili testovací framework, který poskytuje snadné rozhraní pro návrh nových algoritmů, vizualizaci jejich výstupů a testování na zmíněných metrikách.

Na základě testování jsme z popsaných algoritmů zvolili pro každý podproblém jeden algoritmus, který navrhujeme pro použití ve výsledné aplikaci. Předpokladem pro funkčnost aplikace je však vyřešení problému detekce not s mnohem menší chybou, než jaké algoritmus dosáhl při testování. Rozsah práce a především časové možnosti nám již nedovolily přijít s lepším řešením tohoto problému, a proto zde není představena výsledná aplikace. V příloze však nalezneme všechny zmiňované algoritmy, které jsou implementovány tak, aby je bylo možné snadno použít pro vývoj transkripční aplikace.

Otevřené problémy

Hlavním otevřeným problémem zjevně zůstává problém detekce nástupů not. Ačkoliv se v literatuře problém monofonní transkripce běžně označuje jako vyřešený (Klapuri, 1997), Clarisse (2002) upozorňuje, že právě segmentace vstupu do not je i za nejlepších současných prostředků proces, který je velmi náchylný k chybám. Ani soubor zde vybraných algoritmů pro detekci výšky neobsahuje všechny současně uznávané metody pro řešení tohoto problému. Z časových důvodů v této práci nebyl implementován např. algoritmus Konstantní Q transformace (Brown, 1992a), či metoda maximální věrohodnosti (Mahadevan et al., 2011). Bylo by zajímavé vidět, jak si vedou ve srovnání s algoritmy implementovanými zde. Harmonie hraje velkou roli při rytmické interpretaci skladby, a proto se nabízí propojení analýzy tóniny a rytmu, které by mohlo zlepšit její úspěšnost. Rytmický analyzátor by též mohl v budoucnu využívat informace o melodických motivech, které jsou pro vnímání rytmu též důležité. Také nebyly v této práci zmíněny žádné algoritmy využívající strojové učení, ačkoliv v tomto směru již existuje rozsáhlý výzkum.

Přínos práce

Za hlavní přínos této práce považujeme důkladné zmapování velkého rozsahu algoritmů pro řešení automatické hudební transkripce. Jelikož je hudební transkripce problém závislý na své aplikaci, a různé metody jeho řešení dosahují za různých okolností odlišné úspěšnosti, případný řešitel tohoto problému může tuto práci využít jako základního průvodce při návrhu transkripčního systému na míru své aplikace. S tím souvisí druhý hlavní přínos této práce, kterým je testovací framework, vytvořený pro pohodlné navrhování algoritmů pro transkripční systém a jeho čtyři hlavní podproblémy.

Dodatek A

Obsah přiloženého CD

/
- readme.txt ..... stručný popis obsahu CD
- src
  - imp ..... zdrojové kódy implementace
  - thesis ..... bakalářská práce ve zdrojovém formátu LaTeX
- text
  - thesis.pdf ..... text práce ve formátu PDF

Seznam použitých zkratek

ACF Autocorrelation function (Autokorelační funkce)

ADSR Attack (nástup), Decay (útlum), Sustain (podržení), Release (uvolnění).

AMDF Average Magnitude Difference Function (Funkce průměru rozdílů amplitud, zkráceně rozdílová funkce)

BPM Beats per minute (Počet dob za minutu)

CMNDF Cumulative mean normalized difference function (Rozdílová funkce normalizovaná průměrem)

DFT Discrete Fourier transform (Diskrétní Fourierova transformace)

DTFT Discrete time Fourier transform (Fourierova transformace diskrétní v čase)

FFT Fast Fourier Transform (Rychlá Fourierova transformace)

FNR False Negative Rate (Míra chyb)

FPE Fine Pitch Error (Jemná chyba detekce výšky)

FS Fourier series (Fourierova řada)

GPE Gross Pitch Error (Hrubá chyba detekce výšky)

HPS Harmonic Product Spectrum (Spektrální harmonický součin)

IDFT Inverse Discrete Fourier transform (Inverzní diskrétní Fourierova transformace)

LPM Landmark Points Method (Algoritmus mezních bodů)

MIDI Musical Instrument Digital Interface

NCCF Normalized cross-correlation function (Normalizovaná korelační funkce)

PDA Pitch Detection Algorithm (Algoritmy detekce výšky)

SHR Subharmonic-to-Harmonic Ratio (Poměr subharmonických a harmonických složek)

SMF Standart Midi File (Standartní Midi soubor)

SNAC Specially-normalised autocorrelation function (Speciálně normalizovaná autokorelační funkce)

TPR True Positive Rate (Citlivost)

WACF Weighted autocorrelation function (Vážená autokorelační funkce)

Bibliografie

ALLEN, Paul E; DANNENBERG, Roger B, 1990. Tracking Musical Beats in Real Time. In: *Tracking Musical Beats in Real Time. ICMC*.

ASKENFELT, A, 1976. Automatic notation of played music (status report). *Prog, and Status Rep., Speech Transmission Laboratory, Royal Inst, of Technol., Stockholm, l-ll.*

Automatic musical meter detection, 2009. IEEE, International Conference on Acoustics, Speech a Signal Processing.

BABACAN, Onur; DRUGMAN, Thomas; D'ALESSANDRO, Nicolas; HENRICH, Nathalie; DUTOIT, Thierry, 2013. A comparative study of pitch extraction algorithms on a large variety of singing sounds. In: *A comparative study of pitch extraction algorithms on a large variety of singing sounds. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, s. 7815–7819.

BAHJA, F.; MARTINO, J. Di; ELHAJ, E. H. I., 2010. Real-time pitch tracking using the eCATE algorithm. In: *Real-time pitch tracking using the eCATE algorithm. 2010 5th International Symposium On I/V Communications and Mobile Network*. Dostupné z DOI: 10.1109/ISVC.2010.5656254.

BACH 1685-1750, Johann Sebastian. *Bach's well-tempered clavier: 48 preludes and fugues for piano* [London : Music Sales ; New York : Amsco Music Pub. Co., ©1972].

BACH, Johann Sebastian, 1893a. *Ach Gott und Herr, BWV 714*. Salzburg. Chorale Preludes, BWV 714-765.

BACH, Johann Sebastian, 1893b. *Ach, lieben Christen, seid getrost, BWV 114*. Salzburg. Chorale Preludes.

BACHU, RG; KOPPARTHI, S; ADAPA, B; BARKANA, BD, 2008. Separation of voiced and unvoiced using zero crossing rate and energy of the speech signal. In: *Separation of voiced and unvoiced using zero crossing rate and energy of the speech signal. American Society for Engineering Education (ASEE) Zone Conference Proceedings*, s. 1–7.

BELLO, Juan Pablo; DAUDET, Laurent; ABDALLAH, Samer; DUXBURY, Chris; DAVIES, Mike; S, Mark B.; MEMBER, Senior, 2004. A tutorial on onset detection in music signals. In: *A tutorial on onset detection in music signals. IEEE Transactions in Speech and Audio Processing*.

BELLO, Juan Pablo; MONTI, Giuliano; SANDLER, Mark B, 2000. Techniques for Automatic Music Transcription. *ISMIR 2000*.

BENGTSSON, Ingmar; GABRIELSSON, Alf; (SWEDEN), Kungl. Musikaliska akademien, 1977. *Rhythm research in Uppsala / by Ingmar Bengtsson and Alf Gabrielsson*. [Musical Academ] Stockholm.

BERÁNKOVÁ, Petra, 2017. Tonální a harmonické cítění jako základ tvořivého projevu v hudební výchově.

BROWN, Judith C, 1992a. An efficient algorithm for the calculation of a constant Q transform. *The Journal of the Acoustical Society of America*. Roč. 92, č. 5, s. 2698–2701.

BROWN, Judith C, 1992b. Determination of musical meter using the method of autocorrelation. *The Journal of the Acoustical Society of America*. Roč. 91, č. 4, s. 2374–2375.

BRUDERHOFER, Norman, 2005. *Edison cylinder phonograph*. Dostupné také z: https : / / en . wikipedia . org / wiki / Phonograph # /media / File : EdisonPhonograph.jpg. Navštíveno: 2017-04-11.

BYRD, Donald, 2017. *Extremes of Conventional Music Notation*. Dostupné také z: http://homes.soic.indiana.edu/donbyrd/CMNExtremesBody.htm. Navštíveno: 2017-04-05.

CANNAM, C.; LANDONE, C.; SANDLER, M., 2010. Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files. In: *Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files. Proceedings of the ACM Multimedia 2010 International Conference*. Firenze, Italy, s. 1467–1468.

CASTANIE, Francis, 2006. *Spectral Analysis: Parametric and Non-Parametric Digital Methods (Digital Signal and Image Processing Series)*. ISTE. ISBN 1905209053.

CLARISSE, L. P.; MARTENS, J. P.; LESAFFRE, M.; BAETS, B. De; DEMEYER, H.; LEMAN, M., 2002. An auditory model based transcriber of singing sequences. In: *An auditory model based transcriber of singing sequences. in ISMIR*, s. 116–123.

COLER, Henrik von; LERCH, Alexander, 2014. CMMSD: A Data Set for Note-Level Segmentation of Monophonic Music. In: *CMMSD: A Data Set for Note-Level Segmentation of Monophonic Music. Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*.

COOLEY, James W.; TUKEY, John W., 1965. An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation*. Roč. 19, č. 90, s. 297–301. ISSN 00255718, 10886842. ISSN 00255718, 10886842. Dostupné také z: http://www.jstor.org/stable/2003354.

CORDIER, Baude, 2017. *Belle, bonne, sage*. Dostupné také z: https://commons.wikimedia.org/wiki/File:CordierColor.jpg. Navštíveno: 2017-04-05.

COUCH, Alex, 2017. *How I'd Redesign Piano Sheet Music – Alex Couch's portfolio – Medium*. Dostupné také z: https://medium.com/alex-couch-s-portfolio/how-i-d-redesign-piano-sheet-music-355c4f9012f1.

DE LA CUADRA, P; MASTER, A, 2001. Efficient pitch detection techniques for interactive music. In: *Efficient pitch detection techniques for interactive music. In Proceedings of the 2001 International Computer Music Conference, La Habana*, s. 1–3.

DEUTSCH, Diana, 1986. A Musical Paradox. *Music Perception: An Interdisciplinary Journal*. Roč. 3, č. 3, s. 275–280.

DONADIO, Matt, 1999. *How to Interpolate the Peak Location of a DFT or FFT if the Frequency of Interest is Between Bins - dspGuru*. Dostupné také z: https://dspguru.com/dsp/howtos/how-to-interpolate-fft-peak/. Navštíveno: 2017-04-05.

DUAN, Zhiyao; PARDO, Bryan; ZHANG, Changshui, 2010. Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE Trans. Audio Speech Language Process.* Roč. 18, č. 8, s. 2121–2133.

DUCKWORTH, W., 2012. *A Creative Approach to Music Fundamentals*. Cengage Learning. ISBN 9780840029997. Dostupné také z: https://books.google.cz/books?id=-bmJtzhOyu4C.

GASIOR, M; GONZALEZ, J L, 2004. Improving FFT Frequency Measurement Resolution by Parabolic and Gaussian Spectrum Interpolation. *AIP Conference Proceedings*. Roč. 732, s. 276–285.

GOUYON, Fabien; KLAPURI, Anssi; DIXON, Simon; ALONSO, Miguel; TZANETAKIS, George; UHLE, Christian; CANO, Pedro, 2006. An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech, and Language Processing*. Roč. 14, č. 5, s. 1832–1844.

HAINSWORTH, Stephen Webley, 2004. *Techniques for the Automated Analysis of Musical Audio*.

HESS, W., 1983. *Pitch determination of speech signals: algorithms and devices*. Springer-Verlag. Springer series in information sciences. ISBN 9783540119333. Dostupné také z: http://books.google.cz/books?id=PWlTAAAAMAAJ.

HURON, David; PARNCUTT, Richard, 1993. An improved model of tonality perception incorporating pitch salience and echoic memory. *Psychomusicology: A Journal of Research in Music Cognition*. Roč. 12, č. 2, s. 154.

CHEVEIGNÉ, A, 2008. *Pitch Perception Models. Pitch pp. 169–233*.

CHEVEIGNÉ, Alain de, 2002. YIN, a fundamental frequency estimator for speech and music. *Acoustical Society of America Journal*. Roč. 111, č. 4, s. 1917–1930.

JONES, Richard, 2017. *Seebeck vs. Ohm*. Dostupné také z: http://wtt.pauken.org/?page_id=1630.

KASI, KAVITA; ZAHORIAN, STEPHEN A., 2002. *Yet Another Algorithm for Pitch Tracking*. IEEE.

KITAHARA, Tetsuro, 2010. Mid-level Representations of Musical Audio Signals for Music Information Retrieval. In: *Advances in Music Information Retrieval*. Ed. RAŚ, Zbigniew W.; WIECZORKOWSKA, Alicja A. Berlin, Heidelberg: Springer Berlin Heidelberg, s. 65–91. ISBN 978-3-642-11674-2. Dostupné z DOI: 10.1007/978-3-642-11674-2_4.

KLAPURI, Anssi, 1997. *Automatic Transcription of Music*.

KLAPURI, Anssi, 2004. Automatic Music Transcription As We Know It Today. *Journal of New Music Research*. Roč. 33, č. 3, s. 269–282.

KLAPURI, Anssi, 2006. *Signal Processing Methods for Music Transcription*. Ed. KLAPURI, Anssi; DAVY, Manuel. Boston, MA: Springer US.

KLEJCHOVÁ, Mgr. Martina, 2008. *Rychlá Fourierova transformace*. Praha.

KRAEMER, Brandy, 2016. *Multilingual Tempo Commands and BPM*. Dostupné také z: https://www.thoughtco.com/multilingual-tempo-commands-and-bpm-2701376. Navštíveno: 2017-04-05.

LANZ, H., 1931. *The Physical Basis of Rime: An Essay on the Aesthetics of Sound*. Stanford University Press. ISBN 9780804731713. Dostupné také z: https://books.google.cz/books?id=woGaAAAAIAAJ.

LINKWARE-GRAPHICS, 2015. *Circle of Fifths - Free Printable Music Theory*. Dostupné také z: http://linkwaregraphics.com/music/circle-of-fifths/. Navštíveno: 2017-04-15.

MAHADEVAN, Vijay; ESPY-WILSON, Carol Y, 2011. Maximum likelihood pitch estimation using sinusoidal modeling. In: *Maximum likelihood pitch estimation using sinusoidal modeling. 2011 International Conference on Communications and Signal Processing (ICCSP)*. IEEE, s. 310–314.

MASNICK, Mike, 2017. *Massive Growth In Independent Musicians & Singers Over The Past Decade*. Dostupné také z: https://www.techdirt.com/blog/casestudies/articles/20130529/15560423243/massive-growth-independent-musicians-singers-over-past-decade.shtml.

MCLEOD PH.D., Philip; WYVILL, G, 2003. Visualization of musical pitch. In: *Visualization of musical pitch. Computer Graphics International 2003*. IEEE Comput. Soc, s. 300–303.

MCLEOD, Philip, 2009. Fast, accurate pitch detection tools for music analysis.

METFESSEL, Milton Franklin; (1793-1962)., University of North Carolina, 1928. *Phonophotography in folk music; American negro songs in new notation*. Chapel Hill: The University of North Carolina press. Dostupné také z: http://hdl.handle.net/2027/uc1.32106010326780. Bibliography: p. 179-180.

MILLER, Dayton Clarence, 1916. *The science of musical sounds*. New York, The Macmillan company. Dostupné také z: https://archive.org/details/sciencemusicals01millgoog.

MOORER, J., 1974. The optimum comb method of pitch period analysis of continuous digitized speech. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. Roč. 22, č. 5, s. 330–338. ISSN 0096-3518. Dostupné z DOI: 10.1109/TASSP.1974.1162596.

MOORER, James A., 1975. *On the Segmentation and Analysis of Continuous Musical Sound by Digital Computer*. Stanford, CA. Dostupné také z: https://ccrma.stanford.edu/files/papers/stanm3.pdf. Diplomová práce. Stanford University.

MULLER, Meinard; ELLIS, Daniel PW; KLAPURI, Anssi; RICHARD, Gaël, 2011. Signal processing for music analysis. *IEEE Journal of Selected Topics in Signal Processing*. Roč. 5, č. 6, s. 1088–1110.

NIELSEN, Michael, 2016. Toward an exploratory medium for mathematics. *Cognitive medium*. Dostupné také z: http://cognitivemedium.com/emm/emm.html. Navštíveno: 2017-04-03.

NOLL, A Michael, 1967. Cepstrum pitch determination. *The journal of the acoustical society of America*. Roč. 41, č. 2, s. 293–309.

NONKEN, M., 2014. *The Spectral Piano: From Liszt, Scriabin, and Debussy to the Digital Age*. Cambridge University Press. Music Since 1900. ISBN 9781139916110. Dostupné také z: https://books.google.cz/books?id=SHY9AwAAQBAJ.

ONDŘÍČKOVÁ, Marie, 2012. *Hudba starověkého Blízkého východu*. Plzeň.

PAPENBURG, Hagen, 2017. *Flötenetüde*. Dostupné také z: http://musescore.com/user/12620401/scores/3684516. Navštíveno: 2017-04-05.

PATERSON, Jim, 2015. A Short History of Musical Notation. Dostupné také z: http://www.mfiles.co.uk/music-notation-history.htm. Navštíveno: 2017-04-03.

PHILHARMONIA ORCHESTRA, Members of, 2017. *Philharmonia Orchestra Sound Samples*. Dostupné také z: http://www.philharmonia.co.uk/explore/sound_samples Navštíveno: 2017-04-03.

Phonautograph, 2015. Dostupné také z: http://cylinders.library.ucsb.edu/history-early.php. Navštíveno: 2017-04-11.

PISZCZALSKI, Martin; GALLER, Bernard A, 1982. A Computer Model Of Music Recognition. In: *A Computer Model Of Music Recognition. Music, Mind, and Brain*. Boston, MA: Springer US, s. 399–416.

RABINER, L.; CHENG, M.; ROSENBERG, A.; MCGONEGAL, C., 1976. A comparative performance study of several pitch detection algorithms. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. Roč. 24, č. 5, s. 399–418. ISSN 0096-3518. Dostupné z DOI: 10.1109/TASSP.1976.1162846.

RINALDI, C.; GUSTAVINO, G.; SANTIC, M.; POMANTE, L.; PENNESE, M., 2015. Comparison of pitch detection algorithms for the Crazy Square project. In: *Comparison of pitch detection algorithms for the Crazy Square project. 2015 International Symposium on Signals, Circuits and Systems (ISSCS)*, s. 1–4. Dostupné z DOI: 10.1109/ISSCS.2015.7203969.

ROSENTHAL, David Felix, 1992. *Machine Rhythm: Computer Emulation of Human Rhythm Perception*. Cambridge, MA, USA: Massachusetts Institute of Technology. Disertační práce. Not available from Univ. Microfilms Int.

RYYNÄNEN, Matti, 2008. *Automatic Transcription of Pitch Content in Music and Selected Applications*.

SANGSIN, Na, 2002. *Introduction to Signals Course – Notes*. Dostupné také z: http://www.eecs.umich.edu/courses/eecs206/archive/spring02/.

SENGPIEL, Eberhard, 1993. *Frequenzbereich von Musikinstrumenten, Gesangsstimmen und Keyboards*. Dostupné také z: http://www.sengpielaudio.com/FrequenzbereichMusikinstrumente.pdf.

SHEPARD, Roger N., 1964. Circularity in Judgments of Relative Pitch. *The Journal of the Acoustical Society of America*. Roč. 36, č. 12, s. 2346–2353. Dostupné z DOI: 10.1121/1.1919362.

SHIMAMURA, T.; KOBAYASHI, H., 2001. Weighted autocorrelation for pitch extraction of noisy speech. *IEEE Transactions on Speech and Audio Processing*. Roč. 9, č. 7, s. 727–730. ISSN 1063-6676. Dostupné z DOI: 10.1109/89.952490.

SMITH, Julius O., 2011. *Spectral Audio Signal Processing*. Dostupné také z: http://ccrma.stanford.edu/~jos/sasp/. online book, 2011 edition, accessed 5/4/2017.

SMITH, Steven W., c1997. *The scientist and engineer's guide to digital signal processing*. 1st ed. San Diego, Calif.: California Technical Pub. ISBN 978-096-6017-632.

SOKOL, Jan, 2004. *Čas a rytmus*. 2., rozš. vyd. Praha: Oikoymenh. ISBN 8072981234.

SUN, X., 2002. Pitch determination and voice quality analysis using Subharmonic-to-Harmonic Ratio. In: *Pitch determination and voice quality analysis using Subharmonic-to-Harmonic Ratio. 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Sv. 1. ISSN 1520-6149. Dostupné z DOI: 10.1109/ICASSP.2002.5743722.

SYROVÝ, Václav, 2013. *Hudební akustika*. 3., dopl. vyd. V Praze: Akademie múzických umění. ISBN 978-80-7331-297-8.

ŠMIDÁK, Miron, 2005. *Absolutní sluch*. Praha.

ŠRÁMKOVÁ, Kateřina, 2016. *Problematika transkripce v hudebních skladbách [online]*. Disertační práce. Masarykova univerzita, Pedagogická fakulta, Brno. Vedoucí práce Michal KOŠUT.

TALKIN, D., 1995. A robust algorithm for pitch tracking (RAPT). In: KLEIN, W. B.; PALIVAL, K. K. (ed.). *Speech Coding and Synthesis*. Elsevier.

TEMPERLEY, David, 1999. What's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered. *Music Perception: An Interdisciplinary Journal*. Roč. 17, č. 1, s. 65–100.

TEMPERLEY, David, 2000. The line of fifths. *Music Analysis*. Roč. 19, č. 3, s. 289–319.

TEMPERLEY, David, 2002. A Bayesian approach to key-finding. In: *A Bayesian approach to key-finding. Music and artificial intelligence*. Springer, s. 195–206.

TEMPERLEY, David; SLEATOR, Daniel Dominic, 1999. Modeling Meter and Harmony - A Preference-Rule Approach. *Computer Music Journal*.

VASS, Jiří, 2004. *Automatic transcription of audio signals*. Prague, Czech Republic.

VINET, Hugues, 2004. The Representation Levels of Music Information. In: *Computer Music Modeling and Retrieval: International Symposium, CMMR 2003, Montpellier, France, May 26-27, 2003. Revised Papers*. Ed. WIIL, Uffe Kock. Berlin, Heidelberg: Springer Berlin Heidelberg, s. 193–209. ISBN 978-3-540-39900-1. Dostupné z DOI: 10.1007/978-3-540-39900-1_17. Navštíveno: 2017-04-03.

WALMSLEY, P.J., 2001. *Signal Separation of Musical Instruments: Simulation-based Methods for Musical Signal Decomposition and Transcription*. University of Cambridge. Dostupné také z: https://books.google.cz/books?id=kFNUoAEACAAJ.

WEISSTEIN, Eric W., 2006. *Wiener-Khinchin Theorem*. MathWorld–A Wolfram Web Resource. Dostupné také z: http://mathworld.wolfram.com/Wiener-KhinchinTheorem.html.

WEST, Blake, 2014. *Hummingbird Notation*. Dostupné také z: http://www.hummingbirdnotation.com/images/hummingbird-sample(big).png. Navštíveno: 2017-04-05.