

Debreceni Egyetem
Informatikai Kar

Hangjelek analízise

Témavezető:
Dr. Papp Zoltán
egyetemi adjunktus

Készítette:
Hankószky Enikő
programtervező informatikus

Debrecen
2011

Tartalomjegyzék

Bevezetés	4
A digitális jelfeldolgozás alapjai	5
Mintavételezés	5
Kvantálás	6
Fourier-transzformáció	7
Konvolválás	8
Diszkrét Fourier-transzformáció (DFT)	9
Gyors Fourier-transzformáció (FFT)	9
Hangmagasság	10
Inger- és érzet-oldali magassági skálák	10
Ingeroldali skálák:	10
A frekvencia skálája	10
A zenei hangköz	11
Érzetoldali skála	11
A melodikus hangmagasság skálája	11
Hangmagassági modellek	12
Egy csúcs-völgy becslő metóduson és egy adaptálható szűrőn alapuló modell.....	12
Harmonikus szorzat spektrum (HPS)	14
Maximum Likelihood (MA)	16
Kepsztrum.....	17
Autokorreláció (ACF).....	17
Átlagos magnitúdó-különbség függvény (AMDF)	19
Zérus-átmenet detektálás	20
YIN algoritmus (Alain de Cheveigné és Hideki Kawahara)	20
1. lépés: Az autokorrelációs módszer	20
2. lépés: Különbség függvény	21
3. lépés: Halmozott átlag normalizált különbségfüggvény	21
4. lépés: Abszolút küszöb	22
5. lépés: Parabolikus interpoláció	22
6. lépés: Legjobb lokális becslés	22
Hangosság.....	23
Kritikus hallási sávok	23

Az egyenlő hangosság szintek görbéi.....	24
A valódi hangosság.....	25
Az összhangosság	26
Hangossági modell implementáció.....	27
Lineáris prediktív kódolás (LPC)	30
Az LPC modell	30
Ablakozás	31
Legkisebb négyzetes autokorrelációs módszer.....	32
Legkisebb négyzetes kovariancia módszer.....	33
Számítási megfontolások.....	34
Példa a felhasználásra.....	35
MIDI Toolbox	36
MIDI fájl beolvasása Matlabba	36
Hivatkozás a mátrix egyes részeire	37
Hangjegymátrixok manipulálása	37
MIDI fájlok elmentése.....	38
A hangjegymátrixok lejátszása.....	38
MIDI adatok megjelenítése	38
Lilypond	43
A Lilypond alapvető funkcióinak bemutatása.....	43
Hangmagasság	43
Ritmusértékek.....	44
Módosítójelek	45
Ívek.....	45
Dinamika és egyéb jelölések	46
Abszolút hangmagasság-megadás	47
Automatikus és kézi gerendázás.....	47
Transzponálás	47
Az elkészített program bemutatása.....	49
Összefoglalás	56
Köszönetnyilvánítás	57
Irodalomjegyzék	58

Bevezetés

Napjainkban a technika abba az irányba fejlődik, hogy a digitális beszéd- és hangfeldolgozás egy nagyon jelentős és új lehetőségeket rejtő tudományterületként kerül előtérbe. Egyre inkább elvárás a számítógépekkel kapcsolatban, hogy képesek legyenek a műveletek hangvezérlés alapján történő végrehajtására, valamint az ugyanilyen formájú válaszadásra is. Felmerült annak is az igénye, hogy a felvett zeneműveket képesek legyünk számítógépes programok segítségével elemezni, javítani és módosítani.

Több éven keresztül foglalkoztam komolyzenével, és amikor kiderült, hogy a zene és az informatika összekapcsolható, tudtam, hogy ez a terület az, amivel szívesen foglalkoznék. Ezért választottam szakdolgozatom témájaként a hangjelek analízisét. A dolgozat megírásakor sok cikket olvastam a témában, amelyek közül elég jelentős mennyiségű került ki kínai egyetemekről. Több olyan cikket is találtam ezek között, ahol csak egy adott hangszeren – néha tradicionális hangszeren – lejátszott műveket elemeztek. Innen jött az ötlet, hogy én is egyetlen hangszerrel foglalkozzak. A választásom a gordonkára esett, mert nemcsak gyönyörű hangja van, hanem vonós hangszer lévén, a rajta játszott művek feldolgozása is egy izgalmas kihívás.

A szakdolgozatom megírásakor az elsődleges célom egy program elkészítése volt, amely képes egy csellón lejátszott mű elemzésére. Másodsorban egy átfogó elméleti áttekintést készítettem a hangmagasság meghatározásának témakörében.

A dolgozatom elején egy rövid áttekintést nyújtok a digitális jelfeldolgozásnál felmerülő alapfogalmakról. A további fejezetekben pedig szemléltetek néhány hangmagassági modellt, többek között a harmonikus szorzat spektrumot, az autokorrelációs módszert és az AMDF algoritmust. Egy rövid áttekintést adok a hangosságról, annak fogalmairól és az érzetoldali skálákról. Bemutatok egy kottaíró programot – amiben szöveges utasítások segítségével igényes megjelenésű kották készíthetők –, valamint egy MIDI fájlokat feldolgozó függvénycsomagot. Végül részletezem az elkészült program működését képekkel illusztrálva. A program Matlab környezetben készült, mert ez a matematikai szoftvercsomag nagyfokú támogatottsággal rendelkezik a hangfeldolgozás területén.

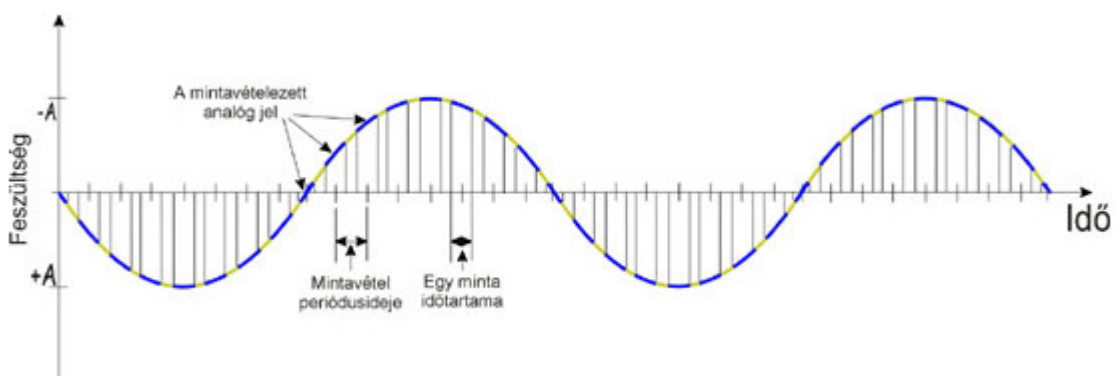
A digitális jelfeldolgozás alapjai

Egy jel értelmezési tartománya mind az időkoordináta mentén, mind az amplitúdóban lehet folytonos vagy diszkrét. Az időben is és amplitúdóban is folytonos értékészletű jelet analóg jeleknek nevezzük. Analóg jelet a számítógépek nem képesek feldolgozni, ezért van szükség ezeknek a jeleknek a digitalizálására. A digitalizált jel már amplitúdóban véges értékészletű, időben pedig vagy véges, vagy megszámlálhatóan végtelen értékészletű lesz. Az analóg jelek digitalizálásának két lépése van: mintavételezés és kvantálás.

Mintavételezés

Mintavételezésnek nevezzük azt a folyamatot, amikor az analóg jelet időben diszkrété tesszük. Tehát a folytonos jelből meghatározott T_0 időközönként értéket vagy más néven mintát veszünk. A gyakorlatban használt legelterjedtebb mintavételezési technikában a mintavételi időpontok egymást azonos időközönként követik. A mintavételezési frekvencia adja meg, hogy másodpercenként hány mintavétel történik, tehát a mintavételezés gyakoriságát jelenti. Hertzben mérjük, és a mintavételi időköz reciprokával egyenlő.

$$f_s = \frac{1}{T_0}$$



A fenti ábra a mintavételezés folyamatát szemlélteti. A mintavételezéskor felmerül még egy nagyon fontos fogalom, a rekonstrukció. Ez azt jelenti, hogy milyen feltételeknek kell teljesülnie ahhoz, hogy a mintákból egyértelműen visszaállítható legyen az eredeti jel. Nyquist-tétele azt mondja ki, hogy annak elegendő feltétele, hogy egy $x_s(t)$ mintavett jelből

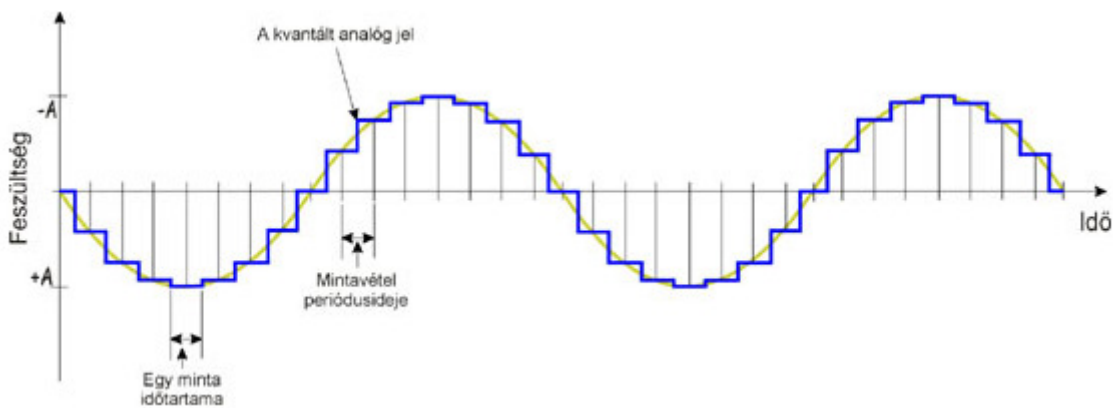
egyértelműen visszaállítható legyen az eredeti $x(t)$ jel az, hogy $X(f)$ – a jel spektruma (ld. később) – tűnjön el, ha $|f| \geq f_s/2$.

Kvantálás

A kvantálás úgynevezett függőleges irányú diszkrétizálást hajt végre. A minták amplitúdójának folytonos értékkészletét alakítja diszkrété úgy, hogy az eredeti értékkészletet kvantálási lépcsőkkel intervallumokra osztja. Minden egyes intervallumban kijelöl egy értéket, az ún. kvantálási szintet. Az adott mintának az eredeti pillanatnyi amplitúdójához azt a kvantálási szintet (kerekített értéket) fogja rendelni, amelyik a pillanatnyi amplitúdóval egy kvantálási lépcsőben van. Nem szükséges, hogy a kvantálási lépcsők azonosak legyenek, és a kvantálási szinteknek sem kell a kvantálási lépcsők közepére esniük. De ha a kvantálási lépcsők mégis azonosak és a kvantálási szintek ezen lépcsők közepére esnek, akkor lineáris kvantálásról beszélünk. Mivel a kvantálási szint általában nem egyezik meg a minta eredeti amplitúdójával, ezért beszélhetünk a kvantálás hibájáról, amit zajnak vagy torzításnak is szoktak nevezni.

$$e(t) = \hat{x}(t) - x(t)$$

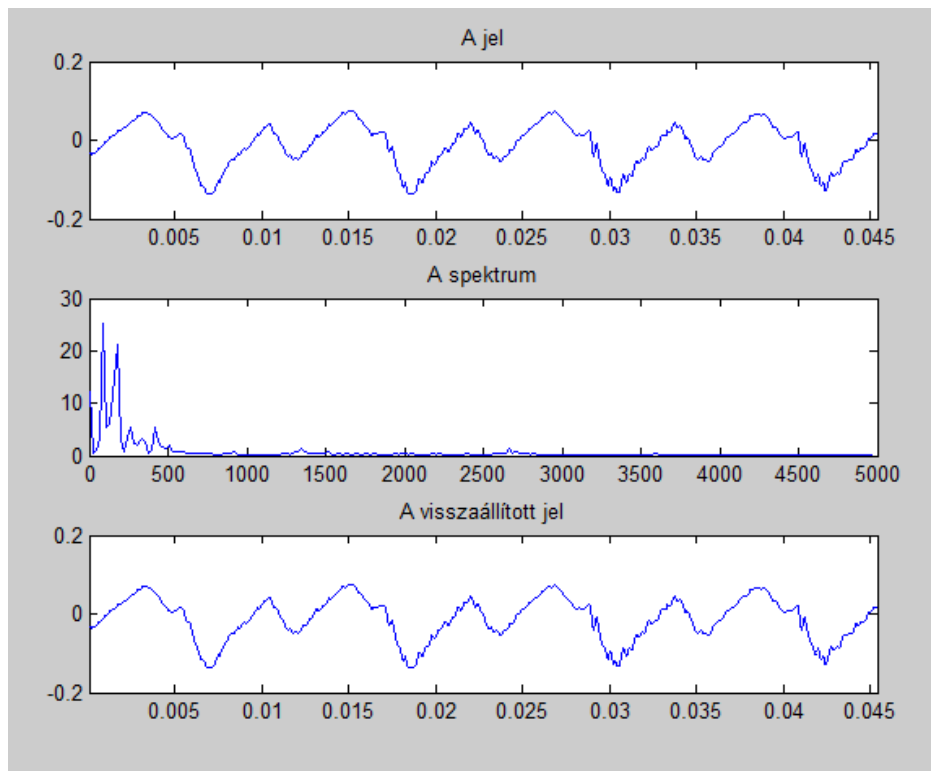
Ezen összefüggés szerint a kvantált jel az eredeti jel és a kvantálási hiba összegére bontható.



Ez az ábra a kvantálás folyamatát szemlélteti. Megkülönböztetünk *nulltartó* és *nullkitérő* kvantálást. Akkor mondjuk azt, hogy a kvantálás *nulltartó*, ha a kvantálás a 0 ponthoz és a környezetéhez a 0 kvantálási szintet rendeli. Ha nincs zérus értékű kvantálási szint, akkor a kvantálást *nullkitérőnek* nevezzük.

Fourier-transzformáció

A jelek többsége egyértelműen felbontható harmonikus jelek összegére. Ha adott egy $x(t)$ időjel, akkor azt a *Fourier-transzformáció* komponensek (szinuszoidok) összegére bontja. Megmutatja, hogy az egyes frekvenciák milyen súllyal szerepelnek a jelben. A jelhez ilyen módon hozzárendelt összetevők frekvenciáit, és amplitúdóit meghatározó adatok összességét a jel spektrumának nevezzük. A Fourier-transzformáció inverz művelete pedig arra szolgál, hogy a frekvencia spektrumból megadja az időtől függő jelet.



Fourier-transzformált:

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-i2\pi ft} dt$$

Maga a jel az $X(f)$ spektrumból az *inverz-Fourier- integrállal* számítható ki:

$$x(t) = \int_{-\infty}^{\infty} X(f)e^{i2\pi ft} df$$

Az $X(f)$ *Fourier-integrál* az $x(t)$ folytonos jelet minden időpillanatban egyértelműen meghatározza.

A Fourier-integrálható $x(t)$ időfüggvényeket egyszeri folyamatoknak szokták nevezni. Ezek fontos jellemzője a Parseval-egyenlőség, mely szerint a jel és a spektrum energiája megegyezik.

$$E = \int_{-\infty}^{\infty} |x(t)|^2 dt = \int_{-\infty}^{\infty} |X(f)|^2 df$$

A Fourier-transzformáció tulajdonsága például a linearitása: $\mathcal{F}(\alpha x + \beta y) = \alpha \mathcal{F}(x) + \beta \mathcal{F}(y)$. Meg kell még említeni a konjugálás és a Fourier-transzformált inverzének kapcsolatát is: $\mathcal{F}^1(X)(t) = \mathcal{F}^*(X^*)(t)$.

Konvolválás

Tegyük fel, hogy f és g integrálható függvények. Az f és g függvények közötti konvolúció műveletét pedig jelölje $*$, ami a következőképpen számítandó:

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

Legyenek f és g integrálható függvények és jelölje $\mathcal{F}$ a Fourier-transzformációt. Ekkor $\mathcal{F}(f * g) = \mathcal{F}(f)\mathcal{F}(g)$, illetve $\mathcal{F}(fg) = \mathcal{F}(f) * \mathcal{F}(g)$. A konvolúció a jelfeldolgozás alapvető művelete. Ez a lineáris művelet két függvényből állít elő egy harmadikat. A konvolúció tulajdonságai megegyeznek a szorzásával:

- Kommutativitás: $f * g = g * f$
- Asszociativitás: $f * (g * h) = (f * g) * h$
- Disztributivitás: $f * (g + h) = f * g + f * h$

ahol f, g és h tetszőleges függvények.

Diszkrét Fourier-transzformáció (DFT)

A diszkrét Fourier-transzformáció egy véges számsorozathoz egy másik véges számsorozatot fog rendelni. Azért van nagy jelentősége, mert a számítógépek csak diszkrét értékekkel tudnak dolgozni. Van egy N elemű számsorozatunk, ami egy időfüggvény mintavételezéséből származó számsorozat: $x(0), x(1), x(2), \dots, x(n), \dots, x(N-1)$. Akkor a diszkrét Fourier-transzformáció ehhez a számsorozathoz egy ugyancsak N elemű $X(0), X(1), X(2), \dots, X(k), \dots, X(N-1)$ számsorozatot rendel a következő módon:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-ink \frac{2\pi}{N}}, \quad k = 0, 1, \dots, (N-1)$$

Az inverz Fourier-transzformáció pedig a következő összefüggésen alapul:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{ink \frac{2\pi}{N}}, \quad k = 0, 1, \dots, (N-1)$$

Gyors Fourier-transzformáció (FFT)

Mivel nagyméretű N esetén a diszkrét Fourier-transzformáció végrehajtása nagyon nagy műveletszámot igényel, ezért valós idejű feldolgozás esetén lehetetlen a használata. A gyors Fourier-transzformáció ugyanazt az eredményt szolgáltatja, mint a DFT, de mivel több egyszerűsítést is tartalmaz, így jóval kevesebb számítással jut el ugyanoda. Az első változtatás az, hogy a pontok számát $N = 2^y$ -ként határozza meg, ahol y egész szám. Például, ha $y = 3$, 4, és ekkor $N = 8, 16$. Továbbá észre kell venni a redundanciát, amely egyrészt abból következik, hogy a trigonometrikus függvények periodikusak, másrészt abból, hogy transzformációs műveletek lineárisak.

Hangmagasság

A hallható hangok tartománya 10 oktávot tesz ki, ami nagyjából 16 Hz és 16000 Hz között van. Ennek ellenére a zenében általában csak 7 oktávot használnak. Ezek elnevezése a zongora esetében kontra, nagy, kis, egyvonalas, kétvonalas, háromvonalas és négyvonalas. Ez a tartomány 32 Hz és 4200 Hz közé esik, és 84 félhang fér bele. Az egyes hangszerek kitöltik, sőt sok esetben túl is haladhatják ezt a 7 oktávnyi szélességet. A hangmagasság a hangfrekvenciától függ. A frekvencia növelésével a hangmagasság érzete is növekedni fog. Kutatások alapján megállapították, hogy kétféle hangmagasság érzetünk működik: egy melodikus és egy harmonikus. A kettő közül a melodikus adja meg a valódi hangmagasságot. Az érzettulajdonságok háromfélék lehetnek. Ha például két hang egyszerre szól, akkor azok hangossága nagyobb, mint a hangoké külön-külön volt, ezt összeadódónak nevezzük. Ha két hang két különböző irányból szól, akkor a közös irányuk valahol a kettő között lesz. Az ilyen tulajdonságokat közepelhető tulajdonságoknak nevezzük. Azonban két egyszerre szóló hang hangmagasságának viselkedése az előbbieken bemutatott két példa egyikére sem hasonlít. Ugyanis a két egyszerre szóló hangmagasság hatására egy összeolvadó érzet keletkezik bennünk.

„A különbségi küszöb azt a legkisebb ingerváltozást jelenti, amely még éppen észrevehető érzetváltozást okoz.” A Weber-féle törvény szerint a különbségi küszöb relatív értéke állandó. Azonban mostanra mérésekkel is alá tudják támasztani, hogy ez a különbségi küszöb nem állandó. Az eredmények azt mutatják, hogy 1000 Hz felett beszélhetünk nagyjából állandóságról, de alatta nem.

Inger- és érzet-oldali magassági skálák

Ingeroldali skálák:

A frekvencia skálája

Ez egy fizikai lineáris mérték, ahol a skála egysége a hertz (Hz). A zenei normálhang (az egyvonalas a) frekvenciája 440 Hz. Sok-sok év alatt alakult ez ki. A barokk idején például 422,5 Hz frekvenciájú hang volt az alaphang, ami a romantika alatt is folyamatosan tovább emelkedett. Végül 1951-ben vezették be magyar szabványnak a 440 Hz-et.

Az egyenletesen temperált hangsor hangjainak alapfrekvenciái ($a_1 = 440$ Hz)

	C_{-2}	C_{-1}	C	c	c_1	c_2	c_3	c_4	c_5	
	szub-kontra	kontra	nagy	kis	egy-vonalas	két-vonalas	három-vonalas	négy-vonalas	öt-vonalas	
C	16,352	32,703	65,406	130,813	261,626	523,251	1046,502	2093,005	4186,009	C
Cisz Desz	17,324	34,648	69,296	138,591	277,183	554,365	1108,731	2217,461	4434,922	Cisz Desz
D	18,354	36,708	73,416	146,832	293,665	587,330	1174,659	2349,318	4698,636	D
Disz Esz	19,445	38,891	77,782	155,563	311,127	622,254	1244,508	2489,016	4978,032	Disz Esz
E	20,602	41,203	82,407	164,814	329,628	659,255	1318,51	2637,021	5274,042	E
F	21,827	43,654	87,307	174,614	349,228	698,456	1396,913	2793,826	5587,652	F
Fisz Gesz	23,125	46,249	92,499	184,997	369,994	739,989	1479,978	2959,955	5919,91	Fisz Gesz
G	24,499	48,999	97,999	195,998	391,995	783,991	1567,982	3135,964	6271,928	G
Gisz Asz	25,957	51,913	103,826	207,652	415,395	830,609	1661,219	3322,438	6644,876	Gisz Asz
A	27,500	55,000	110,000	220,000	440,000	880,000	1760,000	3520,000	7040,000	A
Aisz B	29,135	58,270	116,541	233,082	466,164	932,328	1864,655	3729,31	7458,62	Aisz B
H	30,868	61,735	123,471	246,942	493,883	987,767	1975,533	3951,066	7902,132	H

A zenei hangköz

„Arányos két hang frekvenciahányadosának logaritmusával.” Alapegysége az oktáv. Az oktáv frekvencia-aránya 2. A jól hangolt skálában az oktávot 12 egyenlő részre osztjuk, ekkor két szomszédos hang távolsága (frekvencia-aránya) mindig egységesen ugyanannyi ($\sqrt[12]{2}$). Tehát a jól hangolt skála frekvenciaértékek egy mértani sorozatát jelenti, amelyben a hányados $\sqrt[12]{2}$.

Néhány tiszta hangköz frekvencia-aránya:

- tiszta oktáv = 2 : 1
- tiszta kvint = 3 : 2
- tiszta kvart = 4 : 3
- nagy terc = 5 : 4
- kis terc = 6 : 5

Érzetoldali skála

A melodikus hangmagasság skálája

Ezt röviden mel-skálának is szokták nevezni. Egysége éppen ezért a mel, aminek a frekvenciához viszonyítottan rögzített alapértéke:

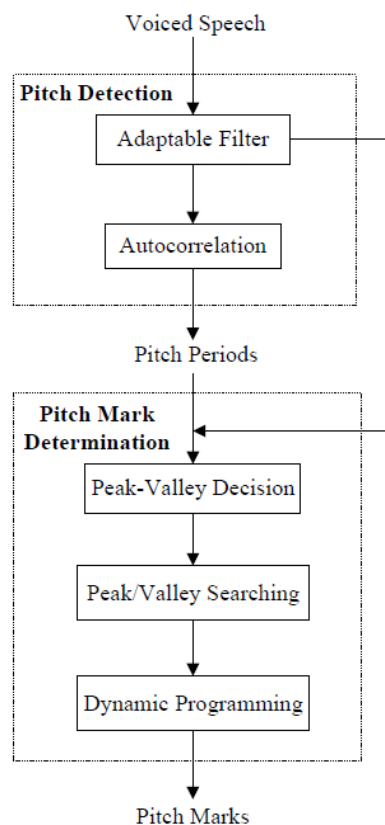
$$1000 \text{ Hz} = 1000 \text{ mel} \text{ vagy } 131 \text{ Hz} = 131 \text{ mel.}$$

Hangmagasság-detektáló algoritmusok

A hangmagasság-detektáló modellekkel szemben támasztott egyik legfontosabb követelmény, hogy a hangmagassági hibákat minimalizálják. A másik legkritikusabb tényező a késleltetés. Attól kezdve, hogy a hangot lejátszották, a fülnek legalább hét ciklusig tart pontosan azonosítania a hangmagasságot. Ezért a hangkezdetek és a hangmagasságok érzékelése nem áll közvetlenül kapcsolatban. Ha jó minőségű hangmagasság információra van szükségünk, akkor várunk kell.

Egy csúcs-völgy becselő módszeren és egy adaptálható szűrőn alapuló modell

A bemeneti jelet zöngés beszédre korlátozzuk, mert a modell szempontjából csak a periodikus részek lesznek érdekesek. Bemutatásra kerül továbbá egy adaptálható szűrő azért, hogy a zöngés beszédet egy szinusz-szerű hullámmá tudjuk transzformálni. Továbbá szó lesz egy csúcs-völgy becselő módszerről, hogy el tudjuk dönteni, hogy a beszéd mely része alkalmas a hangmagassági periódus kezdetének becslésére. A pozitív amplitúdójú és a negatív amplitúdójú rész kerül megvizsgálásra ebben a módszerben. Minden hangmagassági periódusban két lehetséges csúcst/völgyet keresünk.



A fenti ábrán látható a módszer folyamata. Első lépésként a hangmagasság detektálása történik meg egy adaptálható szűrő, majd az autokorrelációs módszer használatával. A szűrő áteresztési tartománya 50 Hz-től a zöngés beszéd alapfrekvenciájáig tart.

Ezt a szűrőt a következő három lépéssel érhetjük el:

1. Alkalmazni kell a gyors Fourier-transzformációt, hogy átranzszformáljuk a zöngés beszédet a frekvencia tartományba.
2. Ez után meg kell keresnünk az alapfrekvenciát (f_0). Ez azt jelenti, hogy meg kell keresnünk a spektrális kontúrnak az első kiemelkedését.
3. Ahhoz, hogy megkapjuk a szűrt beszédet, meg kell hívnunk az inverz Fourier-transzformációt az 50 Hz és f_0 közötti tartományban.

A szűrt beszéd már egy szinusz-szerű hullám lesz, ami tisztább periodicitást fog mutatni, mint az eredeti jel.

A következő lépésben a csúcs-völgy döntésről lesz szó. Ennek a résznek annyi a feladata, hogy kiválasszuk vagy a csúcsokat, vagy a völgyeket a további lépésekhez. A továbbiakban az eredeti zöngés beszédre s -sel, míg a már szűrt jelre o -val fogunk hivatkozni. A módszer két költséget számol az $s[m]$ amplitúdók összeadásával, ahol m a hely extrém pontot reprezentálja az o -ra minden hangmagassági periódusban.

$$C_{peak} = \frac{1}{N_{peak}} \cdot \sum_{n=1}^{N_{peak}} s[Pos_{peak}[n]]$$

$$C_{valley} = \frac{-1}{N_{valley}} \cdot \sum_{n=1}^{N_{valley}} s[Pos_{valley}[n]]$$

A képletek megértéséhez szükség van némi magyarázatra is. A C_{peak} az o csúcsaiban becsült költséget, a C_{valley} pedig az o völgyeiben becsült költséget jelenti. Az N_{peak} az o -ban lévő csúcsok teljes száma, míg az N_{valley} az o -ban lévő völgyek teljes számát jelenti. A Pos_{peak} az o n -edik csúcsának a helye, a Pos_{valley} az o n -edik völgyének a helyét adja meg. Végül a csúcs-völgy döntés a következőképpen történik: Ha $C_{peak} > C_{valley}$, akkor csúcsokat, különben pedig a völgyeket adoptáljuk.

A módszer utolsó fázisa dinamikus programozással történik. Ha például a csúcsokat választottuk ki az előző lépésben, akkor a következőket kell még végrehajtanunk (természetesen a lépések a völgyek adoptálása esetén is nagyon hasonlóak): Az i -edik hangmagasság periódusra a két legmagasabb csúcsot keressük meg (ezek rendre L_{i1} és L_{i2} lesznek). Ha a második esetleg nem lenne jelen, akkor ebben az esetben az előbb említett két érték meg fog egyezni. Majd a hangmagassági periódus torzítása ($d_i(j,k)$), és összegyűjtése ($A_i(j)$), valamint egy büntető függvény segítségével megkapunk egy csúcssorozatot, amiből a csúcsok helyei úgy kaphatók meg, hogy visszakövetjük a legkisebb $A_i(1)$ és $A_i(2)$ értékekhez tartozó csúcssorozatot.

Harmonikus szorzat spektrum (HPS)

Ez az algoritmus abból a szempontból nagyon jó, hogy egyszerűen implementálható. Nagy hátránya, hogy növelni kell az alacsony frekvencia felbontást a jel nullákkal való kitöltésével a transzformálás előtt. Ezt a következőképpen tehetjük meg:

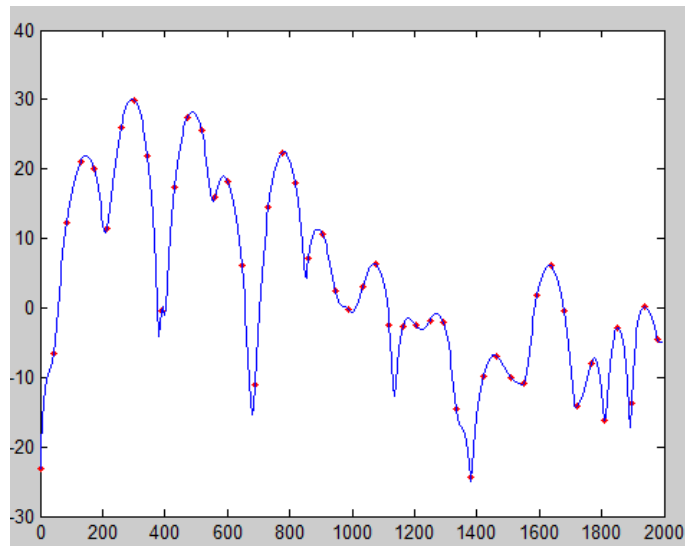
Adott egy jel, amelynek szeretnénk a spektrumát megvizsgálni, különös tekintettel az alacsony frekvenciákra, pl. félhangnál is sűrűbben. Ha az FFT-t N pontra végezzük, akkor a spektrum $0, f_s/N, 2f_s/N, 3f_s/N, \dots (N-1)f_s/N$ frekvenciákra készül. Ilyenkor az elmélet úgy tekinti az N pontot, mint egy periodikus jel egy periódusát. Az FFT eredménye is periodikus lesz. A fenti elem- N -s lesz egy periódus, sőt szimmetrikus is, ezért vesszük az eredménynek csak az első felét figyelembe. Vesszünk egy rövid részletet (például $N = 1024$), amelyet Hanning-ablakkal szorzunk úgy, hogy az ablak a teljes részletet lefedje. Ez simítani fogja a spektrumot. Ha az előbbi szorzatot sok-sok nullával kiegészítjük, akkor a spektrum besűrűsödik, interpolálódik. Ez sok számítási erőforrást elpazarol, mert nem csak az alacsony frekvenciákat interpoláljuk, hanem a magasakat is, de feleslegesen.

Az algoritmus méri a maximális egybeesést a harmonikusokkal az első egyenlet szerint, minden egyes $X(\omega)$ spektrális keretre ($\omega=2\pi f$).

$$Y(\omega) = \prod_{r=1}^R |X(\omega_r)|$$

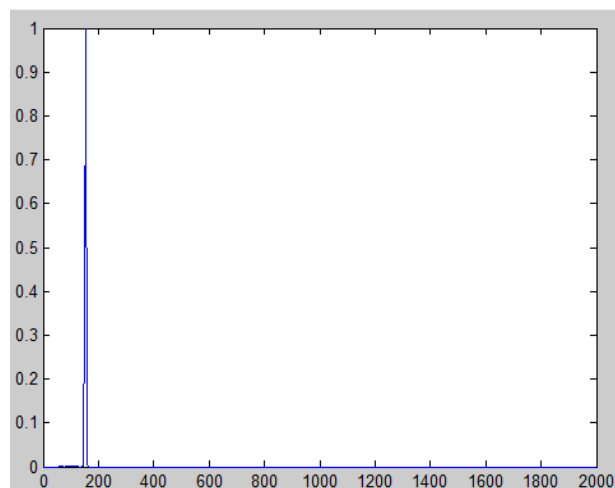
$$\hat{Y} = \max_{\omega_i} \{Y(\omega_i)\}$$

A képletben R azoknak a harmonikusoknak a számát adja meg, amiket figyelembe kell venni ($R = 5$).



A fenti ábra szemlélteti az alacsony frekvencia felbontás növelését a jel nullákkal való kitöltésével. A képen tízszeres sűrűségűre sikerült besűrűtünk a spektrumot. A HPS algoritmus által végzett hangmagasság-meghatározásokban is gyakoriak az oktávhibák. Ez azt jelenti, hogy a hangmagasságot egy oktávval feljebb vagy lejjebb detektálja az algoritmus, mint ahol az igazából van. Ennek a hibának a javítása utófeldolgozást igényel a következő módszer szerint:

Ha a második csúcs amplitúdója a kezdetben választott hangmagasság alatt van, és értéke nagyjából fele a választott hangmagasságnak, és az amplitúdók aránya a határérték fölött van (például 5 harmonikus esetén 0.2), akkor az alacsonyabb oktávú csúcsot kell választani az algoritmus által meghatározott hangmagasságnak.



Maximum Likelihood (ML)

Ez egy statisztikán alapuló módszer, ami előállítja ideális spektrumok egy halmazát, majd ezek közül választja ki azt, amelyik a legjobban illeszkedik a bemeneti spektrum alakjára. Az ideális spektrum a következőképpen definiálható: az ω körfrekvenciához tartozó jel spektrumának az ablak spektrumával való konvolúciója.

Ez után minimalizálni kell a hibát a spektrális keret és a lehetséges spektrumjelöltek között. Így fogjuk tudni majd eldönteni, hogy melyik ideális spektrum illeszkedik a legjobban, tehát a legkisebb hibával.

$$\begin{aligned} E(\omega) &= \|Y - \tilde{Y}_\omega\|^2 \\ &= \|Y\|^2 + \|\tilde{Y}_\omega\|^2 - 2Y\tilde{Y}_\omega^T \end{aligned}$$

ahol Y a megadott input spektrum, míg $\tilde{Y}_\omega$ egy ideális spektrum ω alapfrekvenciával. A fenti képletben az $\|Y\|^2$ konstans érték lesz, míg az $\|\tilde{Y}_\omega\|^2$ szintén konstans marad, ezért a maximumot csak a harmadik tagban kereshetjük. Ezután meg kell találni a két spektrum szorzatának a maximumát, ahol a legvalószínűbb $\tilde{Y}$ hangmagasságot rendeljük hozzá.

$$\hat{Y} = \min_{\omega} \{E(\omega)\} = \max_{\omega} \{Y\tilde{Y}_\omega^T\}$$

A harmonikus szorzat spektrum algoritmusnál még transzformáció előtt meg kellett növelni az alacsony frekvencia felbontást. Ennél az algoritmusnál ilyet nem kell végrehajtani, ezért gyorsabb algoritmust kapunk.

Vizsgálatok eredményeképpen arra jutottak, hogy a billentyűs és fafúvós hangszerek esetén jobb eredményt ad az ML algoritmus, mint a vonósok vagy az énekhang esetén.

Növelhetjük az algoritmus hatékonyságát, ha a hangmagasságoknak csak egy korlátozott tartományát adjuk meg. Ez azt jelenti, hogy kevesebb ideális spektrumot kell legenerálni, mert ami a tartományon kívül esik, arra már nincs szükség.

Kepsztrum

A beszédelemzésben az egyik legnépszerűbb hangmagasság meghatározó módszer. A kepsztrumot úgy számítjuk ki, hogy vesszük a bemenet abszolútérték spektruma logaritmusának a diszkrét Fourier-transzformáltját. Az elmélet szerint a jel egyenlő távolságoként tartalmaz csúcsokat. A kepsztrum vizsgálatakor keresünk egy csúcsot a tartományban, majd egy második, illetve egy harmadik csúcsot is egyenlő távolságra az első, illetve a második csúcstól.

A legkiemelkedőbb csúcs a kepsztrumban az alapperiódus. A kisebb amplitúdójú csúcsok pedig a harmonikusok. Egy periodikus jelnek a spektrumban lesz egy nagy kiemelkedésű csúcsa az alapprofrendvenciánál és több kiemelkedése ennek többszöröseinél. Ezek a csúcsok pedig szabályos közönként helyezkednek el. Ez az intervallum felel meg a kepsztrum alapperiódusának. Azonban egy gyengén periodikus jel spektrumában sok kiemelkedő csúcs lesz, amelyek nem követik egymást ilyen meghatározott intervallumonként.

A harmonikus szorzat spektrum algoritmus már említésre került az egyik korábbi fejezetben. Kutatásokkal kimutatták, hogy ha ezt a két algoritmust kombináljuk, akkor egy jóval megbízhatóbbat tudunk létrehozni. Az első probléma abból adódik, hogy a két függvény különböző tartományokon létezik. Ezért először a kepsztrumot frekvencia tartomány indexelésűvé kell tenni. Ezek után már egy csúcs, ami a periódust jelölte k értékben a kepsztrumban, most egy hangmagasságot jelez egy hozzátartozó értékkel. Ha ezt az újonnan kapott függvényt összeszorozzuk a HPS-sel (megtehetjük, mert már ugyanabban a tartományban vannak) egy új függvényt fogunk megkapni (CBHPS), ami már megbízhatóbban kezeli a zajt és a hangmagassági hibákat.

Autokorreláció (ACF)

Az autokorrelációs függvény a következőképpen írható le:

$$\phi(\tau) = \frac{1}{N} \sum_{n=0}^{N-1} x(n)x(n + \tau)$$

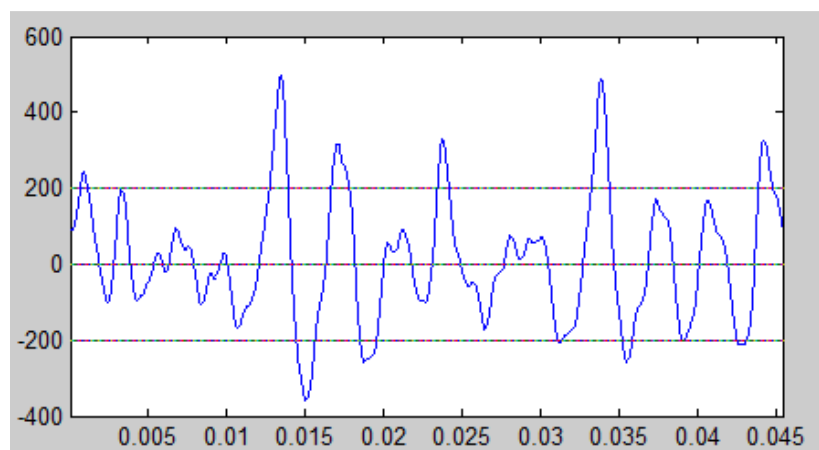
A gyakorlatban az $\frac{1}{N}$ -es szorzót sokszor le szokták hagyni, mivel nincs olyan nagy jelentősége.

A jelnek az eltolójával való hasonlóságát adja meg, az eltolás figyelembevételével. Tehát annak a mértékét méri, hogy a jel mennyire korrelál önmaga időeltolódásos verziójával. A periodicitás tulajdonságai miatt kell lennie egy maximális hasonlóságnak, olyan eltolás értéknél, amely a jel periódusához hasonló. Az autokorreláció kevésbé hatékony magas frekvenciákon, illetve az emberi hang feletti frekvenciákon.

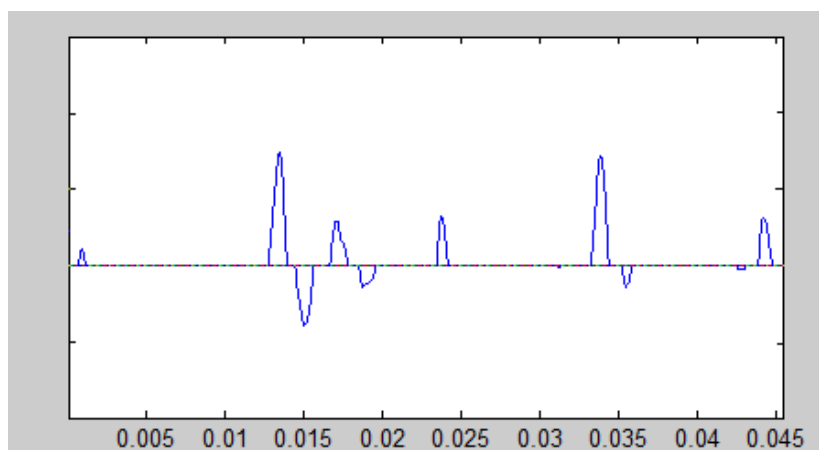
Érdemes itt megemlíteni a középkihágás fogalmát is. A módszer alkalmazásával a jel periodicitása emelhető ki. Azt, hogy éppen mekkora a vágási szint, a pillanatnyi maximális amplitúdóhoz kell igazítani. A középkihágás után a jelből csak szabályosan ismétlődő csúcsok maradnak meg, amikből leolvasható a jel periódusideje. Ezt a módszert az autokorrelációs módszerrel együtt szokták alkalmazni.

Példa középkihágásra:

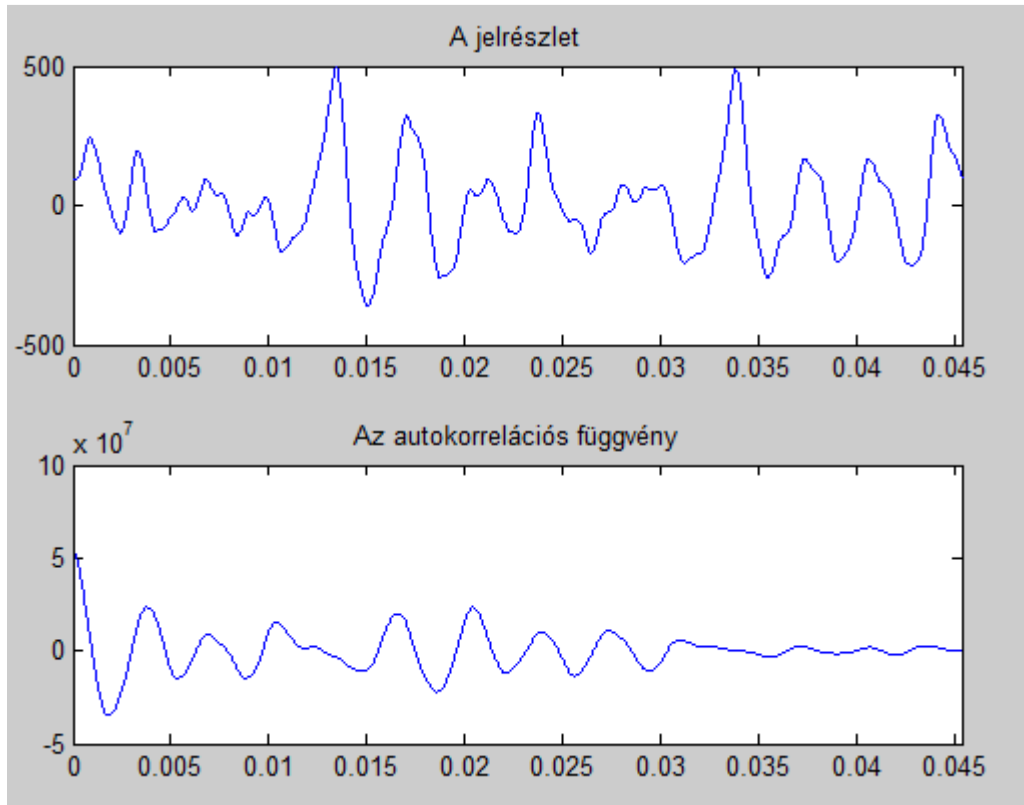
Az eredeti jel egy részlete:



Az előbbi jel a középkihágás végrehajtása után:



Az alábbi ábrák egy jelrészletet és annak autokorrelációs függvényét szemléltetik.



Átlagos magnitúdó-különbség függvény (AMDF)

Az AMDF egy alternatívája az autokorrelációs függvénynek. A következőképpen definiálják:

$$\psi(\tau) = \frac{1}{N} \sum_{n=0}^{N-1} |x(n) - x(n + \tau)|.$$

ahol $x(n)$ a jelsorozat, ami egy N hosszúságú téglalap alakú ablakkal van megszorozva, a τ pedig az eltolás mértéke. A τ 0 és $N-1$ közötti értékeket vehet fel, és a $\sum$ előtti konstans kifejezés a normalizációért felelős. Ahelyett, hogy a jel önmaga időbeni eltoltjával vett szorzatát néznénk, ez a módszer a két jel közötti különbséget fogja kiszámolni. Amíg az autokorrelációs függvénynél csúcsok vannak ott, ahol maximális a hasonlóság, addig az AMDF esetén ugyanitt völgyeket fogunk találni.

Az AMDF kiszámítása kevésbé számításigényes, mint az autokorrelációs függvényé, mivel a szorzások helyett kivonások vannak benne.

Az ACF és az AMDF kombinálhatók egymással a jobb hangmagasság becslés érdekében. Kutatásokkal alá is támasztották, hogy a kombinált függvény sokkal ellenállóbb a zajjal és a hangmagasság hibákkal szemben.

$$f(\tau) = \frac{\phi(\tau)}{\psi(\tau) + 1}.$$

Zérusátmenet-detektálás

Az alapötlet az, hogy számoljuk egy adott időegység alatt a jelben történő zérusátmeneteket. Az elmélet szerint az alaphfrekvencia ennek a számnak a fele kellene, hogy legyen. Azonban a gyakorlatban ennek a technikának egyik nagy gyengesége például a zajos jel. Ez az elképzelés javítható, ha egy adaptív szűrőt használunk [Tadokoro, Matsumoto, Yamaguchi 2002].

YIN algoritmus (Alain de Cheveigné és Hideki Kawahara)

Ez az algoritmus alaphfrekvencia meghatározására alkalmas beszéd vagy zenei hangok esetén. A jól ismert autokorrelációs módszeren alapul néhány változtatással a hibák megelőzése érdekében. A következő részekben bemutatásra kerül maga a módszer és annak lépései.

1. lépés: Az autokorrelációs módszer

Egy x diszkrét jel autokorrelációs függvénye a következőképpen definiálható:

$$r_t(\tau) = \sum_{j=t+1}^{t+W} x_j x_{j+\tau}$$

ahol az $r_t(\tau)$ az autokorrelációs függvény τ -val eltolva a t időindexben kiszámolva, W pedig az ablak mérete. A definíció szerint a jel a $[t+1, t+W]$ tartományon kívül 0. Ha x egy periodikus jel, akkor annak az autokorrelációs függvénye csúcsokat mutat a periódus többszöröseinél. Az autokorrelációs függvény a legnagyobb nem nulla eltolású csúcsot választja az eltolások tartományában való alapos keresés után. Több helyen is lehetőség van hibára. Ha az alsó határ túl közel van a nullához, akkor az algoritmus hibásan egy nulla eltolású csúcsot választhat. Ha azonban a magasabb határ túl nagy, akkor hibásan magasabb rendű csúcsot választhat. Az autokorrelációs módszer tehát a jelet annak eltolt változatához hasonlítja. Az AMDF már az összehasonlításokat szorzatok helyett különbségek alkalmazásával végzi.

2. lépés: Különbség függvény

Tudjuk az x jelről, hogy periodikus és a periódusa T , amiből következik, hogy definíció szerint invariáns a T időbeli eltolásra:

$$x_t - x_{t+T} = 0, \quad \forall t.$$

Ugyanez igaz, ha vesszük a négyzetét és átlagoljuk egy ablakon át:

$$\sum_{j=t+1}^{t+W} (x_j - x_{j+T})^2 = 0.$$

Ezek után a különbségfüggvény a következőképpen néz ki:

$$d_t(\tau) = \sum_{j=1}^W (x_j - x_{j+\tau})^2,$$

A periódus összes többszöröse olyan érték, amire a függvény nulla. A négyzetes összeg kiterjeszthető és a függvény kifejezhető ACF-fel:

$$d_t(\tau) = r_t(0) + r_{t+\tau}(0) - 2r_t(\tau)$$

Az első két kifejezés az energia kifejezés. Ha állandóak lennének, akkor a $d_t(\tau)$ különbségfüggvény úgy változna, mint az $r_t(\tau)$ ellentéte, és a minimumának vagy a másik maximumának a keresése ugyanazt jelentené. Azonban a második energia kifejezés szintén változik a τ -val, arra utalva, hogy az $r_t(\tau)$ maximuma és a $d_t(\tau)$ minimuma néhány esetben nem esik egybe.

3. lépés: Halmozott átlag normalizált különbségfüggvény

A különbség függvény hibáinak elkerülésére az algoritmus által ajánlott megoldás az, hogy helyettesítsük a halmozott átlag normalizált különbségfüggvénnyel:

$$d'_t(\tau) = \begin{cases} 1, & \text{if } \tau = 0, \\ d_t(\tau) / \left[(1/\tau) \sum_{j=1}^{\tau} d_t(j) \right] \end{cases}$$

Az új függvényt úgy kapjuk, hogy a régiek minden értékét elosztjuk a rövid eltolású értékek átlagával. Abban különbözik a $d(\tau)$ -tól, hogy 0 helyett 1-től indul, és csak ott esik 1 alá, ahol $d(\tau)$ az átlag alá esik.

4. lépés: Abszolút küszöb

Lehetséges, hogy a különbségfüggvény magasabb rendű mélypontjainak egyike alacsonyabban van, mint a periódus mélypont. Ha beleesik a keresési tartományba, akkor hibás eredményt kapunk. A megoldás az lehet, hogy beállítunk egy abszolút küszöb értéket és kiválasztjuk a legkisebb τ értéket, amely a d' minimumát adja a küszöbnél mélyebben. Ha nem találunk egyet sem, akkor egy globális minimumot választunk helyette.

5. lépés: Parabolikus interpoláció

Az előző lépések megfelelően működnek, ha a periódus többszöröse a mintavételezési periódusnak. Ha nem, akkor a becslés eltérhet akár fél mintavételezési periódussal is. A megoldás erre a problémára a parabolikus interpoláció. A $d'(\tau)$ minden lokális minimuma és a közvetlen szomszédjai illeszkednek egy parabolára, és az ordinátáját az interpolált minimumnak a mélypontválasztó folyamatban használjuk. A kiválasztott minimum abszcisszája a periódus becsléseként szolgál. Azért, hogy az így kapott becslés elfogultságát csökkentsük a nyers különbségfüggvény ($d(\tau)$) hozzátartozó minimum abszcisszáját használjuk helyette.

6. lépés: Legjobb lokális becslés

Minden t időindexre keressük meg a $d'_\theta(T_\theta)$ minimumát a $[t-T_{\max}/2, t+T_{\max}/2]$ intervallumban, ahol a T_θ a θ időbeli becslés és a $T_{\max}$ a legnagyobb várt periódus. Ezen a kezdeti becslésen alapulva a becslő algoritmust újra alkalmazzuk egy leszűkített keresési tartománnyal, hogy megkapjuk a végső eredményt.

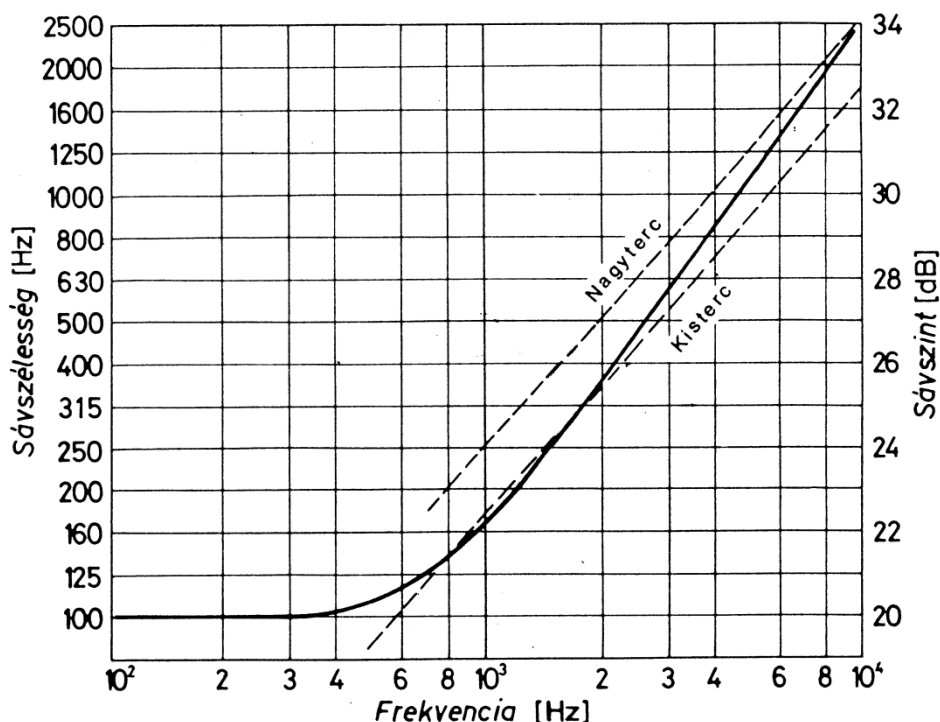
Hangosság

A hallott hang nem azonos a fizikai hangjelenséggel, és nem is mérhető annak a mértékeivel. Az érzetnek szintén vannak mértékei és egységei. Az intenzitástól (objektív) megkülönböztetendő a hangosság (szubjektív). A fizikai hatás az inger, az agyban tudomásul vett belső válasz pedig az érzet.

A hallásterület frekvencia szerinti alsó határa nem alakult ki, mert 20 Hz táján fokozatosan más minőségű érzetbe megy át, valamint ez a határ ide-oda tolható a figyelem megfelelő összpontosításával. A felső határ azonban a korral fokozatosan csökken. Nagyon fiatal korban akár a 25 000 Hz-et is képes az ember érzékelni, míg a határ 20-25 éves kor környékén állandósul kb. 16 000-18 000 Hz-en, ami a korral tovább csökken.

Kritikus hallási sávok

Hallásunk egyidejű összegző sajátossága alapján az észlelhető frekvencia-tartományt kritikus sávokra osztjuk fel. Több egyidejű komponens összegződése eltérő módon történik, ha azok egy kritikus sávon belül, illetve sávon kívül vannak. Ezen kritikus sávok sáv szélessége a frekvencia növekedésével nő: (500 Hz alatt 100Hz, 1 KHz-en kisterc, míg 10 KHz-en nagyterc). A hallástartomány 24 ilyen egymást érintő kritikus sávval fedhető le.

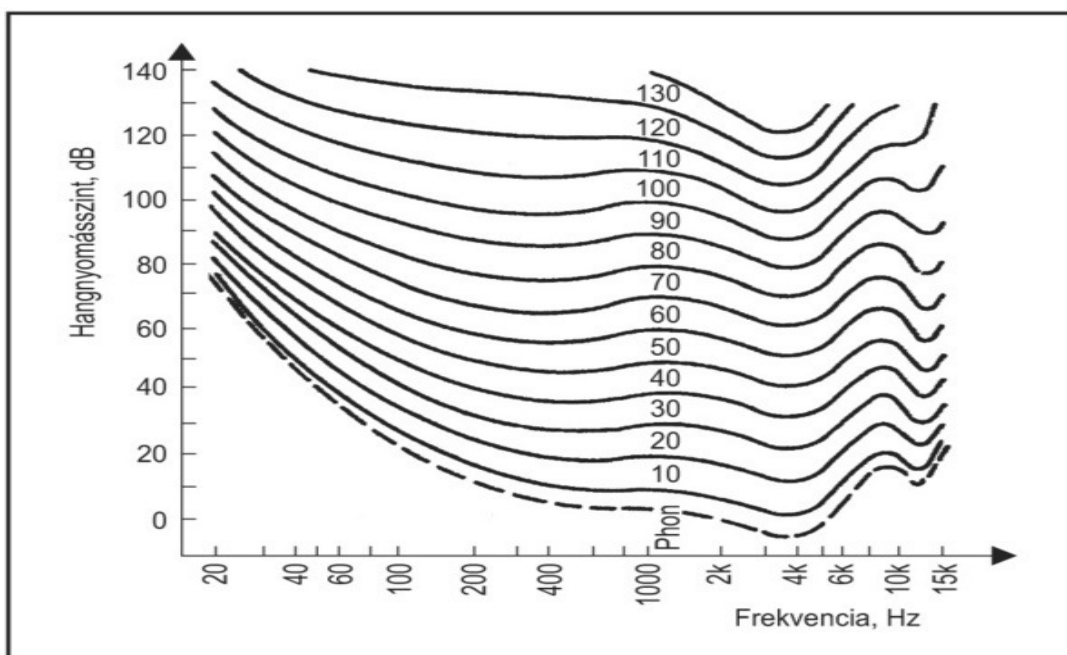


Az egyenlő hangosság szintek görbéi

A hangosság egyrészt a fizikai intenzitástól, másrészt a hang frekvenciájától függ. Azonos intenzitás-változásokhoz azonban nem azonos hangérzet-változások járnak.

Régebben úgy gondolták, hogy a hangosság az intenzitás logaritmikus függvénye (ez a Weber-Fechner-féle "törvény"), amiből az következik, hogy a decibel-skála az érzetoldalnak megfelelő lépték. Másképp, kétszeres intenzitásnövekedés a hangosság szintjét három egységgel, tízszeres növekedés tíz egységgel, százszoros pedig húsz egységgel emelné. Azaz a decibel-skálát kiterjesztették az érzeti oldalra. Ezt a skálát phon skálának nevezték el.

Hogy milyen összefüggés van az érzékelt hangosság, a fizikai intenzitás és a frekvencia között, erre adnak választ az egyenlő hangosság szintek görbéi. A görbéket egy mérés során határozták meg. A mérésben két hanggenerátort felváltva alkalmaztak. Az elsővel először egy adott intenzitásszintű (dB) 1000 Hz-es szinuszos hangot sugároztak. Ennek a hangnak a hangosságával kellett összevetni a másik generátor eltérő frekvenciájú hangosságát. A kísérleti személy addig változtatta a második generátor intenzitását, amíg azt azonos hangosságúnak ítélte az etalonhanggal. Minél mélyebb volt a hang, annál nagyobb intenzitás adódott. Az így kapott pontokat összekötve adódott az egyenlő hangosság szintek egy görbéje. Definíció szerint a *phon* és a *decibel* érték 1000 Hz-en megegyezik, másutt a görbéről olvasható le. Például 100 Hz 60 dB intenzitású tiszta hang hangosság szintje 52 phon.



Amíg 1000 Hz-en 10 dB változás mindig 10 phon változásnak felel meg, addig 30 Hz-en már 4-7 dB intenzitásváltozás is létrehozza a 10 phon hangosság szint változást.

Az I intenzitású hang intenzitás szintje:

$$H.\acute{e}rzet(phon) = 10 \lg \frac{I}{I_0}$$

Megállapították, hogy kb. 10 phon hangosság szint-növekedés felel meg kétszer olyan hangos hangérzetnek, 20 phon négyszeresnek, 30 phon pedig nyolcszorosnak, és így tovább.

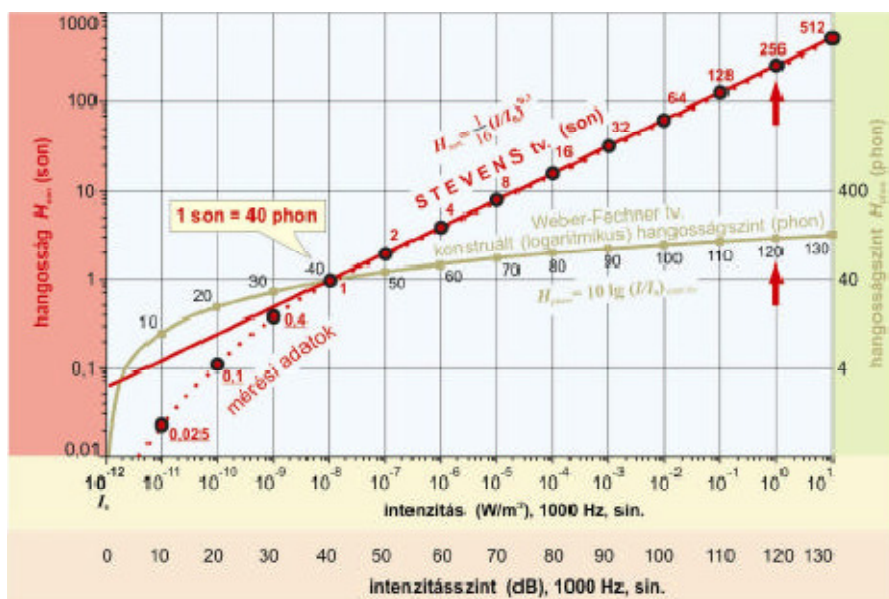
A valódi hangosság

Később kísérletekkel kimutatták, hogy a phon skála mérőszámai nem fejezik ki számszerűen, lineárisan hangosságérzetünk változásait. Ezért kitaláltak egy olyan skálát, amely a hangosságérzetet jobban leírja. Ennek a neve son-skála, az egysége a son, és a phon-rendszerhez kötődő azonosítási pontja

$$40 \text{ phon} = 1 \text{ son}$$

A kétszer ilyen hangos hang (50 phon) hangossága 2 son, a négyszer hangosabbé (60 phon) 4 son, a nyolcszor hangosabbé (70 phon) pedig 8 son és így tovább. Kísérleti személyek segítségével igazolták, hogy ha a fizikai intenzitás (az inger) 40 dB fölött kétszeresére nő, a hangosság csak $2^{0,3}$ értelemben emelkedik. Ha a fizikai intenzitás tízszeresére növekszik, akkor lesz a hangosság körülbelül kétszeres, mert $10^{0,3} = 1,995$. A további értékek az alábbi hatványfüggvényből (Stevens, 1936) számíthatók:

$$Hangosság(son) = \left(\frac{I}{I_0} \right)^{0,3}$$



Phon-son átszámítás (Három jegyre kerekítve, a zárójeles értékek pontatlanok)

Phon	30	40	50	60	70	80	90	100	110	120	130
0	(0,50)	1,00	2,00	4,00	8,00	16,0	32,0	64,0	128	256	512
0,5	(0,52)	1,04	2,07	4,14	8,28	16,6	33,2	66,5	133	265	530
1	(0,54)	1,07	2,14	4,29	8,58	17,1	34,3	68,5	137	275	549
1,5	(0,56)	1,11	2,22	4,44	8,87	17,7	35,5	71,0	142	284	568
2	(0,58)	1,15	2,30	4,60	9,20	18,4	36,8	73,5	147	294	588
2,5	(0,60)	1,19	2,38	4,75	9,50	19,0	38,0	76,0	152	304	608
3	(0,62)	1,23	2,46	4,92	9,85	19,7	39,4	78,8	158	315	630
3,5	(0,64)	1,27	2,55	5,10	10,2	20,4	40,8	81,6	163	326	653
4	(0,66)	1,32	2,64	5,29	10,6	21,1	42,2	84,5	169	338	676
4,5	(0,68)	1,37	2,73	5,46	10,9	21,9	43,8	87,6	175	350	699
5	0,71	1,41	2,82	5,65	11,3	22,6	45,2	90,5	181	362	724
5,5	0,73	1,46	2,93	5,86	11,7	23,4	46,8	93,7	187	375	750
6	0,75	1,52	3,03	6,06	12,1	24,2	48,5	97	194	388	776
6,5	0,78	1,57	3,14	6,28	12,6	25,1	50,2	100	201	402	803
7	0,81	1,62	3,25	6,50	13,0	26,0	52,0	104	208	416	832
7,5	0,84	1,68	3,36	6,72	13,4	26,9	53,8	108	215	430	861
8	0,87	1,74	3,48	6,96	13,9	27,8	55,7	111	223	446	891
8,5	0,90	1,80	3,61	7,22	14,4	28,8	57,7	115	231	462	923
9	0,93	1,86	3,73	7,46	14,9	29,8	59,7	119	239	478	955
9,5	0,96	1,93	3,86	7,72	15,4	30,9	61,8	124	247	495	989

A fenti ábra a phon-son átszámítást szemlélteti.

Az összhangosság

Itt az a kérdés, hogy mi történik abban az esetben, ha két hang egyszerre szól? Egy lehetséges kísérlet, ha összehasonlítunk két egyidejűleg szóló hangot egyetlen változtatható intenzitású hanggal. A kísérletben szereplő személy felváltva hallgatja a kétféle jelet, és megmondja, mikor hallja őket egyformán hangosnak.

- Mivel a son lineáris mérték, ezért számtanilag összegezzük:

$$1 \text{ son} + 1 \text{ son} = 2 \text{ son},$$

holott 40 phon + 40 phon = 43 phon lenne az eredmény a logaritmikus szabályok szerint. Tehát a son hangosságok általában simán összeadhatók, ha a két hang elég távol van egymástól.

- Ha két egyidejűleg szóló hang közül az egyik sokkal hangosabb, akkor a másik hatását elhanyagolhatónak vesszük. Az erősebb mély hang jobban hatástalanítja a gyengébb magasat, mint az erősebb magas a gyengébb mélyet.
- Minél hangosabb a két összetevő külön, annál távolabb kell lennie a két hangnak egymástól, hogy a son-összegződés érvényben maradjon. Zenei fff esetén 10-12 kritikus sáv kell hozzá, hogy az összegződés igaz legyen. Viszont két gyenge hang son-hangossága akkor is összegezzük, ha egy kritikus sáv választja el őket egymástól.

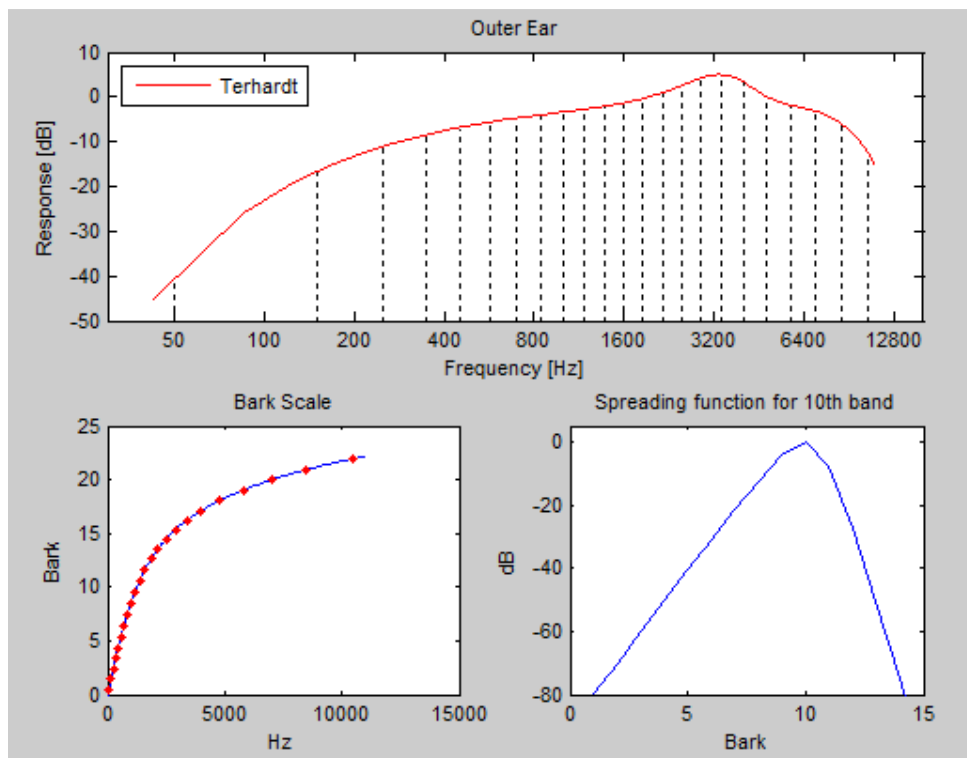
Hangossági modell implementáció

Az általam felhasznált hangosság algoritmus a <http://www.ofai.at/~elias.pampalk/ma/> címen érhető el, a MA Toolbox for Matlab függvényei között található meg.

A függvény két bemenő paramétert vár. Az első egy wavread-del beolvasott adathalmaz, ami csak mono lehet és a 11 kHz az ajánlott mintavételezési frekvencia. A második egy struktúra, ami a következő adatokat tartalmazza:

- p.fs – a megadott wav mintavételi frekvenciája,
- p.do_visu – készüljenek-e ábrák vagy sem,
- p.do_spread – alkalmazzon-e spektrális maszkolást vagy sem,
- p.outerear – külsőfül modell (terhardt vagy nincs),
- p.fft_size – a Fourier-transzformáció ablakmérete (egysége a minta),
- p.hopsize – az fft ablak lépésköze
- p.bark_type – a használt Bark skála típusa
- p.dB_max – az input wav maximális dB értéke

Ha bemenő paraméterek nélkül hívjuk meg, akkor tesztábrákat kapunk eredményül, amelyek közül az egyik a következőképpen néz ki:

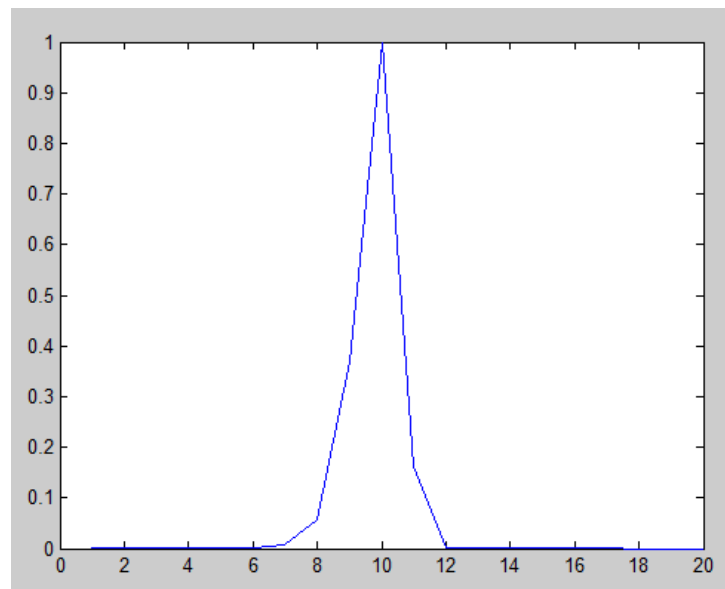


Ha megadunk a függvénynek input adatokat, akkor az fft frekvencia skálát készíti el először. Ezután, ha a `p.bark_type` értéke 'table' volt, akkor a programban be vannak építve a kritikus sávok felső határának és közepének frekvencia értékei. Ezek segítségével határozza meg azokat, amikre szükség lesz a későbbiekben, ugyanis csak a mintavételezési frekvencia feléig kell vizsgálnunk. Ha az értéke nem 'table', akkor egy 3 elemű vektorként kell megadnunk, ami a minimum frekvenciát, a maximum frekvenciát és a kritikus sávok számát tartalmazza ebben a sorrendben. Ekkor képletek segítségével történik meg az előző esetben adott értékek kiszámítása.

Ezután jön az árnyékolás kiszámítása. Ami a következő képlet segítségével történik:

```
10.^( (15.81+7.5*((i-(1:cb))+0.474)-17.5*(1+((i-(1:cb))+0.474).^2).^0.5)/10)
```

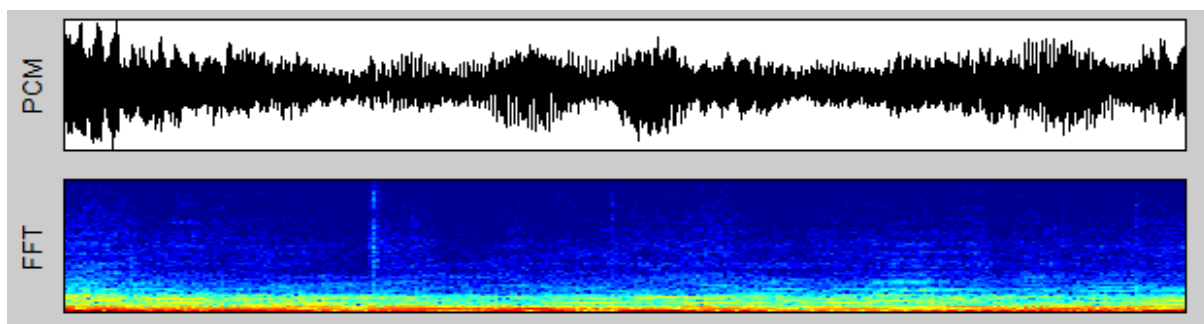
Ezt adjuk értékül a `c.spread(i,:)`-nek, miközben a for ciklus 1-től `cb`-ig megy.



A fenti ábra szemléltet a képlet alapján kapott árnyékok közül egyet.

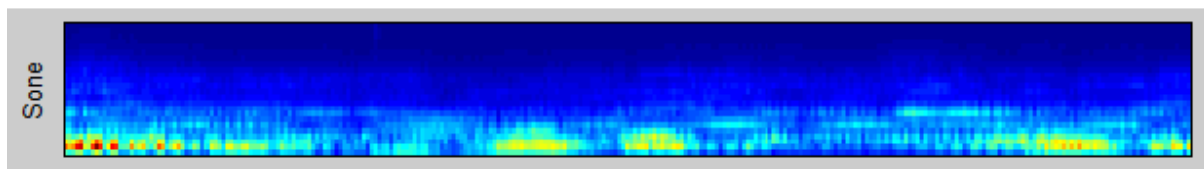
A következő lépésben meghatározásra kerül, hogy a bemenetként megadott wav hány keretre lesz felosztva. Ez a `p.fft_size` és a `p.hopsize` segítségével történik egy ciklusban. Ez után vesszük az adott keret Fourier-transzformáltját, és megszorozzuk egy `p.fft_size` méretű hann ablakkal, majd elkészítjük a normalizált energiaspektrumot. Majd kiszámításra kerülnek a `phon` és abból pedig a `son` értékek, és megtörténik a spektrális maszkolás is. A felhasznált képlet a `phon`-ból `son`-ba történő átszámításra a következő:

$$son = 2^{\frac{phon-40}{10}}$$



A fenti képen a függvény által eredményül adott ábrák közül az első kettőt látjuk. Az első a beolvasott hang ábrázolva, míg a második a normalizált energiaspektrumot mutatja. Ezt úgy kaptuk meg, hogy az adott `p.fft_size` hosszú keretet megszoroztuk egy ugyanekkora hann ablakkal, majd vettük ennek a Fourier-transzformáltját. Az így megkapott vektornak a továbbiakban csak a `p.fft_size/2`-ig tartó részét használjuk, és ezt lenormáltuk (azaz elosztottuk a `sum(w)`-vel). Az így kapott `X`-nek vettük az abszolútértékét és annak a négyzetét, ezzel megkaptuk a normalizált energiaspektrumot.

Az alábbi kép pedig a függvény által kiszámolt hangosság értékeket mutatja, ami a feldobott ábrában az utolsó koordináarendszerben jelenik meg.



Lineáris prediktív kódolás (LPC)

A lineáris predikció egy olyan matematikai módszer, amivel egy bizonyos időben lejátszódó jelenségnél a jelenség adott időbeni állapotát meg lehet becsülni a jelenségre vonatkozó eddigi tapasztalataink birtokában a korábbi időpillanatokban kapott értékekből. Akkor nevezzük lineárisnak a predikciót, ha a becslést a felhasznált értékek lineáris függvényeként kapjuk meg.

A szintézis modell áll egy ingerforrásból $U(z)$, ami inputot nyújt a $H(z)$ spektrumú átalakító filternek, ami az $\hat{S}(z)$ output beszédet szolgáltatja. A $H(z)$ egy véges sok együtthatóval definiált lineáris filter, ami legáltalánosabban törtalakú. Nagyon fontos szempont, hogy az $\hat{S}(z)$ outputnak olyan közel kell lennie az $S(z)$ eredeti jelhez, amennyire csak lehetséges. Az $U(z)$ -t és a $H(z)$ -t is az előbbi megszorítást figyelembe véve kell megválasztani.

Az LPC modell

A $H(z)$ megadása egy bonyolult feladat. Ezt úgy tudjuk egyszerűsíteni, ha az adott $s(n)$ beszédjelet állandó spektrumúnak tekintünk egy N mintányi ablakon keresztül. Tehát a beszédet úgy tekintjük, mint az N elemű minta egy ablakra történő leszűkítését. Ez megengedi azt, hogy a $H(z)$ filtert konstans együtthatókkal modellezzük, amiket természetesen minden új ablakban frissíteni kell. Feltételezzük, hogy $H(z)$ -nak p pólusa és q zérushelye van az általános pólus-zérushely esetben, ami azt jelenti, hogy az $\hat{s}(n)$ szintetizált beszéd modellezhető az LPC szintetizáló p db előző kimenetének és $q+1$ előző bemenetének a lineáris kombinációjaként:

$$\hat{s}(n) = \sum_{k=1}^p a_k \hat{s}(n-k) + G \sum_{l=0}^q b_l u(n-l),$$

ahol a G az input beszéd erősítési tényezője (feltételezzük, hogy $b_0=1$).

Ennek megfelelően,

$$H(z) = \frac{\hat{S}(z)}{U(z)} = G \frac{1 + \sum_{l=1}^q b_l z^{-l}}{1 - \sum_{k=1}^p a_k z^{-k}}.$$

A legtöbb LPC munka feltételez egy all-pole (csak pólusos) modellt, ahol $q = 0$. Ha megnézzük az előbbi képletet, akkor láthatjuk, hogy amennyiben $q = 0$, akkor a számlálóban található $\sum$ nem kerül végrehajtásra, mivel 1-től indul. Egy csak zérushelyes modellt, ahol $p = 0$, mozgó vagy gördülő átlagnak nevezünk, amikor az output a q megelőző input súlyozott átlaga.

Ha az $s(n)$ beszéd egy inverz filterrel van szűrve (egy végtelen pólusú $H(z)$ inverze)

$$A(z) = 1 - \sum_{k=1}^p a_k z^{-k},$$

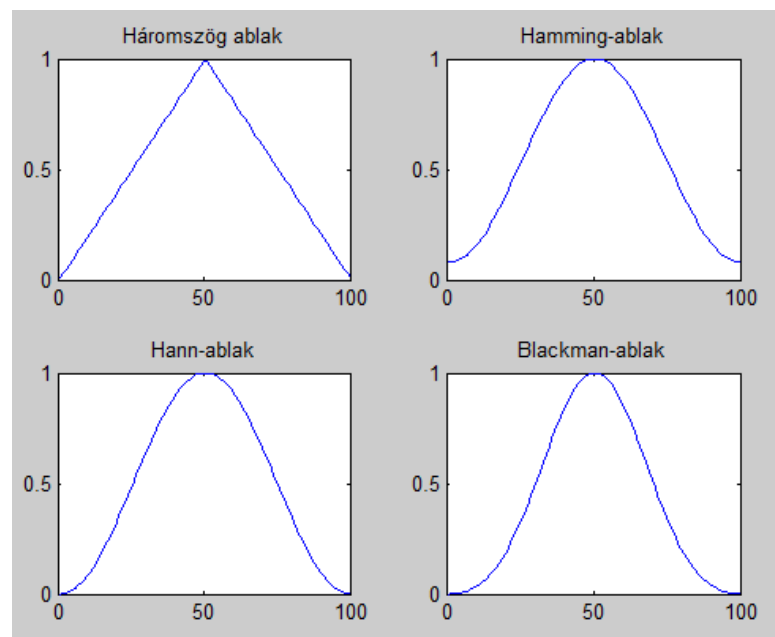
Az $e(n)$ kimenetet hibának vagy maradék jelnek nevezzük:

$$e(n) = s(n) - \sum_{k=1}^p a_k s(n-k).$$

Az $A(z)$ -re adott egység impulzus válasznak $p+1$ mintája van, és közvetlenül az LPC együtthatók számából jön: $a(0) = 1$, $a(n) = -a_n$, $n = 1, 2, \dots, p$ esetén.

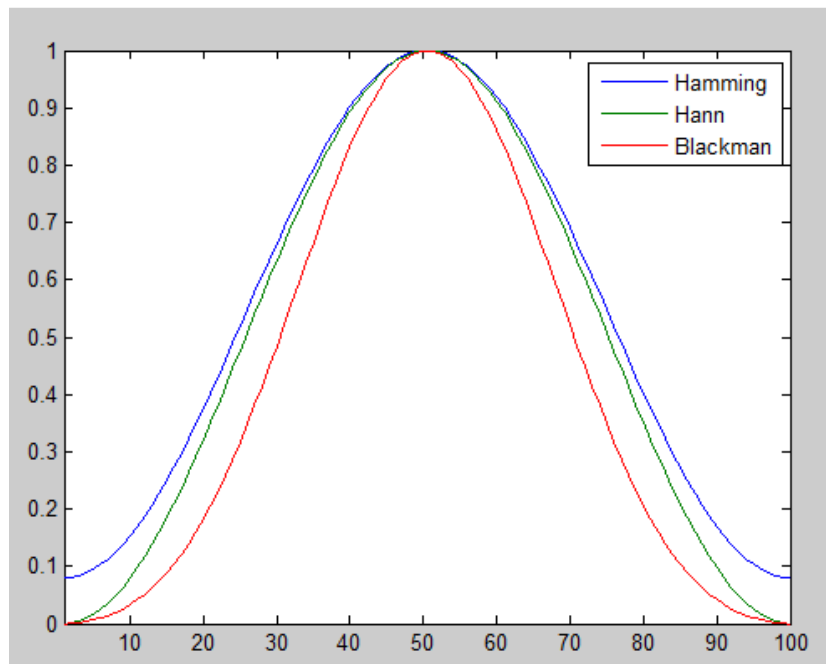
Ablakozás

Bizonyos algoritmusok előtt ablakozást kell alkalmazni. Az ablakozás annyit jelent, hogy időben leszűkítjük a jelet. Azaz a függvényt csak egy meghatározott intervallumon belül tekintjük, és azon kívül mindenütt 0-nak vesszük.



A fenti kép szemléltet négyet a legismertebb ablakok közül. A legegyszerűbb téglalap ablak nincs az ábrán. Ennek az alkalmazásakor az intervallumból veszem a jelet. Tehát az adott intervallumban az értéke 1, azon kívül pedig mindenütt 0. A cél az ablakozáskor az, hogy az intervallumon kívül eső jeleket kisebb súllyal vegyük, hogy ne torzítsák el a kapott eredményt. Erre a célra alkalmasabb az első képen látható háromszög ablak. A Hamming-ablak nem éri el a nullát, így a vele való szorzással nem fogjuk a jel szélső értékeit kinullázni.

Az alábbi kép három, egymáshoz eléggé hasonlító ablak közötti különbséget szemlélteti.



Legkisebb négyzetes autokorrelációs módszer

Kétféle megközelítést használhatunk ahhoz, hogy megkapjuk az a_k LPC együtthatókat. A klasszikus legkisebb négyzetek módszere ezt úgy teszi meg, hogy minimalizálja a hibajelben az átlagos energiát egy ablaknyi adatban, miközben a hálós megközelítés megengedi az együtthatók azonnali frissítését, ami természetesen extra számítást jelent. Az előző technikában vagy a beszéd vagy a hibajel ablakolt azért, hogy korlátozzuk az elemzés alatt álló jel terjedelmét.

Az első a két legkisebb négyzetes technikából az adatablakozás, vagy más néven autokorrelációs módszer, amely a jelet megszorozza egy Hamming vagy más időablakkal:

$$x(n) = w(n)s(n)$$

Tehát $x(n)$ véges időtartamú, és $x(n) = 0$, ha $0 \leq n \leq N-1$ tartományon kívül.

Az $s(n)$ -t állandónak feltételezzük minden ablakban. Legyen E az energia a maradék jelben:

$$E = \sum_{n=-\infty}^{\infty} e^2(n) = \sum_{n=-\infty}^{\infty} \left[x(n) - \sum_{k=1}^p a_k x(n-k) \right]^2$$

ahol $e(n)$ a maradék jelnek, míg $x(n)$ az ablakolt jelnek felel meg. Az a_k értékei, amelyek minimalizálják az E -t, úgy kaphatók meg, hogy $\partial E / \partial a_k = 0$, $k = 1, 2, 3, \dots, p$ esetén. Ez p darab lineáris egyenletet eredményez p darab a_k ismeretlennel.

$$\sum_{n=-\infty}^{\infty} x(n-i)x(n) = \sum_{k=1}^p a_k \sum_{n=-\infty}^{\infty} x(n-i)x(n-k), \quad i = 1, 2, 3, \dots, p.$$

Felismerve az első kifejezést mint az $x(n)$ autokorrelációs $R(i)$ -jét, és kihasználva $x(n)$ véges hosszát, a következőt kapjuk

$$R(i) = \sum_{n=i}^{N-1} x(n)x(n-i), \quad i = 1, 2, 3, \dots, p.$$

Tehát a kettővel fentebb leírt egyenlet a következőre egyszerűsödik:

$$\sum_{k=1}^p a_k R(i-k) = R(i), \quad i = 1, 2, 3, \dots, p.$$

Az autokorrelációt minden i egészre ki lehetne számítani, de mivel $R(i)$ páros függvény, ezért csak $0 \leq i \leq p$ tartományra kell meghatározni.

Legkisebb négyzetes kovariancia módszer

Egy alternatív legkisebb négyzetes technika, ami az $e(n)$ hibát ablakolja $s(n)$ helyett:

$$E = \sum_{n=-\infty}^{\infty} e^2(n)w(n).$$

A $\partial E / \partial a_k = 0$ beállítás itt is p darab lineáris egyenlethez vezet:

$$\sum_{k=1}^p a_k \phi(i, k) = \phi(0, i), \quad 1 \leq i \leq p.$$

ahol

$$\phi(i, k) = \sum_{n=-\infty}^{\infty} s(n-k)s(n-i)w(n)$$

lesz $s(n)$ kovariancia függvénye.

Az autokorrelációs R és a kovariancia φ függvények eléggé hasonlóak, de eltérnek az ablakozás megvalósulásában. Az autokorrelációs módszer ablakolt beszédmintát használ, míg a kovariancia módszer közvetlenül a beszédmintákat használja. Az autokorrelációs módszer így behoz egy torzítást a spektrumbecslés eljárásába, míg a kovariancia módszer elkerüli ezt a torzítást.

Számítási megfontolások

Az autokorrelációs módszerben a megoldandó p darab lineáris egyenletet nézhetjük mátrixos formában úgy, mint $\mathbf{R}\mathbf{A} = \mathbf{r}$, ahol $\mathbf{R}$ egy $p \times p$ -s mátrix, ami $R(i,k) = R(l_i - k_l)$, ($1 \leq i, k \leq p$) elemekből áll. Az $\mathbf{r}$ egy oszlopvektor $(R(1), R(2), \dots, R(p))^T$, és $\mathbf{A}$ az LPC együtthatókból álló oszlopvektor $(a_1, a_2, \dots, a_p)^T$. Ahhoz, hogy megkapjuk az LPC vektort szükségünk van az $\mathbf{R}$ mátrix inverzére és az eredményül kapott $p \times p$ -s mátrix és az $\mathbf{r}$ vektor szorzatára. Hasonló a helyzet a kovariancia esetben is, ahol $\mathbf{R}$ helyett Φ kovariancia mátrix van, amelynek elemei $\Phi(i, k) = \varphi(i, k)$. A $\mathbf{r}$ vektor helyett pedig φ vektor van $(\varphi(0,1), \varphi(0,2), \dots, \varphi(0,p))$.

Redundanciák az $\mathbf{R}$ és Φ mátrixokban lehetővé teszik az LPC együtthatók hatékony kiszámítását anélkül, hogy explicit módon invertálnánk egy $p \times p$ -s mátrixot. Főleg, ha mindkét mátrix szimmetrikus. Azonban $\mathbf{R}$ Toeplitz mátrix is, de Φ nem. Ennek eredményeképpen az autokorrelációs módszer egyszerűbb, mint a kovariancia módszer.

Feltételezve, hogy a Φ mátrix pozitív definit a szimmetriája lehetővé teszi a megoldást a négyzetgyökös vagy Cholesky felbontással, ami nagyjából megfelel a direkt mátrix invertálás technikához szükséges számítás- és helyigényt.

A további redundancia az $\mathbf{R}$ Toeplitz mátrixban lehetővé teszi a hatékonyabb Levinson-Durbin eljárást, amelyben a következő rendezett egyenlethalmazt rekurzív módon oldjuk meg $m = 1, 2, \dots, p$:

$$k_m = \frac{R(m) - \sum_{k=1}^{m-1} a_m(k)R(m-k)}{E_{m-1}}, \quad (1)$$

$$a_m(m) = k_m, \quad (2)$$

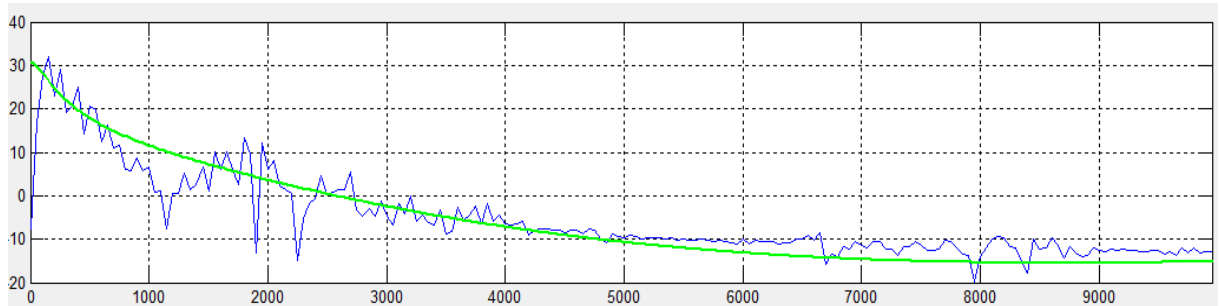
$$a_m(k) = a_{m-1}(k) - k_m a_{m-1}(m-k) \quad i \leq k \leq m-1, \quad (3)$$

$$E_m = (1 - k_m^2) E_{m-1}, \quad (4)$$

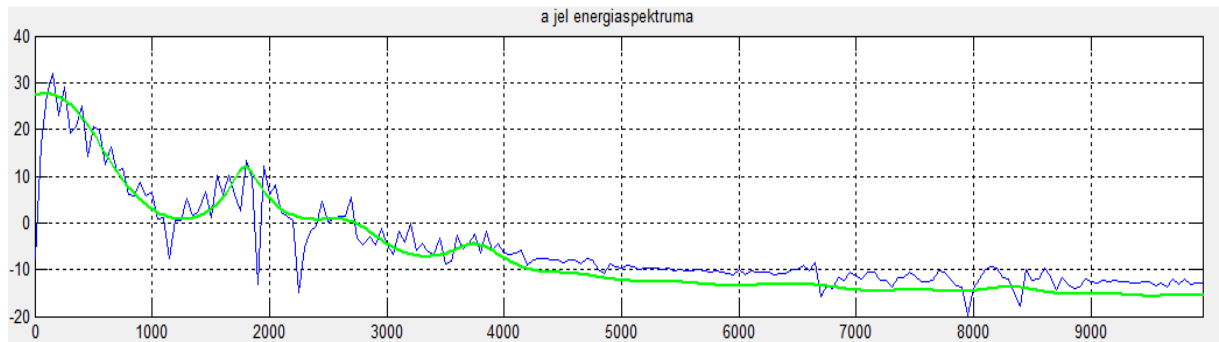
ahol kezdetben $E_0 = R(0)$ és $a_0 = 0$.

Példa a felhasználásra

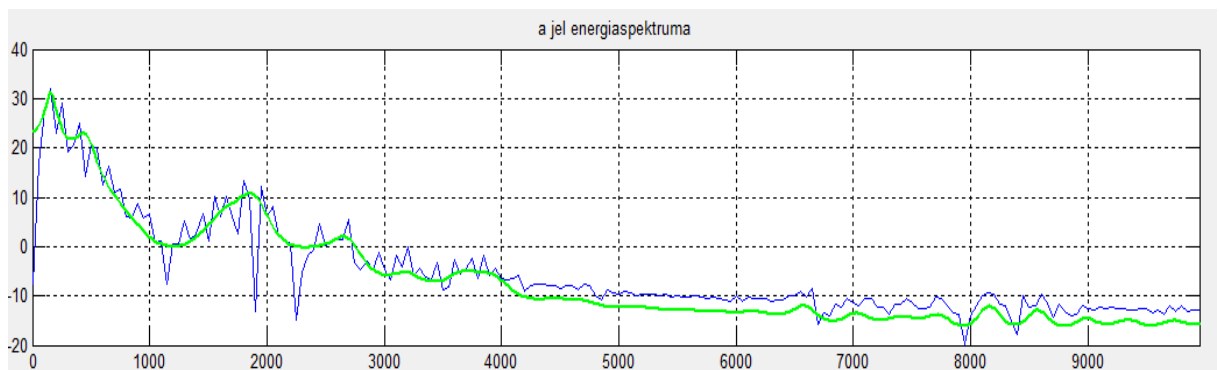
Az LPC-t használhatjuk a spektrum becslésére. A következő példák ezt fogják szemléltetni.



A fenti kép egy negyedfokú polinom segítségével közelíti a spektrumot. Az ábrán látszik, hogy a csúcsokra egyáltalán nem illeszkedik, csupán az ívet követi le.



Ez a példa már harmincad fokú polinomot használ. Látszik, hogy a zöld vonal a csúcsokat már szépen közelíti. Végül az utolsó ábrán nézzük meg ugyanezt hatvanad fokú polinommal.



MIDI Toolbox

A MIDI Toolbox Matlab függvények egy csoportját adja a kezünkbe, amelyek segítségével MIDI fájlokat tudunk feldolgozni és megjeleníteni. A MIDI a Musical Instrument Digital Interface rövidítéséből jött létre, ami zenei eszközök digitális csatolófelületét jelenti. Szintetizátorok és stúdióeszközök összekötésére alkalmas protokoll.

MIDI fájl beolvasása Matlabba

A MIDI Toolbox *readmidi* parancsa segítségével tudunk egy *.mid* kiterjesztésű fájlt beolvasni Matlabba.

```
hangok = readmidi('uj.mid');
```

A parancs végrehajtása után kapott *hangok* egy mátrix lesz, amiben az *uj.mid* nevű fájl adatai lesznek megtalálhatóak. Ez a mátrix 7 oszlopból áll, és annyi sorból, ahány hangesemény volt.

```
hangok =
```

0	0.2500	2.0000	43.0000	90.0000	0	0.2500
0.2500	0.2500	2.0000	50.0000	90.0000	0.2500	0.2500
0.5000	0.2500	2.0000	59.0000	90.0000	0.5000	0.2500
0.7500	0.2500	2.0000	57.0000	90.0000	0.7500	0.2500
1.0000	0.2500	2.0000	59.0000	90.0000	1.0000	0.2500
1.2500	0.2500	2.0000	50.0000	90.0000	1.2500	0.2500
1.5000	0.2500	2.0000	59.0000	90.0000	1.5000	0.2500
1.7500	0.2500	2.0000	50.0000	90.0000	1.7500	0.2500
2.0000	0.2500	2.0000	43.0000	90.0000	2.0000	0.2500
2.2500	0.2500	2.0000	50.0000	90.0000	2.2500	0.2500

A fenti ábra megmutatja a mátrix első néhány sorát. Az oszlopok különböző típusú információkra utalnak, mint például a MIDI hangmagasság. Az első oszlop jelzi az ütemekben a hang kezdetét, míg a második oszlop a hang időtartamát jelenti ugyanazon ütemértékben. A harmadik oszlop jelöli a MIDI csatornát, a negyedik pedig a már említett MIDI hangmagasságot, ahol az egyvonalas C a 60. Az ötödik oszlop a sebességet adja meg, tehát azt, hogy a hangot milyen gyorsan ütötték le. Az utolsó két oszlop pedig az első kettőnek felel meg kivéve, hogy itt másodpercben vannak megadva az adatok, nem pedig ütemben.

Természetesen nincs mindig szükségünk mind a 7 oszlopára a mátrixnak, hiszen lehet, hogy csak a hangmagasságokra vagyunk kíváncsiak. A Toolbox a kezünkbe ad függvényeket, amikkel csak egy-egy oszlopot tudunk elérni. Ilyen függvények például: *onset*, *dur*, *channel*, *pitch*, *velocity*. Az *onset* megadja az első oszlopot, ha *'beat'* a másik paramétere, valamint az utolsó előtti oszlopot, ha *'sec'*. A *dur* függvény a második oszlopot adja, ha *'beat'* a paramétere, és az utolsót, ha *'sec'*. A *channel*, *pitch* és a *velocity* pedig rendre a harmadik, negyedik és ötödik oszlop értékeit adják vissza.

Hivatkozás a mátrix egyes részeire

Sokszor a mátrixnak csak bizonyos részeit akarjuk elemezni. Ha például az első 10 sorra van szükségünk, akkor használhatjuk a Matlab beépített lehetőségeit is.

```
nmat = hangok(1:10, :);
```

Használhatjuk a MIDI Toolbox definícióit és függvényeit is:

```
elso_negy_masodperc = onsetwindow(hangok, 0, 4, 'sec');
```

Az *onsetwindow* visszatér azokkal a *hangok*ban tárolt hangokkal, amelyek kezdeti ideje kielégíti a

```
mintime <= onsettime[beats/sec](NMAT) <= maxtime
```

feltételt. A *mintime* az ablak minimum korlátja, ami alapértelmezés szerint ütésben van megadva, de átállítható másodpercre. A *maxtime* pedig az ablak maximum korlátja, ami alapértelmezés szerint szintén ütésben van megadva, de ez is átállítható másodpercre.

```
nmatf = dropshortnotes(nmat, 'beat', 1/16);
```

A *dropshortnotes* függvény kiszűri a rövid hangjegyeket. A fenti példa kiszűri az 1/16-nál rövidebb hangokat.

Hangjegymátrixok manipulálása

A beolvasás után előállt mátrixot tudjuk változtatni is. Lehetőség van például a tempó lekérdezésére és megváltoztatására is.

```
tempo = gettempo(hangok);  
tempo = 90;
```

```
ujtempo = settempo(hangok,tempo);
```

Sokszor szükség lehet a beolvasott dallamunk transzponálására. Ez azt jelenti, hogy a teljes dallamunkat mondjuk egy hangközzel eltoljuk.

```
neggyel_lejjobb = shift(nmat,'pitch',-4);
```

Ez a példa transzponálja a minta dallamunkat négy félhanggal lejjebb.

MIDI fájlok elmentése

Egy a *hangok*hoz hasonló mátrixot MIDI fájlá tudunk konvertálni a *writemidi* parancs segítségével. A szintaxis a következő:

```
writemidi(nmat,ofname,<tpq>,<tempo>,<tsig1>,<tsig2>);
```

Az *nmat* azt a mátrixot jelenti, amit a *readmidi* parancs eredményül ad. Az *ofname* a generált MIDI fájl nevét jelenti. A többi paraméter opcionális. Ezeknek van alapértelmezett értéke.

A hangjegymátrixok lejátszása

Két módszer van arra, hogy egy mátrixot megszólaltassunk. Az első módszer a következő:

```
playmidi(hangok);
```

A másik pedig a

```
playsound(hangok);
```

MIDI adatok megjelenítése

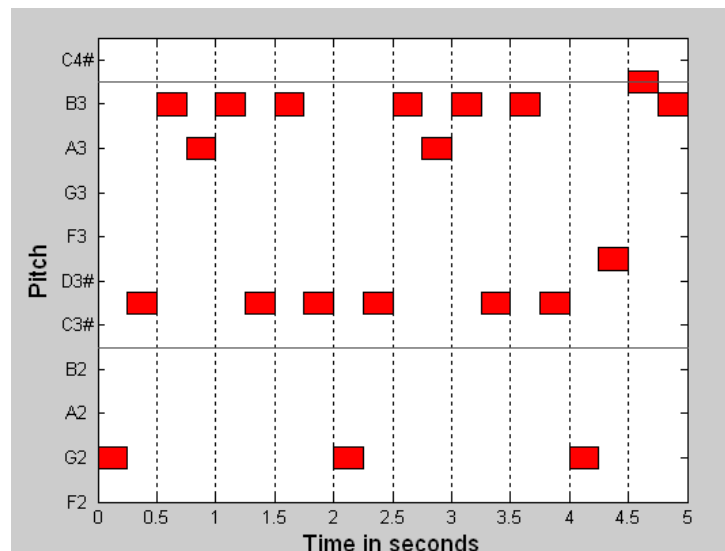
```
pianoroll(nmat,<varargin>);
```

Az első argumentum a hangjegymátrixra utal, a többi argumentum pedig opcionális. Opcionális input argumentumok például:

- 'name' = hangjegyek neve az y tengelyen (alapértelmezés)
 - 'num' = MIDI számok az y tengelyen
- 'beat' = ütések az x tengelyen (alapértelmezés)
 - 'sec' = másodperc az x tengelyen
- 'vel' = megjeleníti a hangok sebességét
 - 'mh' = metrikus hierarchiát jelenít meg

```
pianoroll(nmat, 'name', 'sec');
```

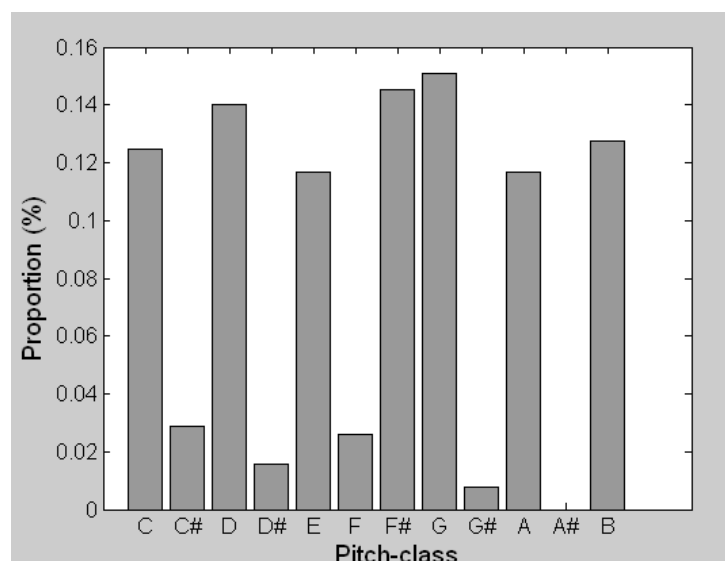
A fenti példa megjeleníti a hangmagasságokat és a sebességeket úgy, hogy az y tengelyen a hangmagasságok nevei jelennek meg, míg az x tengelyen másodpercek.



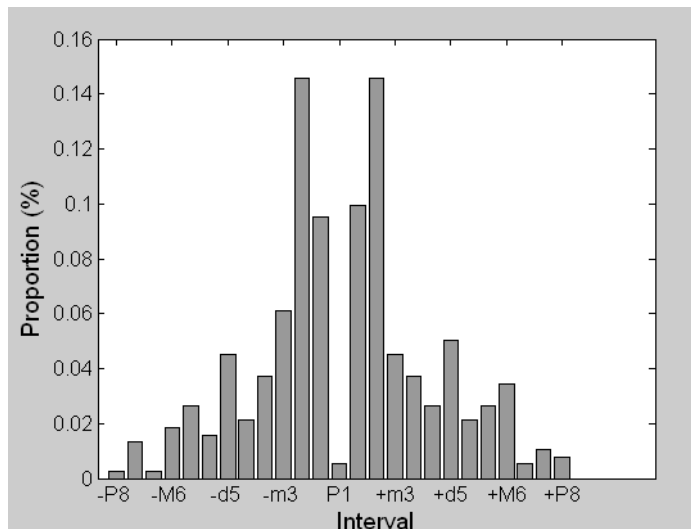
Ezen az ábrán a vízszintes vonalak a C-ket jelölik, míg a függőleges vonalak az ütések kezdetét jelentik.

```
plotdist(pcdist1(hangok));
```

A *pcdist1* függvény kiszámolja a *hangok* mátrixban lévő hangmagasságok eloszlását azaz, hogy milyen arányban szerepelnek a mátrixban a kromatikus skála hangjai. Ezután a *plotdist* parancs fog ebből a kimentből egy ábrát készíteni.



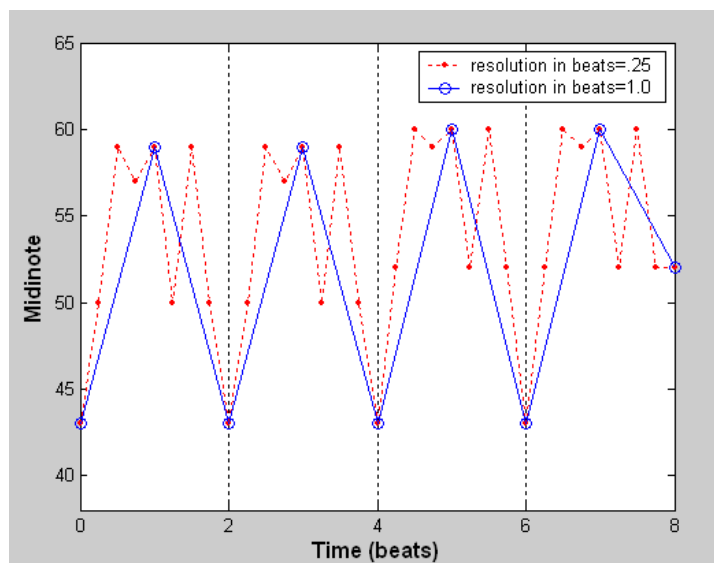
```
plotdist(ivdist1(hangok));
```



A fenti ábra azt mutatja meg, hogy milyen típusú intervallumok szerepeltek a *hangok* nevű mátrixban, és melyik milyen arányban. P1 a tiszta primet, M3 a nagytercet és m3 a kistercet jelenti meg. Az *ivdist1* függvény visszatér a *hangok* hangjegymátrixban található hangközök eloszlásával egy 25 elemű vektorként. A vektor elemei félhang távolságra vannak egymástól. Az első komponens reprezentálja lefelé az oktávot, és az utolsó komponens pedig felfelé az oktávot.

A dallam kontúrja megjeleníti a dallam teljes alakját.

```
nmat = hangok(1:32, :);
plotmelcontour(nmat, 0.25, 'abs', ':r. '); hold on
plotmelcontour(nmat, 1, 'abs', '-bo '); hold off
```



A felbontás foka függ a mintavételi lépés értékétől tehát, hogy milyen időközönként veszünk adatot a mátrixból. Minél ritkábban, azaz minél nagyobb a felbontás, annál durvább lesz a kontúr. Az ábrán a szaggatott vonal egy részletesebb kontúrt reprezentál 0.25 lépésű felbontással. A folytonos vonal egy elnagyoltabb kontúrt mutat be, ami sokkal hasznosabb lehet, ha a dallam teljes struktúráját tanulmányozzuk.

A MIDI Toolbox lehetőséget biztosít a beolvasott dallam hangnemének a megkeresésére is. „A K-S hangnemkereső algoritmusban a 24 különböző hangnem profil, 12 dúr és 12 moll, korrelálva van a mű hangmagasság osztály eloszlásával a hosszuknak megfelelően súlyozva.”

```
nmat = hangok(1:32, :);  
keyname(kkkey(nmat))
```

A fenti parancsok végrehajtása után az *ans* értéke 'G' lesz. Ami helyes eredmény, mivel a beolvasott dallam hangneme valóban G dúr volt. Ugyanakkor, ha csak az első 16 hangra végzem el a vizsgálatot, akkor az *ans* értéke 'b', ami már nem jó eredményt ad. A belső függvény (kkkey) végrehajtja a K-S hangnemkereső algoritmust, míg a külső függvény a belső numerikus kimenetét megváltoztatja a kapott hangnem betűjére. Így a kimenet sokkal informatívabbá válik.

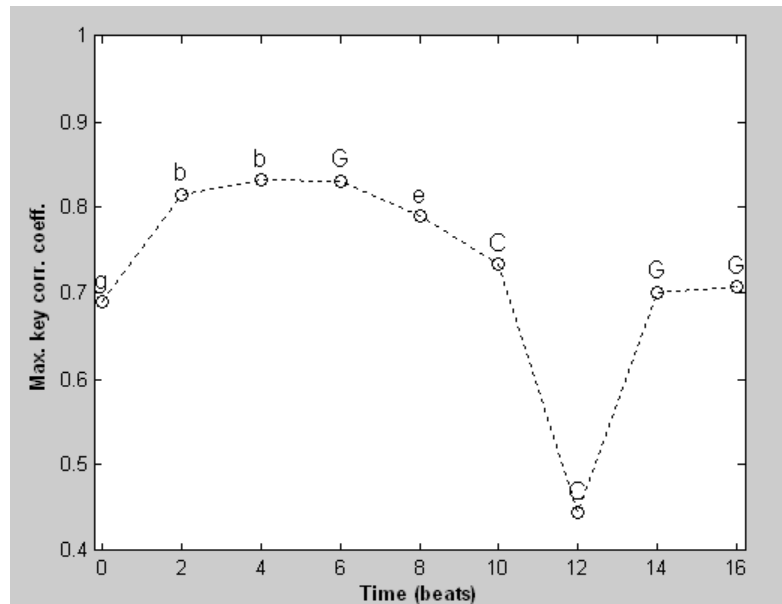
Egy másik lehetőség a hangnem megtalálására az, hogy szemléltetjük a tonalitás időbeni változását. Ha ezt a technikát alkalmazzuk, akkor a hangnem megkeresése mindig egy kis ablakban történik, és ez az ablak megy végig a zene teljes hosszán. A megvalósításhoz egy lehetséges eszköz a *movewindow* függvény.

```
y = movewindow(nmat, wlength, wstep, timetype, varargin);
```

Alkalmazza a *varargin*ban definiált függvényeket ablakolt hangjegymátrixok egy sorozatára. A *wlength* az ablak hosszát, a *wstep* a lépésközt, a *timetype* az időreprezentáció, ami 'beat' (alapértelmezés) szerint vagy 'sec', ha megváltoztatjuk ezt az alapértelmezést, míg az *nmat* a mátrixot jelenti. A megkapott *y* pedig a *varargin* függvény kimenete alkalmazva az *nmat* mátrixra.

```
nmat=onsetwindow(uj, 0, 16, 'beat');  
keys = movewindow(nmat, 4, 2, 'beat', 'maxkkcc');
```

Ebben az utasításban a legnagyobb hangnem-korreláció megtalálásához egy négy ütés hosszúságú ablakkal megyünk végig a dallam teljes hosszán. Ez az ablak az időben mindig két ütéssel mozdul el, és a *maxkcc* függvényt hajtja végre az ablakolt hangjegymátrixon.

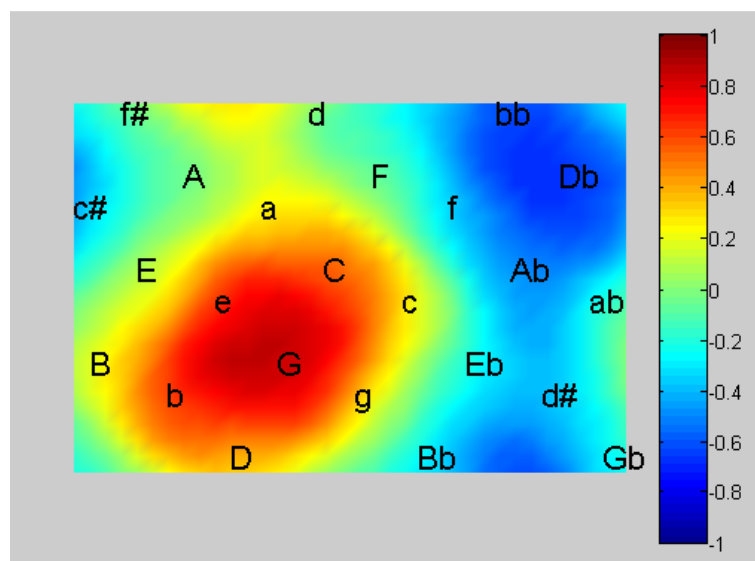


A kapott ábra megmutatja a hangnemváltozásokat időről időre.

A következő példában a függvény kiszámolja a mátrixban található hangmagasságok eloszlását, és létrehoz egy szemléletes ábrát belőlük.

```
keysom(nmat, 1);
```

A második argumentum a példában azt adja meg, hogy legyen-e feliratos színskála (1) vagy ne. Harmadik argumentumként pedig azt adhatjuk meg, hogy színes vagy szürke árnyalatos ábrát szeretnénk látni.



Lilypond

A Lilypond egy nyílt forrású kottaszerkesztő program. Bemenete egyszerű szöveges fájl, ami minden szükséges információt tartalmaz. A program licence GNU General Public License, míg a dokumentációé GNU Free Documentation License, ezáltal mindenki szabadon javíthatja, módosíthatja és bővítheti azt.

A Lilypond alapvető funkcióinak bemutatása

A kotta létrehozásához a forrásfájlba kell azt szöveges formában bevinni, majd a „fordítás” során előáll a nyomtatható kotta (pl.: pdf vagy ps formátumban), illetve egy midi fájl, ha a szükséges parancsot megadtuk a generálásához.

Hangmagasság

A hangmagasság megadásának legegyszerűbb módja a `\relative` mód. Ebben a módban minden hang automatikusan abba az oktávba kerül, ami az őt megelőző hanghoz a legközelebb van. A megértéshez szükséges a különböző hangközök tisztázása. Két hang egy hangközt alkot. Ez lehet prim, szekund, terc, kvart, kvint, szext, szeptim és oktáv. A prim ugyanannak a hangnak az újbóli leütését jelenti, a szekund egy egész hang távolságot jelent a két hang között, a terc már kettőt, és így tovább. Az oktáv a skála két szélső hangja közötti távolság: c-től c-ig vagy akár d-től d-ig ugyanannyi.



Lehet azonban egy terc kisterc és egy kvint szűkített kvint is, ha félhanggal kisebb a két hang között a távolság. Ugyanezen az elven beszélhetünk bővített kvintről is, ha félhanggal nagyobb a távolság. A kottában a szűkített vagy bővített jelleget pedig a módosítójelek adják meg. A `\relative c' { }` azt adja meg, hogy az első viszonyítási hang az egyvonalas c, és minden hang az előzőhöz a lehető legközelebb helyezkedik el, tehát legfeljebb kvart távolságra. Természetesen a `c'` helyére írhattunk volna `c''` is, akkor a kétvonalas c lesz az első viszonyítási hang. A következő példa szemlélteti, hogy hogyan működik a `\relative` mód. Tehát, ha kap egy b és egy d hangot, akkor megvizsgálja, hogy a b és a felette lévő d között

mekkora a távolság (terc), majd megnézi, hogy a b és az alatta lévő d közötti távolság milyen nagy (szext). Ezek után a kisebb távolságra lévőket fogja megjeleníteni.

```
\relative c'' {
  b c % felfelé szekund, lefelé szeptim, így: felső c
  b d % felfelé terc, lefelé szext, így: felső d
  b e % felfelé kvárt, lefelé kvint, így: felső e
  b a % felfelé szeptim, lefelé szekund, így: alsó a
  b g % felfelé szext, lefelé terc, így: alsó g
  b f % felfelé kvint, lefelé kvárt, így: alsó f
}
```



Azonban a kottában lehet szükség ennek az alapértelmezésnek a felüldefiniálására. Ezt úgy tudjuk megtenni, hogy a hang után írunk egy ' vagy , karaktert. Az ' egy oktávnyival feljebb rajzolja a hangot, mint az alapértelmezés, még a , egy oktávval lejjebb.

```
\relative c' {
  c d e f
  c d' e f,
}
```



Ritmusértékek

Ritmus vagy ritmusérték alatt az egyes hangok időbeli terjedelmét értjük. A használt fő ritmusértékeink: fél, negyed, nyolcad, tizenhatod, harmincketted stb. A Lilypondban a hangok ritmusértékét a hangmagasságok után írt szám segítségével adhatjuk meg. Az 1 jelenti az egész hangot, a 2 a félhangot, a 4 a negyedhangot stb. A gerendázás, azaz a hangjegyek összekötése automatikusan történik. Ha nem adunk meg egy hangmagasság után explicit hosszúságot, akkor az előző hang hossza marad érvényben. Valamint, ha semmit nem adunk meg, akkor a hangok alapértelmezetten negyed hosszúságúak.

```
\relative c' {
  c1 d4 e f g a2 b2 c1
}
```





Ütemmutatót a `\time` paranccsal adhatunk meg. Pl.: `\time 3/4` vagy `\time 4/4`. A 4/4-edet a Lilypond automatikusan helyettesíti az eddig a példákban is látott C-vel. Ütemmutató megadásával megmondjuk, hogy milyen ritmusérték adja az ütem alapegységét, továbbá azt is megmondjuk, hogy ebből az alapegységből hány darab adja ki az ütem teljes hosszát. Kulcs beállítása pedig a `\clef` paranccsal történik. Az alapértelmezett a violinkulcs, de át lehet állítani. Pl.: `\clef bass` a basszuskulcs.

Módosítójelek

A kereszt, illetve a bé módosítójelet a hang neve után írt `-is` illetve `-es` utótag segítségével tudjuk megadni. Természetesen megadhatunk kettős bét és kettős keresztet is az `-eses` és az `-isis` utótagok segítségével.

```
\relative c' {
    des ais' c, d eses fis
}
```



Lehetőség van a `\key` paranccsal előjegyzés megadására is, amely parancs tartalmazza a hangnem alaphangját, valamint azt, hogy dúr vagy moll.

Ívek

Az elkészített kottában csak kötőívet és átkötést használtam, így csak ezeket az eszközöket mutattam be. Az átkötés egy hang hosszát megnöveli a következő hangéval, aminek ugyanolyan magasságúnak kell lennie. Átkötött hangot a hang után írt `~` jel segítségével kapunk. A kötőív pedig egy dallamot fog össze. A dallam kezdőhangja után írt `(` és az utolsó hangja után írt `)` segítségével adjuk meg.

```
\relative c' {
    c e( c'2 ~ c4 a g f) e ~ e8 g g g c d
}
```



Dinamika és egyéb jelölések

Az egyes hangoknál megadhatunk dinamikai jelzéseket a hang után írt `\ff`, `\p`, `\mf`, `\pp`, `\f` parancsok segítségével. Tehát a szintaxis úgy néz ki, hogy `\` után írjuk a megfelelő rövidítést. Crescendo és decrescendo a `\<`, illetve a `\>` parancsokkal kezdődik, és hatásköre a `\!` parancsig tart.

```
\relative c' {
  c d\< e f\f\> g a b\! c
}
```



Egyéb jelölések közé tartozik az ujjrend megadása. Ujjrendet csak bizonyos hangoknál szoktunk jelölni, és az értékek automatikusan a hang jó oldalára kerülnek. Természetesen ez az alapértelmezés itt is felüldefiniálható. Ujjrendet a hang után írt - és egy szám segítségével jelölünk. Továbbá itt kell megemlíteni az autentikációs jelzéseket is, amiket szintén a - után írt megfelelő karakterrel jelenítünk meg. Ez a karakter tenuto esetén a `-`, míg staccato esetén a `.` írható ki. Ha mi akarjuk megadni azt, hogy hol legyenek ezek a jelzések, akkor a - helyett `^` kell, ha felette, míg `_`, ha alatta szeretnénk látni az adott jelzést a kottában.

```
\relative c' {
  c_1 d^2 e f-1 g a^> b-4-. c-.
}
```



Adhatunk még az egyes hangokhoz szöveges utasításokat is. Ha ezt a szöveget formázni is szeretnénk, akkor a `\markup` utasítást kell használnunk. A formázó parancsokat és a szöveget is a `\markup` utáni `{ }`-ek között tudjuk megadni. Ilyenek például: `\bold`, `\italic`, `\small` stb. Továbbá lehetőségünk van megválasztani azt is, hogy a hang felett vagy alatt jelenjen meg a kívánt szöveg, amit a már korábban említett módon tehetünk meg.

```
\relative c' {
  gis'-4_\markup{ \bold \dynamic p } d-0( f-2 e)
}
```



Abszolút hangmagasság-megadás

Az előző példákban a `\relative` módban voltak megadva a hangmagasságok, de lehetőség van abszolút hangmagasságok megadására is. Ekkor a `c''` minden esetben a kétvonalas `c` hangot fogja jelölni, míg a `c'` az egyvonalas `c` hangot. Abszolút hangmagasság használatakor átláthatatlanabbá válik a forrás, mert nagyon sok `'` megadására van szükség. Így sokszor a `\relative` mód célravezetőbb lehet.

```
{  
  \key f \major  
  \autoBeamOff  
  c''8[( bes')] r8 bes'8 \grace c''16 bes'8[ a'16 bes']  
  \grace c'' bes'8[ a'16 bes']  
}
```



A `\grace` parancs segítségével előkét tudunk megadni. A fenti példában két `c` hangot is előkeként jelenítettünk meg.

Automatikus és kézi gerendázás

Amennyiben nem adjuk meg másképp, akkor a Lilypond a gerendázást automatikusan elvégzi helyettünk. Ha csak egyszeri alkalommal szeretnénk felüldefiniálni az automatikus gerendázást, akkor a gerenda első hangja után [, míg az utolsó hangja után egy] kell tennünk. Természetesen lehetőség van az automatikus gerendázás teljes kikapcsolására is. Ezt az `\autoBeamOff` utasítással tehetjük meg, míg az `\autoBeamOn` újra automatikussá teszi a gerendázást.

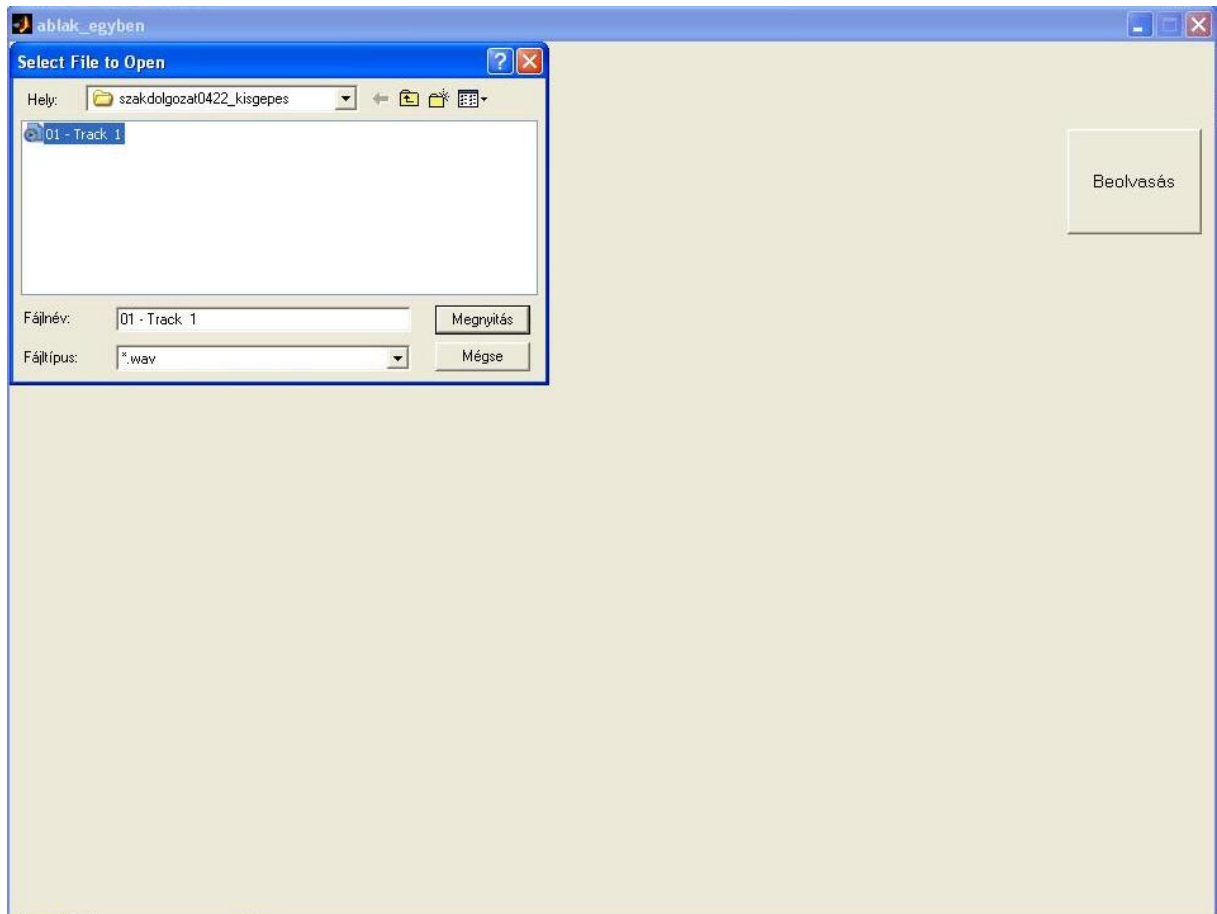
Transzponálás

Egy zenei kifejezés transzponálása a `\transpose` paranccsal történik, aminek a szintaxisa a következőképpen néz ki: `\transpose frompitch topitch { }`. A zárójelek között megadott zenei kifejezés kerül transzponálásra a `frompitch` és a `topitch` hangmagasságok közötti hangközzel. Minden `frompitch` hangmagasságú hang így `topitch` magasságúra fog változni, és minden más hang a megadott hangközzel lesz transzponálva. A parancsban megadott hangmagasságok abszolút módban értendők.

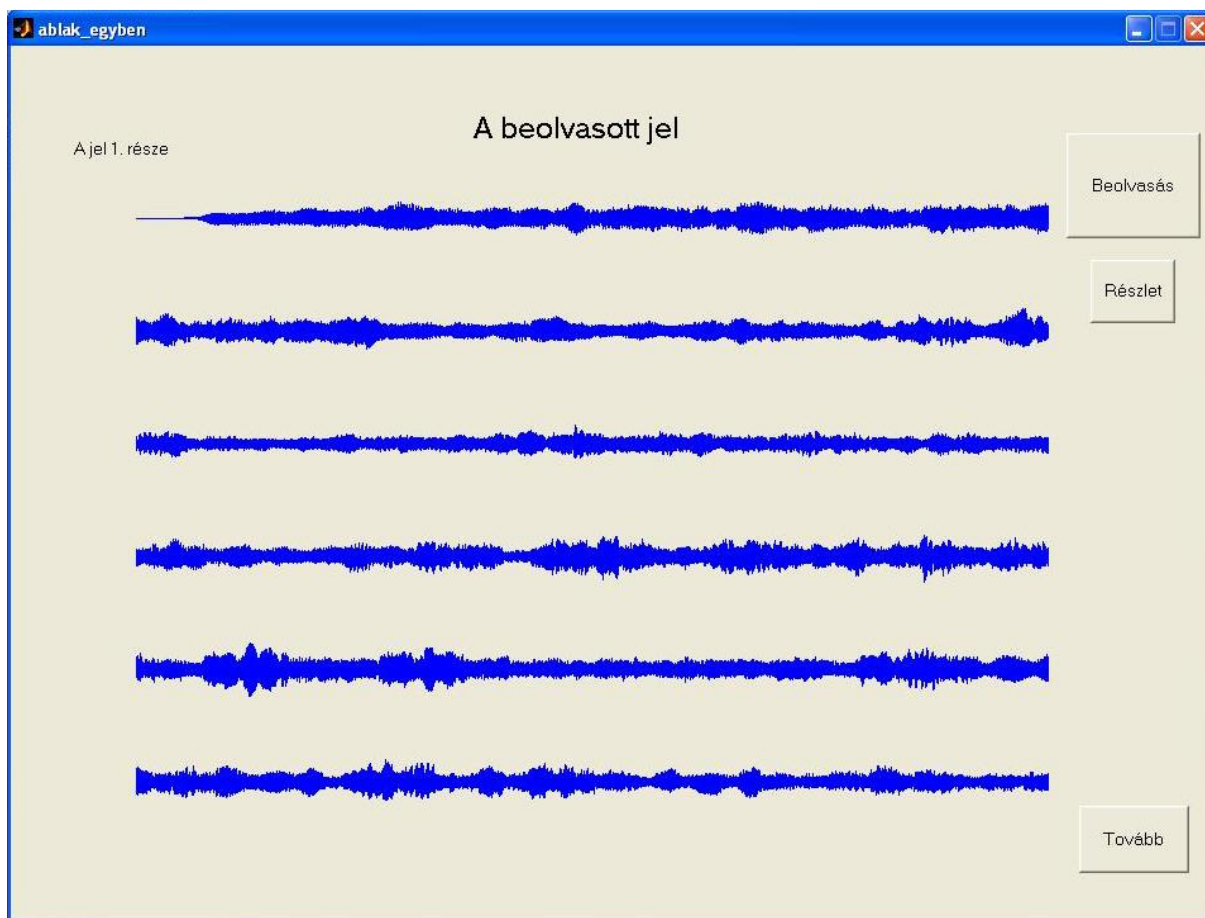
48

Az elkészített program bemutatása

A szakdolgozat keretében egy olyan programot készítettem el, ami képes egy hangszerezen lejátszott dallamban a hangmagasságok meghatározására. Az alkalmazás Matlab 6.5 környezetben készült, és a helyes működés a MIDIToolbox feltelepítését is megköveteli. Az alkalmazás erőssége, hogy nem csupán a kapott wav fájlt dolgozza fel, hanem vele párhuzamosan egy az általam megírt kottából generált MIDI fájlt is. Mindkét fájlról több szemléletes grafikont jelenít meg, amivel még átláthatóbbá teszi az egyes ütemek elemzését. Továbbá az ábrákon kívül még megjeleníti az elemzett ütem kottaképét, valamint lehetővé teszi annak lejátszását is.

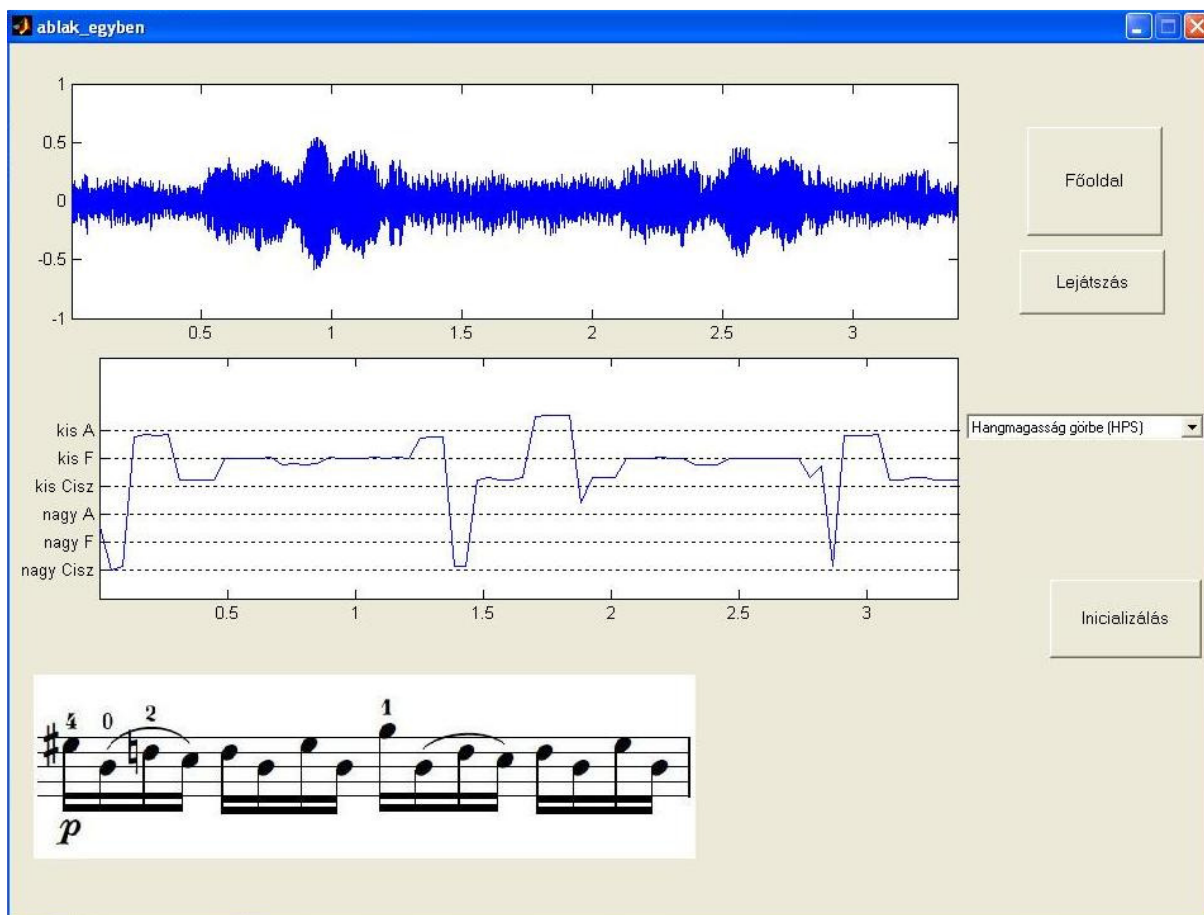


A Beolvasás gombra kattintva kiválaszthatjuk az elemezni kívánt hangfájlt, aminek a képe rögtön meg is jelenik. A program csak wav kiterjesztésű fájlokat képes feldolgozni. A kép közepén egy koordináta-rendszer látható, amelyben soronként történik az adatok kirajzolása. Az egymás alatt ábrázolt sorok folytonosan követik egymást a fájlban. A jobb átláthatóság érdekében kerültek egymás alá, valamint így a zene egy nagyobb részletére kapunk rálátást.



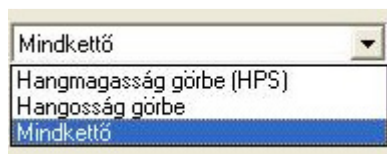
A cím mellett bal oldalt látható, hogy a műben éppen hányadik nagyobb egységnél tartunk. Az általam elemzett zenemű Johann Sebastian Bach által írt 6 csellószvit közül a G-dúrnak a nyitótétele, azaz a Prelúdium tétele. A mű kicsit több mint 3 perc hosszúságú, ami 44100 Hz-es mintavételezési frekvencia mellett több mint 6 millió mintát jelent. Ez a mennyiség egyszerre már elég nehezen kezelhető és feldolgozható. Ezért 3 részre lett osztva az adathalmaz, és azok vannak az egyes oldalakon megjelenítve.

A Tovább gombra kattintva tudunk átlépni a következő részre, ahol már megjelenik egy Vissza gomb is. Így már könnyen navigálhatunk az oldalak között azért, hogy az általunk elemezni kívánt részt ki tudjuk választani. Ezt a Részlet gombra kattintva tehetjük meg, ami egy célkereszt megjelenését eredményezi. A megfelelő részlet kijelölése úgy történik, hogy rákattintunk a célkereszttel, majd leütjük az Enter billentyűt. Ekkor megtörténik a koordináták leolvasása és átkerülünk az elemzéseket tartalmazó oldalra. Ekkor a kezdőoldal, ahol eddig jártunk, letörlődik és megjelenik helyette az új koordinátarendszereket tartalmazó oldal, mint ahogy azt az alábbi ábra is mutatja.

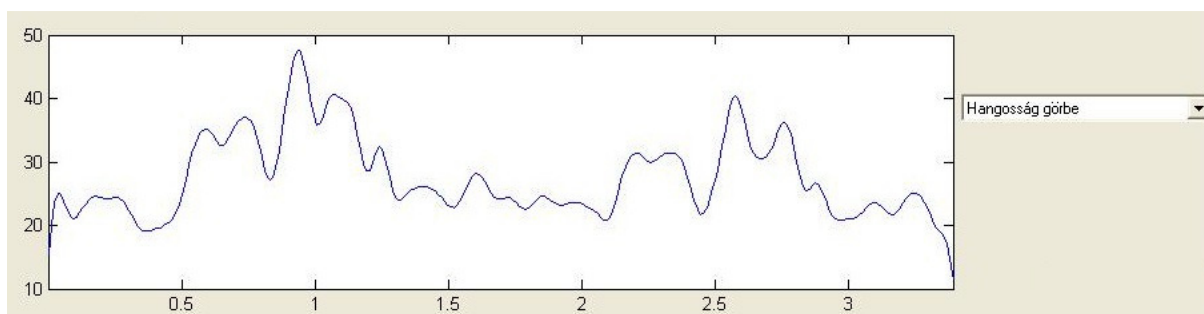


A legfelső koordináta-rendszerben az adott ütem van kirajzolva, amit a Lejátszás gombra kattintva meg is tudunk hallgatni. A középső koordináta-rendszerben az elkészült hangmagasságbecslés látható. A függőleges tengelyen a feliratok mutatják az egyes hangok helyét. Az egyes hangok neve előtt még ott állhat a kontra, nagy, illetve kis jelző is. Mivel az egyvonalas hangok után következnek a kis, majd a nagy, a kontra és legvégül a szubkontra hangok. Azért lett így megjelenítve, mert különben nem látjuk át, hogy pontosan milyen tartományban is jár a becslés. A hangmagasság detektálása HPS algoritmussal történt, ami a Harmonikus szorzat spektrumnak a rövidítése. A HPS algoritmus előtt növelni kell az alacsony frekvencia felbontást, a jel nullákkal való kitöltésével. Az eljárás méri a maximális egybeesést a harmonikusokra vonatkozóan. Egy 2048 minta hosszúságú ablakkal megy végig az ütemen, majd minden ilyen ablakra vonatkozóan meghatároz egy maximumot, ami egy frekvenciaérték lesz. A legalsó koordináta-rendszer az adott ütem kottaképét tartalmazza. A kotta Lilypondban lett elkészítve és a teljes kottából lettek az egyes ütemek kivágvva. Tehát ha egy másik ütemet választunk ki elemzésre, akkor annak a hangmagasság becslése és kottaképe fog megjelenni a koordináta-rendszerekben.

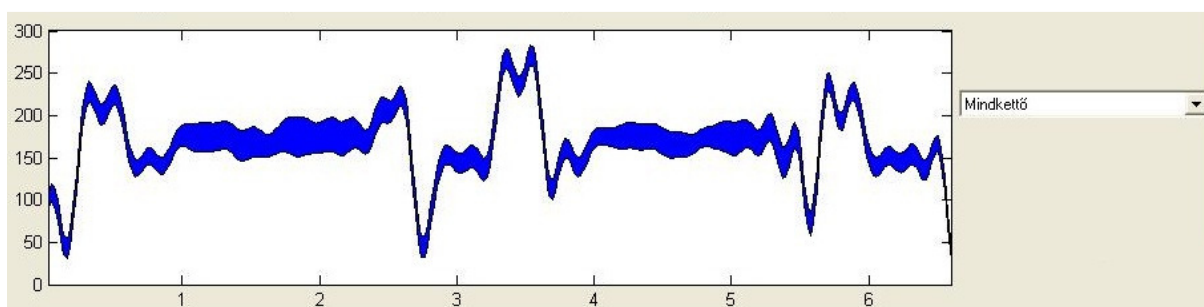
A képernyő jobb oldalán található egy lenyíló lista, ami a következőket tartalmazza:



Ha a hangmagasság görbe van kiválasztva, akkor a fenti képen bemutatotthoz hasonló ábrát kapunk. A hangosság görbe kiválasztása után a következőt láthatjuk:



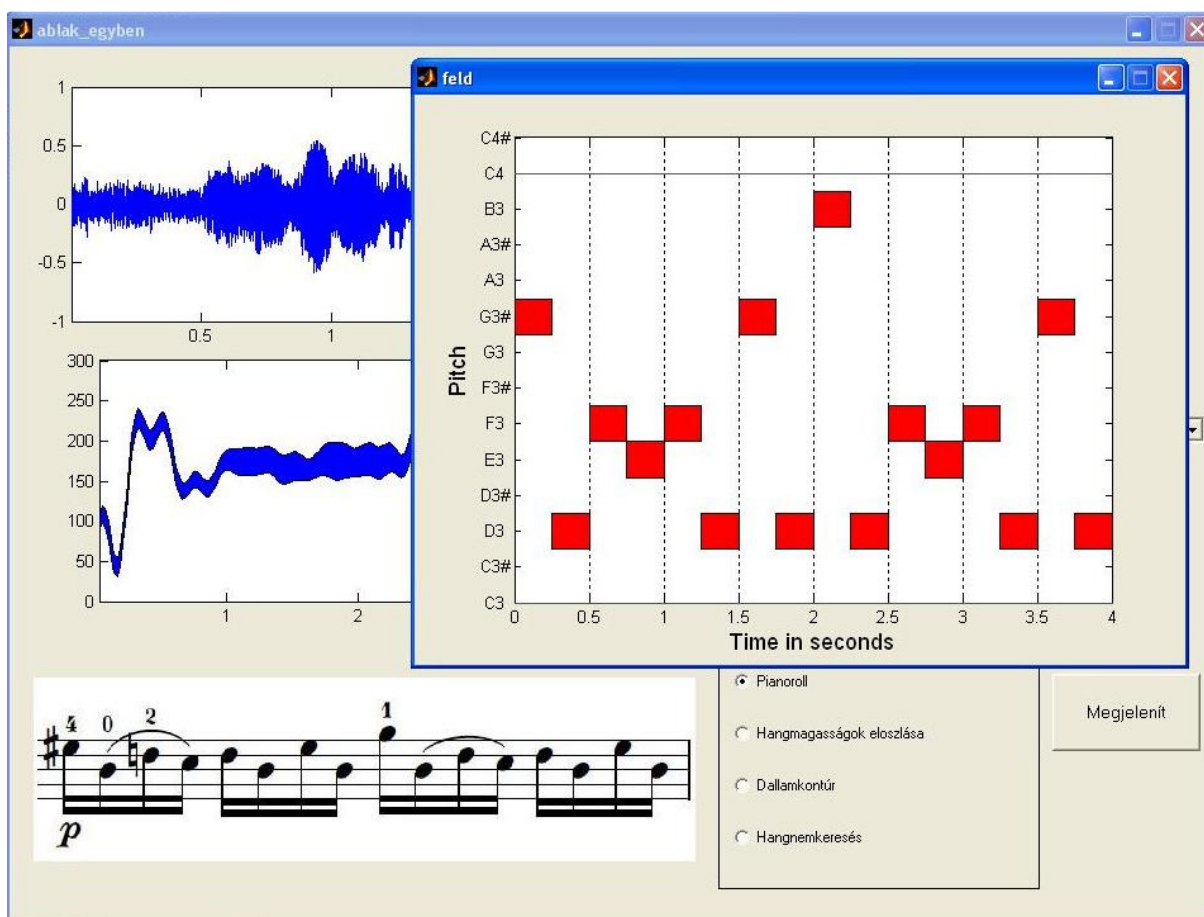
Valamint van még egy lehetőség a lenyíló listában, és ez a Mindkettő. Ha ezt választjuk ki, akkor egy olyan ábrát kapunk, ami a hangmagasság görbét jeleníti meg, de a hangosság görbe függvényében. Tehát ahol hangosabban szól a hang, ott vastagabb lesz a görbe, míg ahol halkabban, ott vékonyabb. Ennek a görbének a megjelenítésére a Matlab egyik beépített függvényét használtam, amit patch-nek hívnak.



Az Inicializálás gombra kattintva kapjuk meg a MIDI fájl elemzését végző panelt. A gombra kattintva háttérfolyamatok kezdődnek el, azaz beolvasásra kerül a kottából generált MIDI fájl. Ez ebben az esetben is elég nagy mennyiségű adatot jelent, aminek az egyben történő feldolgozása bonyolult és lassú lenne. A beolvasott adathalmazból tehát kiválasztásra kerülnek az adott ütem adatai, amelyekből elkészül egy olyan fájl, ami csak ezt az egy ütemet tartalmazza. Majd ennek a beolvasásával kezdődhet meg az elemzés. Az összes művelet a MIDIToolbox függvényei segítségével történik.

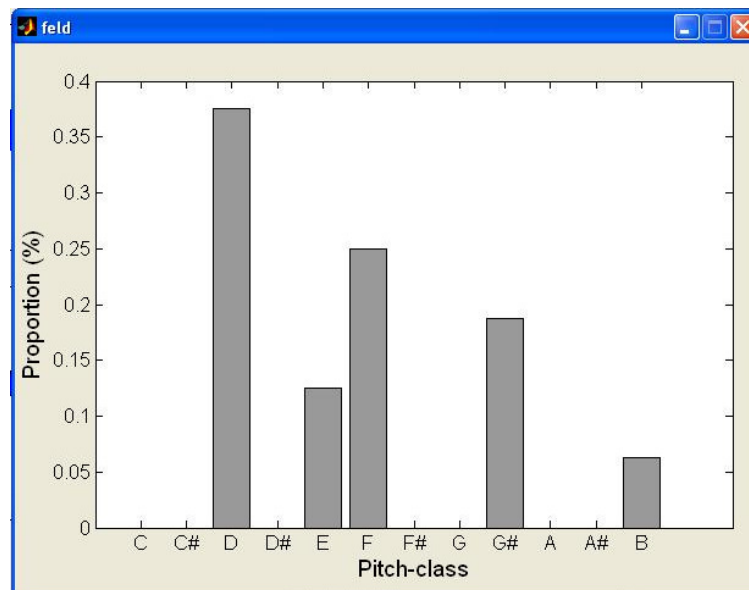


A MIDI panelen 4 rádiógomb található. Az első a pianoroll, ami annyit tesz, hogy megjeleníti az ütem hangjait egy feldobott ablakban található koordináta-rendszerben. Pontosan le tudjuk olvasni az ábráról az egyes hangok nevét (G, Disz, stb.), valamint azt is, hogy milyen hosszan szóltak. A vízszintes vonal az ábrán a C helyét jelöli ki, természetesen több is megjelenhet.

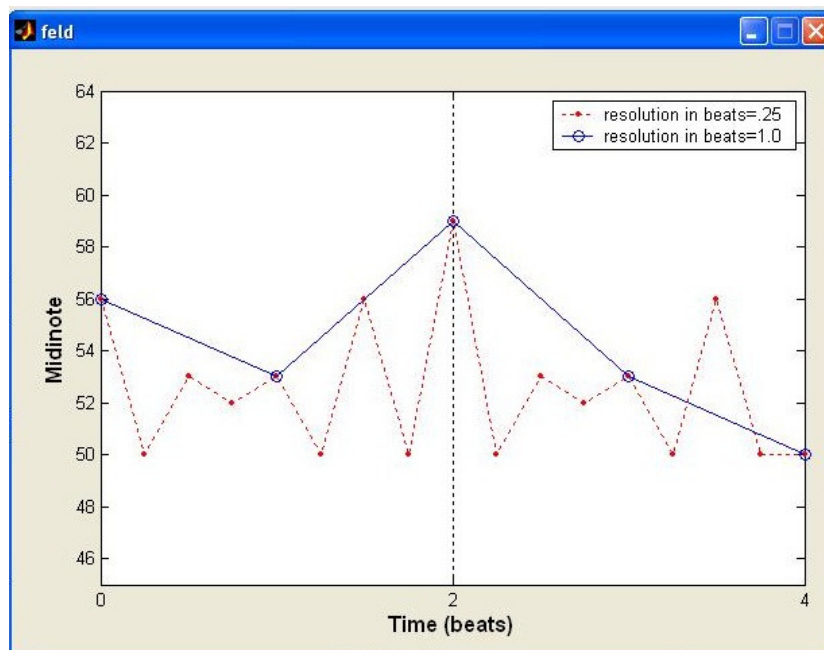


A rádiógombok hatását a Megjelenít gombra kattintva kaphatjuk meg. Egyszerre természetesen csak egy gomb lehet aktív a négy közül.

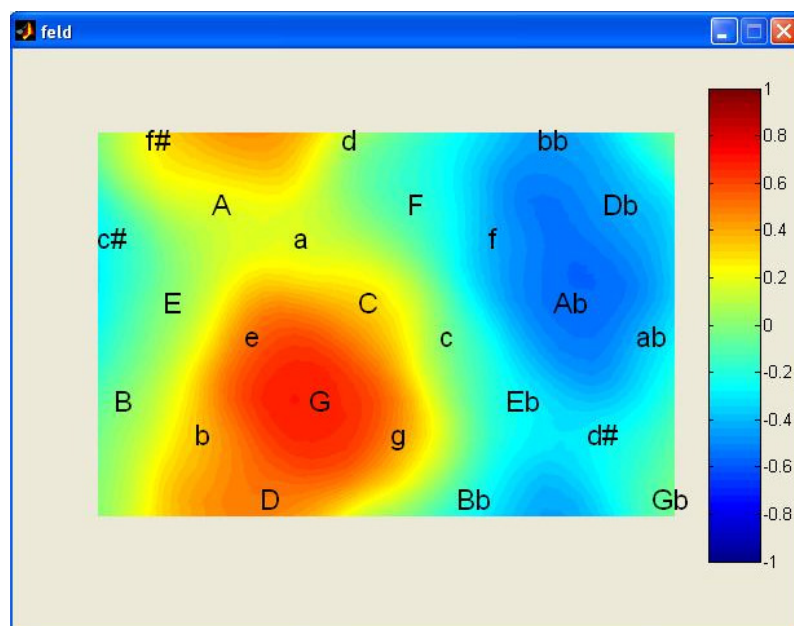
A következő gomb a hangmagasságok eloszlása, ami egy oszlopdiagramot ad eredményül. Erről le tudjuk olvasni, hogy milyen hangok szerepeltek az ütemben, és milyen arányban.



Amint az ábra is mutatja az elemzett ütemben csak D, E, F, Gisz és H hangok szerepeltek. Ezek közül is D-ből volt a legtöbb. A dallamkontúr gomb segítségével az ütem dallamvonalát kapjuk meg. Az eredményül kapott ábra két vonalat tartalmaz. Az egyikben minden hang szerepel, és az értékek összekötéséből kapunk egy részletesebb dallamvonalat. A másik vonal csupán minden negyedik hangot ábrázol, amivel így durvább kontúrt mutat. Ez sokszor hasznosabb lehet számunkra.

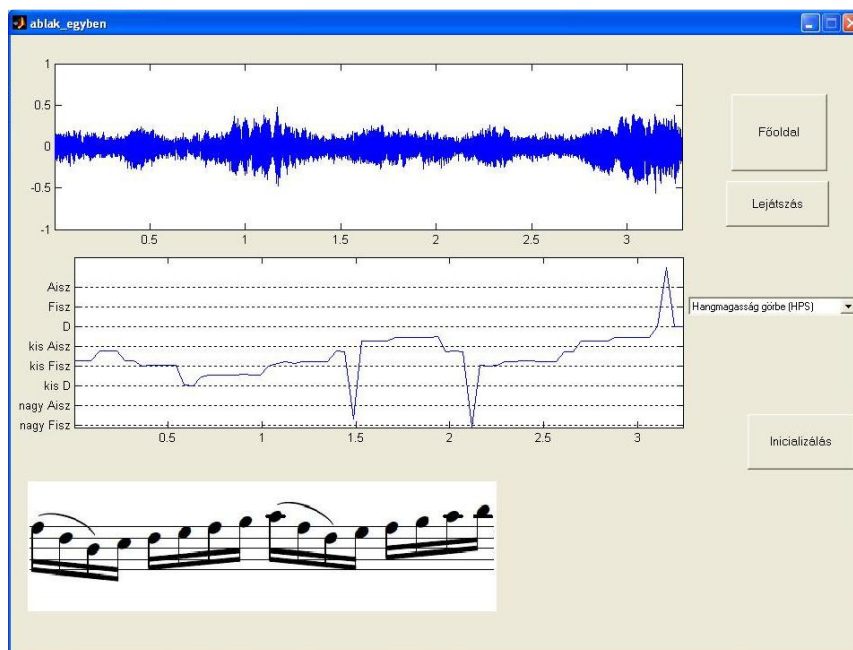


Az utolsó gomb pedig a hangnemkeresés. Ezt választva egy színes ábrát kapunk arról, hogy az ütem milyen hangnemű lehet.



A piros szín mutatja a legvalószínűbbként meghatározott hangnemet, ami a fenti képen éppen a G-Dúr.

Miután mindent megnéztünk a választott ütemről, visszamehetünk az előző oldalra a Főoldal gombra kattintva, és kiválaszthatjuk a következő elemezni kívánt ütemet. Az alábbi ábrán pedig már egy másik ütem adatai és becslése látható.



Összefoglalás

Dolgozatom célja a különböző hangmagasság algoritmusok áttekintése, és egy olyan program megírása volt, amely képes gordonkán lejátszott művekben a hangmagasságok meghatározására. A fenti fejezetek során bemutatásra kerültek a téma megértéséhez szükséges alapfogalmak, valamint a fontosabb hangmagassági modellek és egy hangossági modell is.

Először kifejtettem a digitális jelfeldolgozáskor felmerülő azon alapfogalmakat, amelyekre a későbbi fejezetekben is hivatkoztam. Ezután részleteztem a különböző hangmagassági modelleket, amik közül a harmonikus szorzat spektrum algoritmust implementáltam és a programba is beépítettem. Továbbá bemutattam egy hangossági modellt a témakör alapfogalmaival alátámasztva. Majd az általam használt parancsokon keresztül szemléltettem egy szöveges kottaíró programot, amiben megírtam a tanulmányozott mű kottáját. Ebből készült egy MIDI fájl, amit a MIDIToolbox függvényei segítségével az elemzett hanggal párhuzamosan dolgoztam fel, amitől sokkal szemléletesebbé vált az egyes ütemek vizsgálata. Végül az elkészült alkalmazás leírását olvashattuk, ami képernyőképekkel illusztrálva szemlélteti a program működését.

A programnak számos továbbfejlesztési lehetősége van. Mivel az alkalmazás jelenlegi formájában elég sok kötöttséget tartalmaz, ezért az első feladat ezek eltünttetése lenne. Tehát bármilyen wav fájlra tudjon dolgozni és részekre darabolni. Továbbá több különböző hangmagasság algoritmus működésének összehasonlításával meg lehetne állapítani, hogy melyik milyen esetekben működik jobban. Ezen eredmények alapján több algoritmus kombinációjából egy hatékonyan működőt lehetne létrehozni. Egy már jól működő algoritmus pedig lehetőséget nyújtana egy olyan szoftver megírására, amely segíti a kezdő csellójátékosok számára a gyakorlást.

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani témavezetőmnek, Dr. Papp Zoltánnak, aki a szakdolgozat elkészítésében nagyon sok segítséget nyújtott, és minden felmerülő kérdésemre választ adott.

Továbbá köszönet illeti páromat és családomat. Szeretetük, biztatásuk és tanácsaik nélkül nem készült volna el ez a dolgozat.

Irodalomjegyzék

- [1] Dr. Gordos Géza, Takács György: Digitális beszédfeldolgozás, Műszaki Könyvkiadó, Budapest, (1983)
- [2] Tarnóczy Tamás: Zenei akusztika, Zeneműkiadó, Budapest, (1982)
- [3] Douglas O'Shaughnessy: Speech Communication Human and Machine, Addison-Wesley Publishing Company, (1987)
- [4] Balázs Tamás Zsolt: Akusztikai jelek elemzése, Szakdolgozat, Debrecen (2010)
- [5] Szaszko Mónika: Beszédjelek intonációs és hangsúlyvizsgálata, Szakdolgozat, Debrecen (2003)
- [6] Cseh Miklós Zsolt: A beszéd fonetikai komponenseinek számítógépes vizsgálata, Szakdolgozat, Debrecen (2009)
- [7] Az audiometria fizikai alapjai
http://image.hotdog.hu/_data/members2/238/15238/doksi/aoskpylxmlb.pdf
- [8] http://opt.physx.u-szeged.hu/oktatas/Zenei_akusztika/Eloadas7.pdf
- [9] <http://szft.elte.hu/~kojnok/akuea08/Aku09ea2.ppt>
- [10] Dr. Papp Zoltán: Digitális beszédfeldolgozás jegyzet
<http://www.inf.unideb.hu/~pappzol/hallginf/digitbeszed.pdf>
- [11] Az információ és mérése
http://www.angela.sulinet.hu/hz/info_9/vazlat/informacio.html
- [12] Hangfeldolgozás
<http://itl7.elte.hu/hlabdb/hangf/hangf.pdf>
- [13] http://alpha.tmit.bme.hu/speech/docs/education/beszeddiag_03gyak.pdf
- [14] Pitch Marking Based on an Adaptable Filter and a Peak-Valley Estimation Method
<http://www.aclclp.org.tw/rocling/2001/M08.pdf>

- [15] Efficient Pitch Detection Techniques for Interactive Music
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.142.7667&rep=rep1&type=pdf>
- [16] Overview of Homophonic Pitch Detection algorithms
http://www.music.mcgill.ca/~ich/classes/mumt611_06/Presentations/savard/savard06overview.pdf
- [17] Extended Average Magnitude Difference Function Based Pitch Detection
<http://www.ccis2k.org/iajit/PDF/vol.8,no.2/12-1093.pdf>
- [18] YIN, a fundamental frequency estimator for speech and music
http://audition.ens.fr/adc/pdf/2002_JASA_YIN.pdf
- [19] MIDIToolbox Manual
<https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/miditoolbox/Manual>
- [20] Lilypond dokumentáció
<http://lilypond.org/doc/v2.12/Documentation/user/lilypond-program/>