# 1. UVOD

Pretraživanje muzičkih informacija (eng. *Music Information Retrieval – MIR*)[1)] predstavlja jednu interdisciplinarnu nauku koja datira još od 1960. godine, a svoj procvat doživljava devedesetih godina prošlog vijeka, paralelno sa razvojem računara, interneta i digitalnih audio formata. Jedan od ključnih problema koji se ovde javlja je omogućavanje efikasnog pristupa ogromnim kolekcijama muzičkih sadržaja, kao i pronalaženje algoritama za analiziranje, obradu, pretragu i arhiviranje tih sadržaja. U poređenju sa tradicionalnim pretraživanjem tekstualnih informacija, pretraživanje muzičkih informacija je mnogo kompleksniji proces jer zahtijeva razmatranje nekoliko različitih reprezentacija informacionog sadržaja, koji može biti dat u formi digitalnog audia (mp3, wav, wma i sl.) kao i u formi simboličkih podataka (MIDI, notna transkripcija, tekst pjesme itd.).

Automatsko prepoznavanje akorda već nekoliko godina predstavlja problem od interesa, u kontekstu pretraživanja muzičkih informacija. Stoga su vršeni razni pokušaji implementacije odgovarajućeg sistema, koristeći postojeće znanje iz digitalne obrade signala kao i iz oblasti statističkog modelovanja i prepoznavanja govornih sekvenci.

Akord u muzici se sastoji od nekoliko tonova različite visine koji zvuče istovremeno, ili odvojeno ako pritom ostavljaju utisak sazvučja. Najčešće su to skupovi tri ili više tonova različite visine, koji se tvore dodavanjem različitih tonskih intervala osnovnom tonu. Jedan od mnogih načina klasifikovanja akorda je razvrstavanje prema broju tonova koji se koriste u njihovoj građi, gdje ton određuje stepen u dijatonskoj skali, bez obzira na to u kojoj se oktavi pojavljuje. Akordi koji koriste tri tona zovu se *trikordi*, akordi koji koriste četiri tona *tetrakordi*, pet tonova *pentakordi* a akordi sa šest različitih tonova zovu se *heksakordi*. Tako na primjer, akord C-dur se sastoji od osnovnog tona (C), velike terce (E) i kvinte (G) dok se akord C-mol dobija dodavanjem snižene (male) terce i kvinte osnovnom tonu pa imamo niz C-Es-G.

Ređanje akorda u vremenu, odnosno akordska progresija, čini osnovu harmonije u jednom muzičkom djelu. Analiziranje cjelokupne harmonijske strukture kompozicije najčešće počinje sa obilježavanjem svakog pojedinog akorda. Iz tog razloga, automatsko obilježavanje akorda je veoma korisno za nekog ko se često susreće sa harmonijskom i muzikološkom analizom. Jednom kada se odredi harmonijski sadržaj, sekvenca akorda može biti korištena za dalju strukturalnu analizu, gdje se na višem nivou posmatranja mogu identifikovati fraze i forme muzičkog djela.

Sekvenca akorda može takođe predstavljati solidnu reprezentaciju muzičkog signala za korištenje u sve prisutnijim aplikacijama koje obrađuju probleme pretraživanja muzičkih baza podataka, segmentacije muzike, identifikacije muzičkih i harmonijskih sličnosti kao i problem *audio thumbnailing*a tj. kreiranja kratkih reprezentativnih uzoraka muzičkog sadržaja. Zbog navedenih, kao i zbog mnogih drugih razloga, problem automatskog prepoznavanja akorda je u posljednje vrijeme privukao mnoge istraživače iz oblasti pretraživanja muzičkih informacija.

[1] http://www.ismir.net

Akordska progresija zapisana sekvencom akorda predstavlja jednu kompaktnu formu u kojoj je sadržano mnoštvo karakteristika muzičkog djela. Akordi takođe nose u sebi interesantnu osobinu a to je da se audio zapis može podijeliti na vremenske intervale koji sadrže jedan jedini akord, kao što se snimljeni ljudski govor može podijeliti na vremenske intervale koji odgovaraju riječima, odnosno pojedinim fonemima. Navedena korespondencija upućuje na činjenicu da bi se, prilikom razvoja alata za prepoznavanje akorda, moglo iskoristiti već postojeće znanje iz oblasti prepoznavanja govora. **Skriveni Markovljevi modeli** (eng. *Hidden Markov Models – HMM*) [1],[2] su veoma uspješno korišteni u prepoznavanju govora pa su stoga, u posljednjih nekoliko godina razvijene i korištene slične tehnike i za prepoznavanje akorda.

Ljudski organ sluha je u mogućnosti da iz kompleksnih audio signala izvuče i obradi razne vrste podataka. Stoga je od velikog interesa nastojanje da se modeluje ovaj proces pomoću računara. U domenu obrade govora stvari su mnogo jednostavnije zbog same prirode govornog signala, dok se kod muzičkih signala problem znatno komplikuje, zbog raznih akustičkih varijacija u pogledu širine frekvencijskog opsega, instrumentacije, dinamike, trajanja itd. Takođe, potrebno je baratati sa znatno većom količinom podataka što usložnjava sam proces obrade.

U ovom radu biće izložen metod prepoznavanja sekvence akorda koji se bazira na dostignućima *Lee*a i *Slaney*a (2006) [3] kao i *Sheh*a i *Ellis*a (2003) [4]. Pristup je zasnovan na korištenju Skrivenih Markovljevih modela, koji akorde tretiraju kao skrivena stanja, i koji su obučavani pomoću sofisticiranog algoritma **maksimizacija očekivanja** (eng. *Expectation – Maximization – EM*) odnosno, konkretne primjene tog algoritma na HMM pod nazivom **Baum – Welch** algoritam [1],[2].

Prvi korak u rješavanju problema predstavlja izdvajanje vektora obilježja iz "sirovog" audio zapisa. Hromagram ili ***Pitch Class Profile – PCP,*** kao vektor obilježja, je našao svoje mjesto u gotovo svakom sistemu za automatsko prepoznavanje akorda, od kada je predložen od strane *Fujishim*e 1999. godine [5]. PCP vektori predstavljaju mjerilo energije u spektru, raspoređene u različitim frekvencijskim regionima. Pošto spektralna karakteristika muzičkog signala zavisi od izvora koji stvara zvuk, mjera spektralne energije (tačnije, spektralne gustine snage) će uključiti informacije nezavisno od izvora tj. od pojedinih muzičkih instrumenata.

PCP vektori se koriste za obučavanje skrivenog Markovljevog modela, sa jednim stanjem za svaki akord. Skriveni Markovljev model, kao stohastički konačni automat, predstavlja proširenje diskretnog Markovljevog modela, u kojem su stanja skrivena u smislu da osnovni stohastički proces nije direktno opservabilan (skriven je) ali može biti opažen preko drugog stohastičkog procesa, kojim je definisana raspodjela emitovanja spektralnih komponenti tj. opservacija. Otuda potiče i naziv skriveni Markovljevi modeli. Prelazak iz jednog u drugo stanje ispunjava Markovljevu osobinu da, ako je dato trenutno stanje, buduće stanje je nezavisno od prošlog [1].

Prepoznavanje akorda vrši se pomoću HMM-a gdje svako stanje predstavlja jedan akord. Za modelovanje raspodjele PCP vektora (opservacija) za svako stanje, uzeta je 12-dimenzionalna Gausova raspodjela, opisana vektorom srednjih vrijednosti i kovarijansnom matricom. Pretpostavlja se da su obilježja međusobno nekorelisana, tako da se kovarijansna matrica sastoji samo od dijagonalnih elemenata (varijansi).

Za specifikaciju sistema potrebno je, dakle, odrediti 12-dimenzionalni vektor srednjih vrijednosti i 12-dimenzionalni vektor varijansi, koji su pridruženi emitujućem stanju, kao i vjerovatnoće prelazaka iz pojedinih stanja. Ako je unaprijed poznato koje stanje emituje određenu opservaciju u okviru podataka za obučavanje, parametri modela se mogu direktno procijeniti.

Jednom, kada su parametri modela dobijeni obučavanjem (vjerovatnoće početnih stanja, vjerovatnoće prelazaka iz jednog stanja u drugo kao i vektor srednjih vrijednosti i kovarijansna matrica za svako stanje), za određivanje "optimalnog puta" tj. najvjerovatnije sekvence akorda nepoznatog audio zapisa, koristi se ***Viterbijev*** algoritam [1],[2].

Jedan od najvažnijih zadataka prilikom odabira i specifikacije modela jeste procjena broja stanja tj. skupa akorda koji se posmatraju, kao i procjena veličine skupa podataka (fajlova) za obuku. *Sheh* i *Ellis* su u svom radu [4] koristili relativno malen skup podataka, koji se sastojao od 20 pjesama *Beatles*a za obučavanje i dvije pjesme za prepoznavanje, u poređenju sa jako velikim brojem stanja (147 tipova akorda), što je rezultovalo prilično malim procentom uspješnosti prepoznavanja, koji je iznosio oko 22%. Veći skup podataka za obuku nisu mogli uzeti iz čisto praktičnih razloga jer su morali ručno označavati akorde za svaki fajl iz trening skupa, što iziskuje jako mnogo vremena.

*Lee* i *Slaney* su predložili metod automatskog označavanja akorda koristeći simboličke podatke [3], kao što su MIDI podaci (eng. *Musical Instruments Digital Interface – MIDI*), za generisanje naziva akorda kao i za kreiranje audio signala trening skupa. Audio informacije i oznake akorda su u ovom slučaju u savršenom sinhronizmu i mogu se koristiti za procjenu parametara modela. Sa podacima dobijenim na ovaj način, koristeći 175 fajlova *Haydn*ovih gudačkih kvarteta za obuku i definišući 36 stanja sistema, *Lee* i *Slaney* su postigli jako dobre rezultate (oko 90% tačnosti) u pogledu prepoznavanja sekvenci akorda (premda su u radu prikazali rezultate samo za jedan testni audio signal).

U ovom radu pokušalo se objediniti prednosti i jednog i drugog gore navedenog pristupa. Za razliku od *Lee*a i *Slaney*a koji su koristili audio zapise sa samo jednim instrumentom (klavir), problem je proširen na više instrumenata tako da je stvar približena realnoj situaciji iz prakse. Kao i kod *Sheh*a i *Ellis*a, korišteno je dvadesetak pjesama *Beatles*a, s tim da je broj stanja smanjen na 24 iz čisto praktičnih razloga, jer je (kao što će se vidjeti) broj stanja uslovljen količinom podataka za obučavanje. Takođe, korišten je metod automatskog generisanja labela akorda za trening skup, koji je realizovan kao funkcija u MATLAB-u.

S druge strane, primjenjen je ozbiljniji pristup tumačenju dobijenih rezultata jer je model testiran na svakom pojedinačnom audio fajlu iz odabranog skupa. Posmatran je skup podataka koga čini 21 pjesma *Beatles*a i u svakom koraku je izabrana jedna pjesma za prepoznavanje a ostalih 20 je korišteno za obučavanje modela. Na taj način je moguće uočiti kako izbor trening skupa, odnosno korelacija između trening skupa i test fajla, utiče na rezultate prepoznavanja akorda.

Na samom početku, tačnije u drugoj glavi, izložen je matematički precizan, teorijski pregled skrivenih Markovljevih modela. Definisani su osnovni problemi koji se javljaju (evaluacija, dekodiranje i obučavanje), odnosno najvažniji algoritmi (*Forward, Backward, Baum - Welch i Viterbi*) koji su našli primjenu u praktičnim aplikacijama. PCP vektori kao obilježja koja se koriste u sistemu za prepoznavanje akorda, uvedeni su i objašnjeni u trećoj glavi. Četvrta glava opisuje implementaciju sistema za automatsko prepoznavanje sekvence akorda u okviru kojeg je realizovan i sistem za automatsko označavanje akorda iz trening skupa podataka. Rezultati testiranja sistema na konkretnim primjerima kao i komentari dobijenih rezultata izloženi su u petoj glavi. Nakon toga slijedi zaključak a posljednje dvije glave donose prilog (u kome je data lista korištenih skraćenica, kompletan kôd realizovanih MATLAB skripti i funkcija, kao i opis funkcija koje su korištene za generisanje slika u radu) i spisak korištene literature. Uz rad je priložen i CD na kojem se nalazi elektronska verzija rada, m-fajlovi realizovani za potrebe rada i slike korištene u radu.

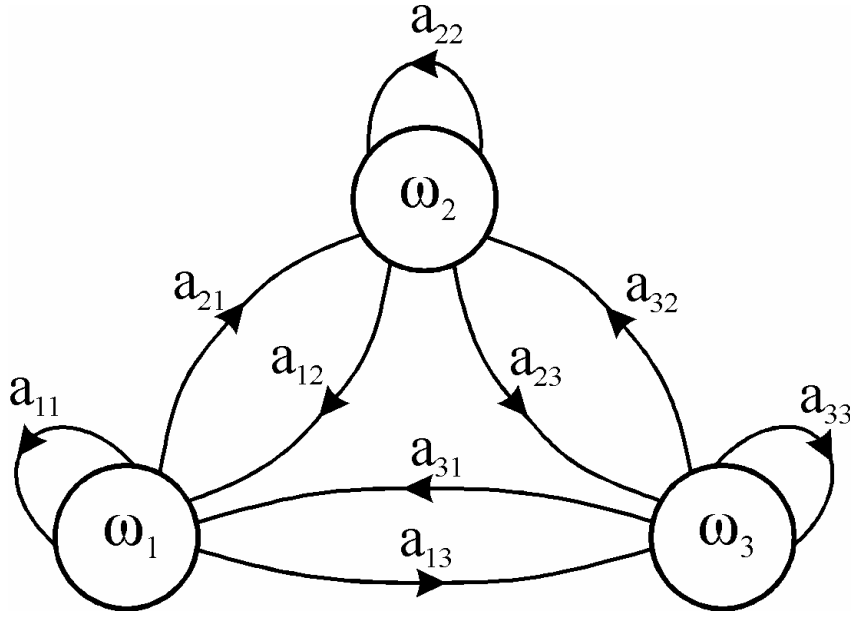
# 2. SKRIVENI MARKOVLJEVI MODELI

Iako su prvi put predstavljene i proučavane krajem šezdesetih i početkom sedamdesetih godina prošlog vijeka, statističke metode modelovanja procesa pomoću Markovljevih modela i skrivenih Markovljevih modela, u posljednjih nekoliko godina postaju izrazito popularne. Postoji nekoliko razloga koji opravdavaju ovu činjenicu. Kao prvo, modeli su jako dobro matematički potkovani i u mogućnosti su da stvore teoretsku osnovu za dalju nadogradnju. Drugo, kada se adekvatno primjene, modeli daju veoma dobre rezultate i mogu biti iskorišteni u raznim aplikacijama od interesa, naročito u oblasti prepoznavanja govora.

## 2.1 Markovljevi modeli prvog reda

Neka je data sekvenca stanja, posmatrana u sukcesivnim diskretnim trenucima vremena, gdje je stanje u svakom vremenskom trenutku označeno sa $\omega(t)$. Sekvenca dužine T je označena sa $\omega^T = \{\omega(1), \omega(2), ..., \omega(T)\}$, kao npr. $\omega^6 = \{\omega_1, \omega_4, \omega_2, \omega_2, \omega_1, \omega_4\}$. Treba napomenuti da sistem može "posjetiti" pojedino stanje u različitim koracima i pri tome određena stanja mogu ostati neposjećena.

Model u kojem se generišu sekvence je opisan *vjerovatnoćama prelaza* $P(\omega_j(t+1)|\omega_i(t)) = a_{ij}$, gdje $a_{ij}$ predstavlja vjerovatnoću (nezavisnu od vremena) da će sistem iz stanja $\omega_i$, u kojem se nalazi u trenutku $t$, preći u stanje $\omega_j$ u trenutku $t+1$. Vjerovatnoće prelazaka ne moraju nužno biti simetrične ($a_{ij} \neq a_{ji}$) i sistem se u pojedinom stanju može zadržati proizvoljno dugo ($a_{ii} \neq 0$), kao što je prikazano na slici 2.1.




***Slika 2.1 – Markovljev model prvog reda.***

Pod pretpostavkom da je definisan model $\theta$, koga čine skup vjerovatnoća $a_{ij}$ i sekvenca $\omega^T$, moguće je izračunati vjerovatnoću da će model generisati konkretnu sekvencu, tako što se jednostavno pomnože sukcesivne vjerovatnoće prelaza. Na primjer, vjerovatnoća da model generiše već spomenutu sekvencu $\omega^6 = \{\omega_1, \omega_4, \omega_2, \omega_2, \omega_1, \omega_4\}$ iznosi $P(\omega^T|\theta) = a_{14}a_{42}a_{22}a_{21}a_{14}$. Ako postoji priorna vjerovatnoća (vjerovatnoća početnog stanja) $P(\omega(1) = \omega_i)$, mogla bi se i ona uključiti u proizvod.

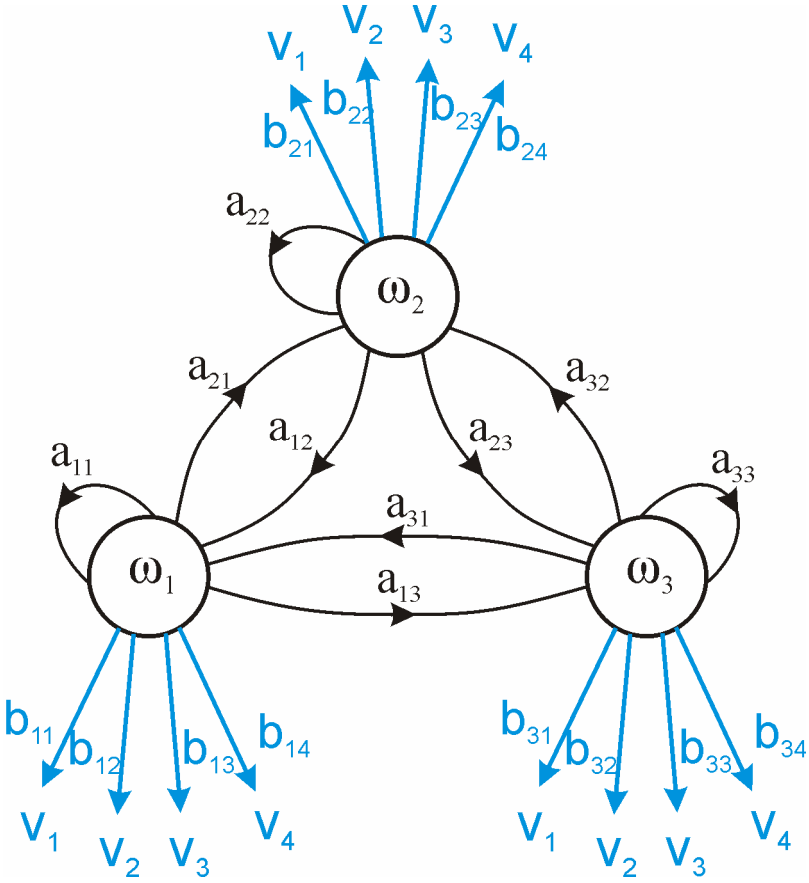
U prethodnom tekstu opisan je Markovljev model, odnosno preciznije govoreći, Markovljev model prvog reda (posmatran u diskretnim vremenskim trenucima) jer vjerovatnoća stanja u trenutku $t+1$ zavisi samo od stanja u trenutku *t.* Treba primjetiti da kod prepoznavanja govora i akorda, nije moguće direktno opaziti (mjeriti, opservirati) stanja $\omega(t)$ tj. stanja su skrivena. Umjesto toga moguće je opaziti neke druge osobine emitovanog zvuka, kao što je spektar. Zbog toga će biti potrebno proširiti Markovljev model sa tzv. *vidljivim stanjima*, koja su direktno dostupna za opažanja i mjerenja, za razliku od stanja $\omega$ koja to nisu.

## 2.2 Skriveni Markovljevi modeli prvog reda

Sada je potrebno, pored činjenice da se u svakom diskretnom vremenskom trenutku *t* sistem nalazi u stanju $\omega(t)$, pretpostaviti da on u svakom koraku emituje jedan vidljivi simbol $v(t)$, koji u opštem slučaju može biti i kontinualna funkcija (npr. spektar). Sekvenca vidljivih stanja je definisana sa $V^T = \{v(1), v(2), ..., v(T)\}$, kao npr. $V^6 = \{v_5, v_1, v_1, v_5, v_2, v_3\}$. Kada se model nalazi u proizvoljnom stanju $\omega_j(t)$, moguće je definisati vjerovatnoću emitovanja pojedinog vidljivog stanja $v_k(t)$. Ta vjerovatnoća je označena sa $P(v_k(t)|\omega_j(t)) = b_{jk}$.

Pošto su samo vidljiva stanja direktno opservabilna (stanja $\omega_i$ su nevidljiva za okolinu), model je dobio naziv *skriveni Markovljev model.*

Mreže poput one na Slici 2.2 u literaturi se često nazivaju *konačnim automatima,* a zajedno sa vjerovatnoćama prelaza čine *Markovljeve mreže.* One su striktno *kauzalne* jer vjerovatnoće zavise samo od prethodnih stanja. Markovljev model je *ergodičan* ako svako stanje ima vjerovatnoću pojavljivanja različitu od nule, pri zadatom početnom stanju. *Konačno* ili *apsorbujuće stanje* je ono koje, jednom kad je posjećeno, ne biva više napušteno.




***Slika 2.2 – Skriveni Markovljev model.***

Kao što je već rečeno, vjerovatnoće prelaza između stanja $a_{ij}$ i vjerovatnoće emisija vidljivih stanja $b_{jk}$ su date sa:

$$\begin{aligned} a_{ij} &= P(\omega_j(t+1)|\omega_i(t)) \\ b_{jk} &= P(v_k(t)|\omega_j(t)) \\ \sum_j a_{ij} &= 1, \quad \forall i \\ \sum_k b_{jk} &= 1, \quad \forall j \end{aligned} \tag{2.1}$$

pri čemu, u svakom koraku ($t \to t+1$), sistem izvrši jedan prelaz (makar to bilo isto stanje) i nakon toga emituje vidljive simbole (vidljivo stanje).

## 2.3 Osnovni zadaci skrivenih Markovljevih modela

Postoje tri osnovna problema od interesa koje treba riješiti prilikom korištenja HMM-a u realnim aplikacijama:

- **Problem evaluacije –** pod pretpostavkom da je dat HMM, zajedno sa vjerovatnoćama $a_{ij}$ i $b_{jk}$, odrediti vjerovatnoću da je model generisao konkretnu sekvencu vidljivih stanja $V^T$.

- **Problem dekodiranja –** ako je dat model, zajedno sa sekvencom opservacija $V^T$, odrediti najvjerovatniju sekvencu skrivenih stanja $\omega^T$, koja je dovela do tih opservacija.

- **Problem obučavanja –** ako je poznata samo gruba struktura modela (broj skrivenih i vidljivih stanja) ali ne i vjerovatnoće $a_{ij}$ i $b_{jk}$, odrediti ove parametre za zadati trening skup opservacija (vidljivih stanja).

### *2.3.1 Evaluacija*

Vjerovatnoća da je model generisao sekvencu vidljivih stanja $V^T$ je:

$$P(V^T) = \sum_{r=1}^{r_{\max}} P(V^T|\omega_r^T)P(\omega_r^T), \tag{2.2}$$

gdje *r* označava pojedinu sekvencu $\omega_r^T = \{\omega(1), \omega(2), ..., \omega(T)\}$ od *T* skrivenih stanja. U opštem slučaju, ako postoji *c* skrivenih stanja, dobiće se $r_{\max} = c^T$ mogućih članova u sumi (2.2) koji odgovaraju svim mogućim sekvencama dužine *T*. Pri proračunu vjerovatnoće da je model generisao konkretnu sekvencu *T* vidljivih stanja $V^T$, potrebno je uzeti u obzir svaku moguću kombinaciju skrivenih stanja, izračunati vjerovatnoću da je ta kombinacija emitovala $V^T$ i na kraju sabrati te vjerovatnoće. Vjerovatnoća pojave pojedine vidljive sekvence stanja je jednostavno proizvod odgovarajućih (skrivenih) vjerovatnoća prelaza $a_{ij}$ i (vidljivih) emisionih vjerovatnoća $b_{jk}$ u svakom koraku.

Pošto se radi o Markovljevom procesu prvog reda, drugi faktor u proizvodu jednačine (2.2), koji predstavlja vjerovatnoće prelaza skrivenih stanja, može se zapisati kao:

$$P(\omega_r^T) = \prod_{t=1}^{T} P(\omega(t)|\omega(t-1)), \tag{2.3}$$

i u suštini predstalja proizvod svih $a_{ij}$ koji odgovaraju skrivenoj sekvenci stanja. U jednačini (2.3), $\omega(T)=\omega_0$ je posljednje, apsorbujuće stanje koje emituje jedinstveno vidljivo stanje $v_0$. Zbog pretpostavke da emisione vjerovatnoće zavise samo od skrivenih stanja, prvi faktor u proizvodu jednačine (2.2) se može zapisati kao:

$$P(V^T|\omega_r^T)=\prod_{t=1}^{T}P(v(t)|\omega(t)), \tag{2.4}$$

i predstavlja proizvod svih $b_{jk}$ definisanih u odnosu na skriveno stanje i odgovarajuće vidljivo stanje.

Sada se, koristeći (2.3) i (2.4), jednačina (2.2) može zapisati kao:

$$P(V^T)=\sum_{r=1}^{r_{\max}}\prod_{t=1}^{T}P(v(t)|\omega(t))P(\omega(t)|\omega(t-1)), \tag{2.5}$$

i interpretirati na način da je vjerovatnoća opservacije konkretne sekvence $T$ vidljivih stanja $V^T$ jednaka sumi uslovnih vjerovatnoća da je sistem generisao pojedini prelaz, pomnoženih sa vjerovatnoćama da je nakon toga sistem emitovao vidljivo stanje o okviru posmatrane sekvence, pri čemu se sumiranje vrši po svim $r_{\max}$ mogućim sekvencama skrivenih stanja. Sve ove vjerovatnoće su sadržane u parametrima $a_{ij}$ i $b_{jk}$ i jednačina (2.5) se može direktno izračunati pri čemu je broj računskih operacija $O(c^T T)$. Ovo je prilično veliki broj operacija i za $c=10$ i $T=20$ on je reda $10^{21}$.

Moguće je dati i jednostavniji algoritam, koji zahtijeva manji broj računskih operacija i koji rekurzivno računa vjerovatnoću $P(V^T)$, pošto svaki član $P(v(t)|\omega(t))P(\omega(t)|\omega(t-1))$ uključuje samo veličine $v(t),\omega(t)$ i $\omega(t-1)$. Potrebno je prvo definisati veličinu

$$\alpha_j(t)=\begin{cases}0 & t=0,\ \omega_j \neq \text{ početno stanje}\\ 1 & t=0,\ \omega_j = \text{ početno stanje}\\ \sum_i \alpha_i(t-1)a_{ij}b_{jk}^{v(t)} & \text{inače}\end{cases} \tag{2.6}$$

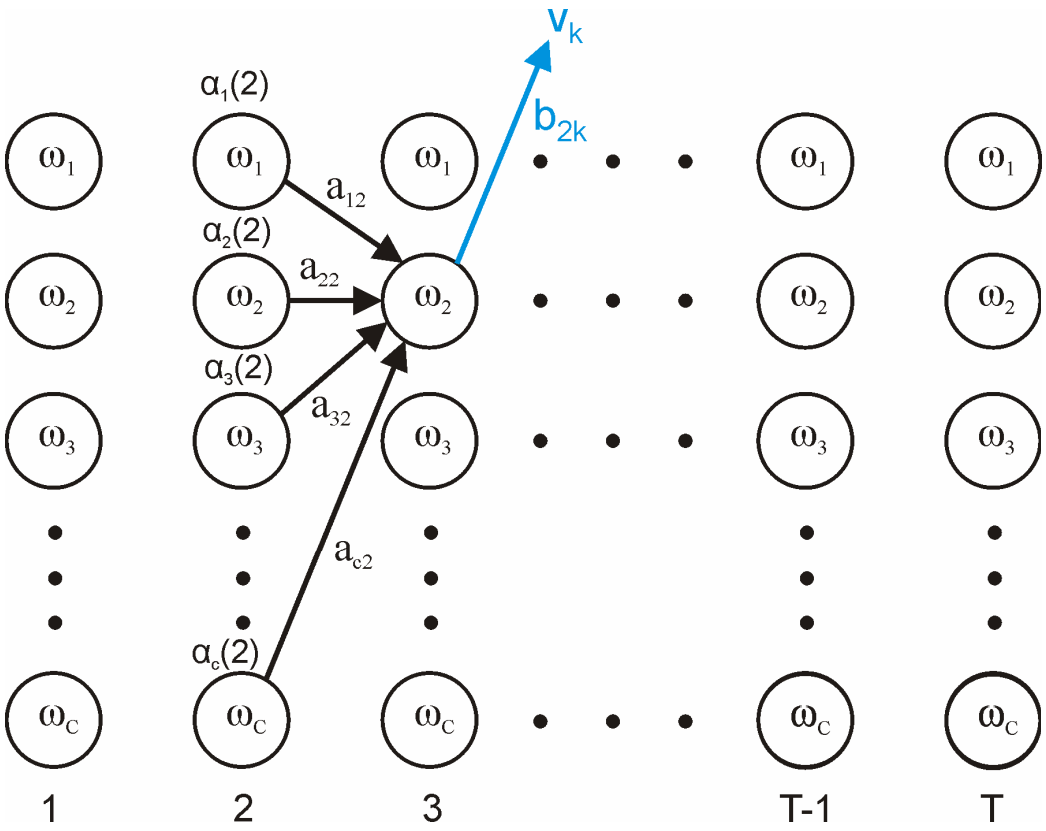
gdje je sa $b_{jk}^{v(t)}$ označena emisiona vjerovatnoća vidljivog stanja emitovanog u trenutku *t*. Veličina $\alpha_j(t)$ predstavlja vjerovatnoću da je, u trenutku *t*, model u stanju $\omega_j$ i da je generisao prvih *t* elemenata sekvence $V^T$. Ovaj proračun je implementiran u tzv. *algoritmu unaprijed* (eng. *Forward algorithm*):

**Algoritam unaprijed (HMM Forward)**

1. **initialize** $\omega(1), t \leftarrow 0, a_{ij}, b_{jk}$, vidljiva sekvenca $V^T, \alpha(0) \leftarrow 1$
2. **for** $t \leftarrow t+1$
3. $\alpha_j(t) \leftarrow \sum_{i=1}^{c} \alpha_i(t-1) a_{ij} b_{jk}^{v(t)}$
4. **until** $t = T$
5. **return** $P(V^T) \leftarrow \alpha_0(T)$
6. **end**

U liniji 5, $\alpha_0$ označava vjerovatnoću da se sekvenca završi poznatim konačnim (aprosbujućim) stanjem.

Proračunavanje vjerovatnoća algoritmom unaprijed može biti jasnije objašnjeno posmatranjem primjera modela datog na slici 2.3. Ako je potrebno izračunati vjerovatnoću da se model nalazio u stanju $\omega_2$ u trenutku $t = 3$ i da je u tom koraku generisao opservaciju $v_k$, potrebno je uočiti vjerovatnoću da je HMM u koraku $t = 2$ bio u stanju $\omega_j$ (i pri tom generisao određenu opservaciju) koja iznosi $\alpha_j(2)$ za $j = 1, 2, ..., c$. Vjerovatnoća $\alpha_2(3)$ se dobije kao suma svih proizvoda $\alpha_j(2) \cdot a_{j2}$ pomnoženih sa vjerovatnoćom $b_{2k}$ emitovanja opservacije $v_k$.




*Slika 2.3 –Ilustracija proračuna vjerovatnoća primjenom algoritma unaprijed.*

Broj računskih operacija je sada $O(c^2T)$, što je mnogo manje nego u prethodnom slučaju. Ilustracije radi, za iste vrijednosti $c=10$ i $T=20$, potrebno je izvršiti oko 2000 operacija, odnosno 17 redova veličine manje.

Veličina $\alpha_i(t)$ uvedena je kao vjerovatnoća da se model u trenutku t nalazi u stanju $\omega_i(t)$ i da je generisao prvih t elemenata sekvence vidljivih stanja. Analogno, moguće je definisati i veličinu $\beta_i(t)$ kao vjerovatnoću da se model nalazi u stanju $\omega_i(t)$ i da će u narednim koracima generisati ostale elemente sekvence opservacija, od $t+1$ do $T$, što matematički zapisano daje:

$$\beta_i(t)=\begin{cases}0 & t=T,\ \omega_i \neq \text{ konačno stanje}\\ 1 & t=T,\ \omega_i = \text{ konačno stanje}\\ \sum_j a_{ij}b_{jk}^{v(t+1)}\beta_j(t+1) & \text{ostalo}\end{cases} \tag{2.7}$$

U svrhu boljeg razumijevanja jednačine (2.7) potrebno je zamisliti situaciju u kojoj su poznate $\alpha_i(t)$ sve do koraka $T-1$, i da je potrebno izračunati vjerovatnoću da će model generisati preostali vidljivi simbol. Ta vjerovatnoća je zapravo proizvod vjerovatnoće da će sistem izvršiti prelaz u stanje $\omega_j(T)$ i vjerovatnoće da će to skriveno stanje emitovati odgovarajuće vidljive simbole.

Sada se može definisati i *algoritam unazad* (eng. *Backward algorithm)* koji u suštini predstavlja algoritam unaprijed invertovan u vremenu:

**Algoritam unazad (HMM Backward)**

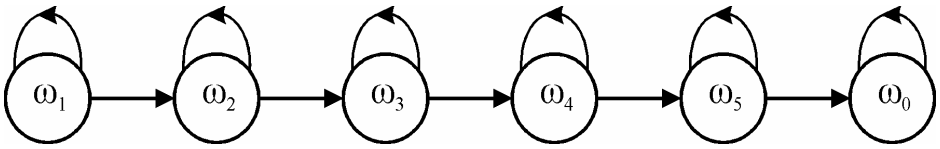
1. **initialize** $\omega(T), t \leftarrow T, a_{ij}, b_{jk},$ vidljiva sekvenca $V^T$
2. **for** $t \leftarrow t-1$
3. $\beta_i(t) \leftarrow \sum_{j=1}^{c} \beta_j(t+1)a_{ij}b_{jk}^{v(t+1)}$
4. **until** $t=1$
5. **return** $P(V^T) \leftarrow \beta_i(0),$ za poznato početno stanje
6. **end**

U liniji 5 $\beta_i$ označava vjerovatnoću da sekvenca počne sa poznatim početnim stanjem.

Ako označimo model (koeficijente $a_{ij}$ i $b_{jk}$) sa $\theta$ dobićemo *Bayes*ovu formulu koja daje vjerovatnoću modela za zadatu sekvencu opservacija:

$$P(\theta|V^T)=\frac{P(V^T|\theta)P(\theta)}{P(V^T)}. \tag{2.8}$$

U praksi su gotovo svi modeli tipa *s lijeva na desno* (npr. kod prepoznavanja govora) kao na slici 2.4.



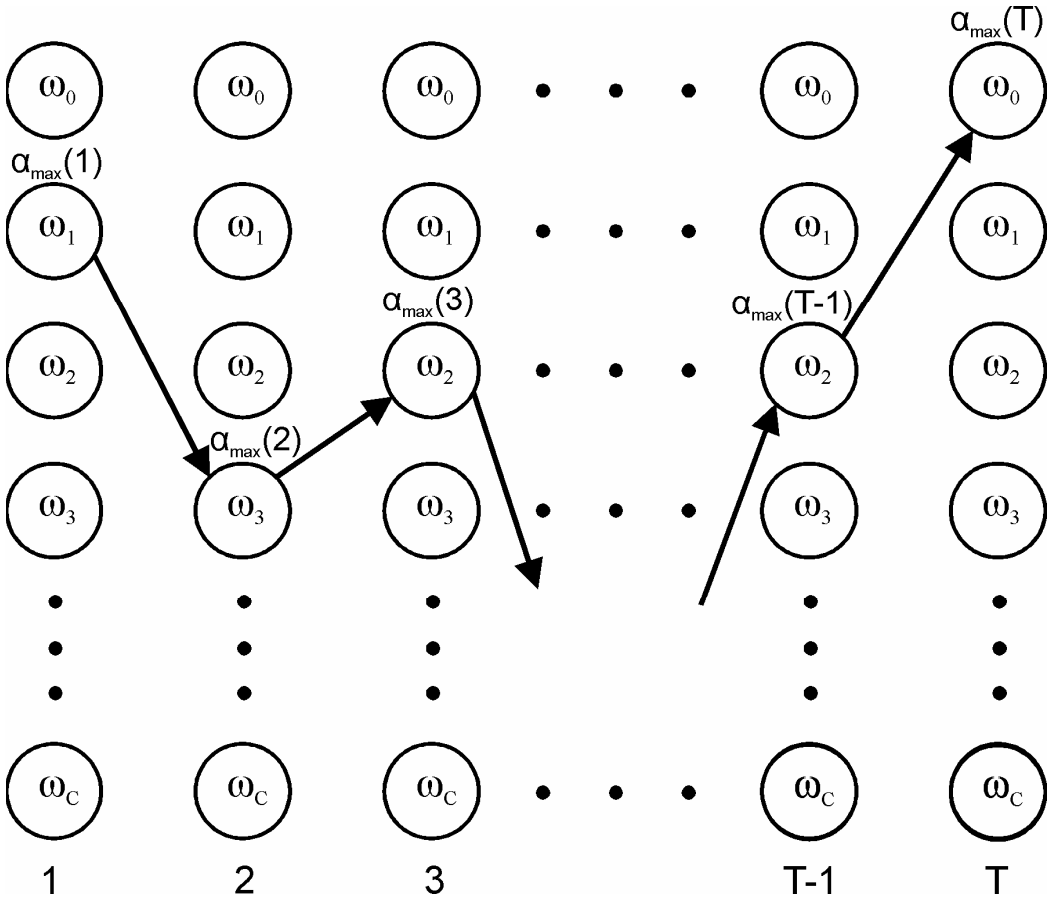

***Slika 2.4 – HMM s lijeva na desno, često korišten u prepoznavanju govora.***

### *2.3.2 Dekodiranje (Viterbijev algoritam)*

Dekodiranje predstavlja pronalaženje najvjerovatnije sekvence skrivenih stanja, pod uslovom da je data sekvenca vidljivih stanja (opservacija). Teoretski, moglo bi se razmišljati na način da se pronađe svaka moguća kombinacija stanja tj. svaki mogući put kroz model i zatim, da se za svaku kombinaciju stanja, izračunaju vjerovatnoće emitovanja date sekvence opservacija. U ovom slučaju, broj računskih operacija koje treba izvršiti je opet $O(c^T T)$, pa se primjenjuje jednostavniji algoritam poznat pod nazivom *Viterbijev* algoritam [1],[2]:

***Viterbijev* algoritam**

1. **initialize** path $\leftarrow \{\}, t \leftarrow 0$
2. **for** $t \leftarrow t+1$
3. $k \leftarrow 0$
4. **for** $k \leftarrow k+1$
5. $\alpha_k(t) \leftarrow b_{jk}^{v(t)} \sum_{i=1}^{c} \alpha_i(t-1) a_{ij}$
6. **until** $k = c$
7. $j' \leftarrow \arg\max_j \alpha_j(t)$
8. dopuniti path sa $\omega_{j'}$
9. **until** $t = T$
10. **return** path
11. **end**




***Slika 2.5 – Viterbijev algoritam.***

U ovom slučaju razmišlja se na sličan način kao kod algoritma unaprijed, samo što se u svakom vremenskom koraku, umjesto sume računa maksimum proizvoda $\alpha_j(2) \cdot a_{j2}$ za svako *j*. Na taj način se u datom koraku određuje najvjerovatnije stanje tj. stanje koje ima najveću vjerovatnoću da bude posjećeno. Sekvenca tih stanja predstavlja najvjerovatniji put kroz model.

Broj računskih operacija koje treba izvršiti je $O(c^2T)$. Neki algoritmi koriste kao veličinu logaritam vjerovatnoće ($\log \alpha_i(t)$) i računaju ukupnu vjerovatnoću kao sumu tih logaritama.

### 2.3.3 Obučavanje (Baum – Welch algoritam)

Osnovni zadatak obučavanja HMM-a je određivanje parametara modela tj. određivanje vjerovatnoća prelaza $a_{ij}$ i emisionih vjerovatnoća $b_{jk}$, iz ansambla uzoraka za obučavanje. Algoritam *unaprijed – unazad* ili *Baum – Welch* algoritam predstavlja specijalan slučaj algoritma očekivanje – maksimizacija [1],[2]. Suština je u iterativnom ažuriranju parametara modela u cilju aproksimiranja, odnosno približavanja opserviranoj trening sekvenci.

Veličine $\alpha_i(t)$ i $\beta_i(t)$ su samo estimati njihovih stvarnih vrijednosti pošto u jednačinama (2.6) i (2.7) nisu poznate konkretne vrijednosti vjerovatnoća prelaza i emisionih vjerovatnoća. Moguće je zato izračunati njihove poboljšane vrijednosti. Prvo je potrebno definisati novu veličinu $\gamma_{ij}(t)$ kao vjerovatnoću prelaza između $\omega_i(t-1)$ i $\omega_j(t)$ (vjerovatnoću da je sistem u trenutku $t-1$ bio u stanju $\omega_i$, a u trenutku $t$ u stanju $\omega_j$), pod uslovom da je model generisao čitavu trening sekvencu $V^T$, što daje:

$$\gamma_{ij}(t) = \frac{\alpha_i(t-1)a_{ij}b_{jk}\beta_j(t)}{P(V^T|\theta)} \tag{2.9}$$

gdje je $P(V^T|\theta)$ vjerovatnoća da je model generisao sekvencu $V^T$, sa bilo kojim prelazom u koraku $t-1 \to t$.

Sada je moguće izračunati poboljšani estimat veličine $a_{ij}$. Očekivani broj prelaza sa stanja $\omega_i$ na stanje $\omega_j$, posmatrano na cjelokupnom intervalu vremena $T$, jednak je $\sum_{t=1}^{T}\gamma_{ij}(t)$, dok je ukupan (očekivani) broj prelaza na $\omega_j$ sa bilo kog drugog stanja dat sa $\sum_{t=1}^{T}\sum_{i}\gamma_{ij}(t)$. Estimat od $a_{ij}$, označen sa $\hat{a}_{ij}$ se računa kao odnos između očekivanog broja prelaza sa $\omega_i$ na $\omega_j$ i ukupnog očekivanog broja posjećivanja $\omega_j$ tj. prelaza na to stanje:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T}\gamma_{ij}(t)}{\sum_{t=1}^{T}\sum_{i}\gamma_{ij}(t)}. \tag{2.10}$$

Na sličan način se dobije i poboljšana vrijednost veličine $b_{jk}$, označena sa $\hat{b}_{jk}$, računanjem odnosa između očekivanog broja posjećivanja stanja $\omega_j$, iz kojeg se emituje odgovarajuća opservacija $v_k$ i očekivanog ukupnog broja posjećivanja stanja $\omega_j$, sa emitovanjem bilo koje opservacije:

$$\hat{b}_{jk} = \frac{\sum_{t=1}^{T}\sum_{i}\gamma_{ij}(t)\big|_{v_k}}{\sum_{t=1}^{T}\sum_{i}\gamma_{ij}(t)}. \tag{2.11}$$

Ukratko, počinje se sa grubo izabranim, proizvoljnim estimatima veličina $a_{ij}$ i $b_{jk}$. Zatim se računaju poboljšane vrijednosti pomoću jednačina (2.10) i (2.11) i postupak se ponavlja sve dok se ne dostigne određeni kriterijum konvergencije, npr. dovoljno dobra tačnost u smislu male promjene vrijednosti estimiranih veličina, u dvije uzastopne iteracije. Ovaj postupak je u literaturi označen kao *Baum – Welch* algoritam ili algoritam *unaprijed – unazad* i predstavlja primjer uopštenog EM algoritma [2].

***Algoritam Baum – Welch***

1. **initialize** $a_{ij}, b_{jk}$, trening sekvenca $V^T$, kriterijum konvergencije $\varepsilon$
2. **do** $z \leftarrow z+1$
3. računanje $\hat{a}_{ij}(z)$ iz $a_{ij}(z-1)$ i $b_{jk}(z-1)$ pomoću jed. (2.10)
4. računanje $\hat{b}_{jk}(z)$ iz $a_{ij}(z-1)$ i $b_{jk}(z-1)$ pomoću jed. (2.11)
5. $a_{ij}(z) \leftarrow \hat{a}_{ij}(z)$
6. $b_{jk}(z) \leftarrow \hat{b}_{jk}(z)$
7. **until** $\max_{i,j,k}[a_{ij}(z)-a_{ij}(z-1), b_{jk}(z)-b_{jk}(z-1)] < \varepsilon$
8. **return** $a_{ij} \leftarrow a_{ij}(z), b_{jk} \leftarrow b_{jk}(z)$
9. **end**

Kriterijum konvergencije u liniji 7 zaustavlja obučavanje kada promjena estimacije vjerovatnoće prelaza bude manja od unaprijed zadate vrijednosti $\varepsilon$. Drugi popularni kriterijum zaustavljanja algoritma se zasniva na računanju vjerovatnoće da obučen model generiše cjelokupan trening skup.

Ako je poznat parametarski oblik gustine vjerovatnoće, moguće je redukovati problem obučavanja i sa pronalaženja same funkcije raspodjele preći na pronalaženje parametara raspodjele. Metod maksimalne vjerovatnoće pronalazi vrijednosti parametara koje najbolje odgovaraju trening skupu, tj. maksimizuje vjerovatnoću pojavljivanja opserviranih uzoraka. U praksi se, kao što je rečeno, zbog praktičnosti često koristi logaritam vjerovatnoće.

## 2.4 HMM – pregled

Skriveni Markovljevi modeli se sastoje od čvorova koji predstavljaju skrivena stanja, međusobno povezana sa uslovnim vjerovatnoćama prelaza sa jednog stanja na drugo. Svakom stanju je pridružen skup vjerovatnoća (jedan ili više stohastičkih procesa) emitovanja konkretnog vidljivog stanja. HMM se može koristiti u slučaju modelovanja sekvenci čiji su elementi korelisani jedni s drugima, kao što je slučaj sa fonemima u govoru ili akordima u muzičkom djelu. Sve vjerovatnoće prelaza mogu biti naučene, odnosno iterativno estimirane iz uzoraka opservacija pomoću algiritma *unaprijed – unazad* tj. *Baum – Welch* algoritma koji predstavlja specijalan slučaj algoritma *očekivanje – maksimizacija* (*EM* algoritma). Klasifikacija počinje sa pronalaženjem odgovarajućeg modela iz skupa modela, koji će sa najvećom vjerovatnoćom generisati datu sekvencu opservacija i završava sa implementacijom *Viterbijevog* algoritma koji za datu sekvencu opservacija pronalazi najvjerovatniju sekvencu skrivenih stanja.

# 3. HROMAGRAM – PCP VEKTORI

Hromagram (hroma spektar) ili Pitch Class Profile – PCP [5] predstavlja efiksanu reprezentaciju audio signala u kojoj je *Fourier*ova transformacija signala, mapirana u dvanaest polustepenih klasa tonova. Razlog za ovakav izbor leži u činjenici da PCP reprezentacija čuva harmonijske informacije sadržane u muzičkom djelu.

Percepcija visine muzičkog tona može se opisati dvjema intuitivnim veličinama od kojih jedna govori kojoj oktavi ton pripada, a druga pod nazivom hroma [6] (eng. *chroma*), upućuje na položaj tona u okviru jedne oktave i njegov odnos sa drugim tonovima iste oktave. Hromagram ili PCP je 12-dimenzionalna vektorska reprezentacija hrome koja daje relativne intenzitete svakog od dvanaest polustepena (polutonova) hromatske skale. Hromagram se zapravo dobija preslikavanjem diskretnih vrijednosti dobijenih kratkotrajnom *Fourier*ovom transformacijom audio signala u 12 polustepenih klasa tonova, odnosno prelaskom iz domena diskretne frekvencije u domen diskretne veličine hroma. Kako se akordi tvore od skupa tonova i njihova oznaka je jedino određena odnosom tih tonova, tj. njihovim hromama, nezavisno od oktave kojoj pripadaju, PCP vektori predstavljaju podesno obilježje za reprezentaciju muzičkog akorda ili tonaliteta.

Za razliku od sistema zasnovanih na identifikaciji pojedinačnih tonova, prepoznavanje akorda na bazi skrivenih Markovljevih modela, obučavanih sa PCP vektorima, daje mnogo bolje rezultate jer postoji značajna korelacija između susjednih akorda.

Polazi se od kratkotrajne *Fourier*ove transformacije [7] digitalizovanog audio signala, date sa:

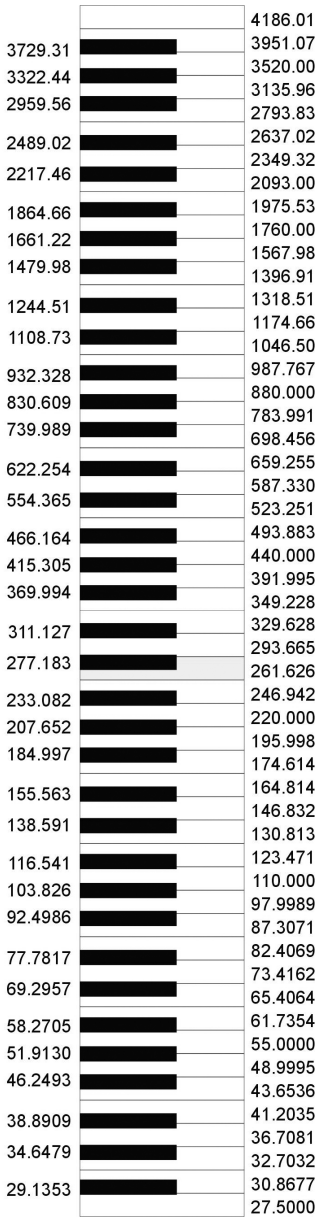
$$X_{STFT}[k,n] = \sum_{m=0}^{N-1} x[n-m]w[m]e^{-j2\pi km/N}, \tag{3.1}$$

gdje $k$ označava frekvenciju, $n$ vrijeme a $w[m]$ je prozor dužine $N$. Nakon toga računa se *hromagram* signala tj. potrebno je spektar signala preslikati u PCP (*Pitch Class* Profile) obilježja.

PCP obilježja su 12-dimenzionalni vektori, kod kojih svaki element odgovara intenzitetu polutona, odnosno hrome. Hromagram preslikava tonove iste vrijednosti hrome, nezavisno od oktave, u istu PCP klasu (PCP bin). Preslikavanje frekvencije u PCP se ostvaruje korištenjem logaritamske karakteristike *jednako temperovane muzičke skale* [8].

Kod jednako temperovane skale rastojanje od oktave podijeljeno je na 12 jednakih intervala (polustepena), posmatrajući frekvenciju u logaritamskoj razmjeri. Na taj način su dva tona udaljena jedan od drugog za polustepen ako im frekvencije stoje u odnosu:

$$f_2 = \sqrt[12]{2} f_1 = 2^{\frac{1}{12}} f_1. \tag{3.2}$$




***Slika 3.1 – Frekvencije i tonovi klavira koji odgovaraju jednako temperovanoj skali.***

Analogno, dva tona su udaljena jedan od drugog za $p$ polustepenih tonova ako su im frekvencije u odnosu:

$$f_i = \sqrt[12]{2^p}\, f_j = 2^{\frac{p}{12}} f_j. \tag{3.3}$$

Veličina $p$, koja predstavlja hromu, dobija se logaritmiranjem jednačine (3.3):

$$p = 12 \log_2 \frac{f_i}{f_j}. \tag{3.4}$$

Da bi se preslikavanje proširilo na sve frekvencije iz skupa $[0,11025]\,\mathrm{Hz}$ (radi lakše obrade, za frekvenciju odmjeravanja audio signala uzima se vrijednost od 11025 Hz) audio signal, potrebno je izvršiti zaokruživanje desne strane izraza (3.4) (oznaka $[\cdot]$), pa se uz $f_j = f_{ref}$ dobije:

$$p = \left[ 12 \log_2 \frac{f}{f_{ref}} \right] \tag{3.5}$$

Zadržavanjem ostatka dijeljenja po modulu 12, eliminiše se uticaj oktave kojoj frekvencija pripada:

$$p = \left( \left[ 12 \log_2 \frac{f}{f_{ref}} \right] \right) \bmod 12 \tag{3.6}$$

Diskretizacijom u frekvencijskom domenu [9], (kod diskretne *Fourier*ove transformacije), frekvencija signala na osnovnom periodu $[0, 2\pi)$ posmatra se u $N$ tačaka (diskretnih vrijednosti):

$$\omega_k = \frac{2\pi}{N} k, \quad k = 0,1,2,...,N-1. \tag{3.7}$$

Ako se u izraz (3.7) uključi i frekvencija odmjeravanja $F_s$, odnosno njena recipročna vrijednost $T_s = 1/F_s$, veličina $\omega_k$ će dimenziono odgovarati ugaonoj frekvenciji:

$$\omega_k = \frac{2\pi}{NT_s} k, \quad k = 0,1,2,...,N-1. \tag{3.8}$$

Iz jednakosti $\omega = 2\pi f$, dobije se izraz koji daje vrijednost diskretne frekvencije $f_k$:

$$f_k = \frac{k}{NT_s} = \frac{k}{N} F_s, \quad k = 0,1,2,...,N-1. \tag{3.9}$$

Uvrštavanjem izraza (3.9) u jednačinu (3.6), dobija se konačan izraz za $p$:

$$p(k) = \left( \left[ 12 \log_2 \frac{k}{N} \frac{F_s}{f_{ref}} \right] \right) \bmod 12, \quad k = 0,1,2,...,N-1. \tag{3.10}$$

Veličina $p$ (hroma) je broj koji uzima vrijednost iz skupa $\{0,1,2,...,11\}$ i predstavlja rezultat preslikavanja diskretne frekvencije $f_k$ iz *Fourier*ovog u PCP domen.

Na kraju je potrebno odrediti elemente 12-dimenzionalne vektorske veličine PCP, koja predstavlja obilježje spektra, veoma pogodno za definisanje opservacija tj. vidljivih stanja koje emituje HMM. Koristeći jednačine (3.1) i (3.10), za svaki diskretni vremenski trenutak $n$, vrijednost elementa na poziciji $p$ PCP vektora se dobije sumiranjem kvadrata amplituda svih odmjeraka čije se frekvencije preslikavaju u odgovarajuću hromu tj. u broj $p$, što matematički zapisano daje:

$$PCP[p,n] = \sum_{k:\, p(k)=p} \left|\mathrm{X}[k,n]\right|^2. \tag{3.11}$$

# 4. AUTOMATSKO PREPOZNAVANJE AKORDA

U ovoj glavi će biti predstavljen i opisan sistem za automatsko prepoznavanje akorda iz audio signala, zasnovan na dostignućima i metodama iz oblasti mašinskog prepoznavanja ljudskog govora [1],[10]. Realizacija je podijeljena u nekoliko koraka:

- Ekstrakcija labela iz simboličkih podataka
- Ekstrakcija PCP obilježja iz audio fajlova
- Inicijalizacija HMM-a
- HMM obuka i prepoznavanje

Akordi se tretiraju kao skrivena stanja u okviru skrivenog Markovljevog modela, dok se raspodjela vidljivih stanja (opservacija) modeluje sa multivarijabilnom Gausovom raspodjelom u 12 dimenzija, definisanom vektorom srednjih vrijednosti $\mu_i$ i kovarijansnom matricom $\Sigma_i$, gdje indeks $i$ označava $i$-to stanje. Pretpostavlja se da elementi nisu međusobno korelisani pa se koristi dijagonalna kovarijansna matrica. Prelazi između stanja ispunjavaju Markovljevu osobinu prvog reda koja kaže da je za zadato trenutno stanje, buduće stanje nezavisno od prošlog. Model će biti ergodičan jer se dozvoljava svaki mogući prelaz sa jednog na drugi akord.

Kada se model obuči sa trening podacima odnosno kada se "nauče" vjerovatnoće početnog stanja, vjerovatnoće prelaza, vektor srednjih vrijednosti i kovarijansna matrica, primjenjuje se *Viterbijev* algoritam za pronalazak optimalnog puta kroz model tj. određuje se najvjerovatnija sekvenca akorda, za datu opservaciju.

U ovom radu, implementacija sistema izvršena je u MATLAB okruženju. Iz čisto praktičnih razloga (zbog veličine trening skupa podataka), posmatran je HMM sa 24 skrivena stanja, odnosno identifikovana su 24 tipa akorda (durski i molski), po dva za svaki od 12 polustepena hromatske skale.

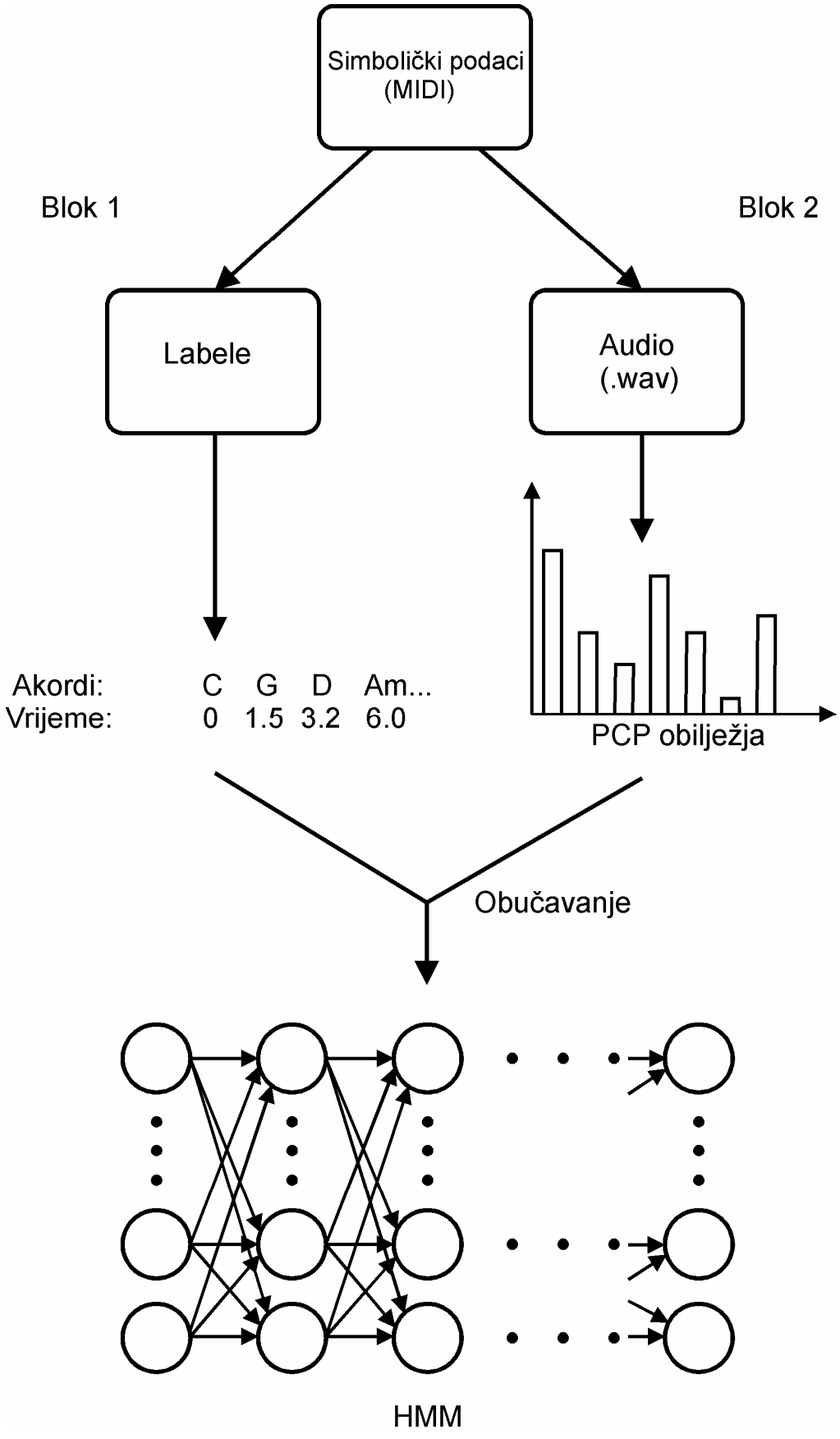
Za obučavanje i testiranje sistema korištene su pjesme *Beatles*a, tačnije njih 21. Nazivi tih pjesama su sačuvani u sklopu tekstualnog fajla ***Pjesme.txt*** i to u sljedećem redoslijedu:

***Pjesme.txt***

*A Hard Days Night.*
*A taste of honey.*
*Across the Universe.*
*Act Naturally.*
*All Ive Got To Do.*
*All my loving.*
*All You Need Is Love.*
*And I Love Her.*
*And Your Bird Can Sing.*
*Baby Its You.*
*Dont Let Me Down.*
*Help.*
*Hey Jude.*
*ObLaDi ObLaDa.*
*Please Please Me.*
*Roll Over Beethoven.*
*Tell Me Why.*
*Ticket to Ride.*
*While My Guitar Gently Weeps.*
*Yellow Submarine.*
*Youve Really Got A Hold On Me.*

S ciljem da se izvrši obučavanje modela, potrebno je bilo prvo označiti akorde u okviru trening skupa podataka. Ručno unošenje oznaka (labela) je prilično obiman posao jer je za svaki frejm audio signala potrebno unijeti po jednu labelu za akord, što znači da je za pjesmu trajanja tri minuta potrebno unijeti približno 1000 labela. Da bi se automatizovao ovaj proces korišteni su simbolički podaci (MIDI), iz kojih je moguće dobiti informaciju o labelama. Takođe, MIDI fajlovi su iskorišteni i za dobijanje audio fajlova. Generisani na ovakav način, audio fajlovi su u savršenom sinhronizmu sa oznakama akorda.

Kao što je prikazano na Slici 4.1, sistem za generisanje označenih trening podataka sadrži dva osnovna bloka, paralelno povezana. Blok 1 vrši harmonijsku analizu simboličkih podataka dok blok 2 vrši ekstrakciju obilježja (PCP vektora) iz audio signala. Spajanjem rezultujućih informacija iz oba bloka dobijaju se označena obilježja koja se zatim vode na ulaz HMM-a i koriste za obučavanje.



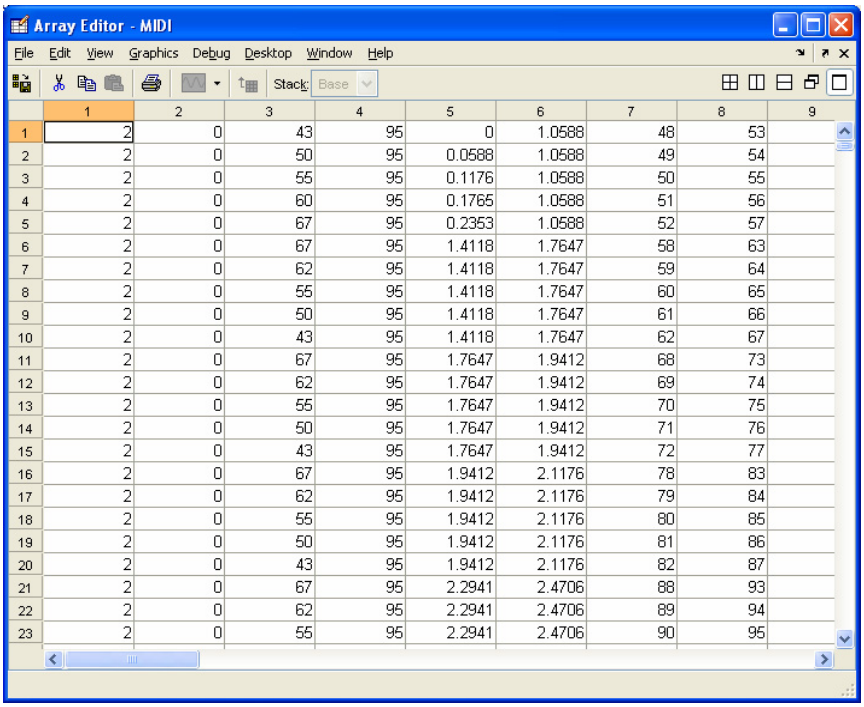

*Slika 4.1 – Sistem za automatsko prepoznavanje akorda.*

MIDI fajlovi pjesama *Beatles*a su dobijeni ekstrakcijom iz *Guitar Pro*[1)] fajlova preuzetih sa sajta *GuitarPro File Sharing*[2)]. Guitar Pro je program za upisivanje nota u sklopu kojeg je dat grafički prikaz kako klasične muzičke notacije tako i tablaturne notacije, i koji paralelno sa unošenjem pojedinih nota, automatski generiše i odgovarajući MIDI fajl.

## 4.1 Ekstrakcija labela iz simboličkih podataka

Funkcija **readmidi.m**, iz *MIDI and MATLAB Toolbox*a [11], učitava MIDI fajl u MATLAB strukturu a funkcija **midiinfo.m** daje detaljne informacije o strukturi MIDI fajla u sklopu matrice dimenzija $N \times 8$, gdje je $N$ broj nota u fajlu (slika 4.2). Kolone redom označavaju:

1. broj MIDI trake,
2. broj MIDI kanala,
3. broj note (MIDI kodovanje visine tona),
4. jačinu tona (*velocity*),
5. vrijeme početka note u sekundama,
6. vrijeme kraja note u sekundama,
7. poruka *note_on*,
8. poruka *note_off*.




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 0 | 43 | 95 | 0 | 1.0588 | 48 | 53 | |
| 2 | 2 | 0 | 50 | 95 | 0.0588 | 1.0588 | 49 | 54 | |
| 3 | 2 | 0 | 55 | 95 | 0.1176 | 1.0588 | 50 | 55 | |
| 4 | 2 | 0 | 60 | 95 | 0.1765 | 1.0588 | 51 | 56 | |
| 5 | 2 | 0 | 67 | 95 | 0.2353 | 1.0588 | 52 | 57 | |
| 6 | 2 | 0 | 67 | 95 | 1.4118 | 1.7647 | 58 | 63 | |
| 7 | 2 | 0 | 62 | 95 | 1.4118 | 1.7647 | 59 | 64 | |
| 8 | 2 | 0 | 55 | 95 | 1.4118 | 1.7647 | 60 | 65 | |
| 9 | 2 | 0 | 50 | 95 | 1.4118 | 1.7647 | 61 | 66 | |
| 10 | 2 | 0 | 43 | 95 | 1.4118 | 1.7647 | 62 | 67 | |
| 11 | 2 | 0 | 67 | 95 | 1.7647 | 1.9412 | 68 | 73 | |
| 12 | 2 | 0 | 62 | 95 | 1.7647 | 1.9412 | 69 | 74 | |
| 13 | 2 | 0 | 55 | 95 | 1.7647 | 1.9412 | 70 | 75 | |
| 14 | 2 | 0 | 50 | 95 | 1.7647 | 1.9412 | 71 | 76 | |
| 15 | 2 | 0 | 43 | 95 | 1.7647 | 1.9412 | 72 | 77 | |
| 16 | 2 | 0 | 67 | 95 | 1.9412 | 2.1176 | 78 | 83 | |
| 17 | 2 | 0 | 62 | 95 | 1.9412 | 2.1176 | 79 | 84 | |
| 18 | 2 | 0 | 55 | 95 | 1.9412 | 2.1176 | 80 | 85 | |
| 19 | 2 | 0 | 50 | 95 | 1.9412 | 2.1176 | 81 | 86 | |
| 20 | 2 | 0 | 43 | 95 | 1.9412 | 2.1176 | 82 | 87 | |
| 21 | 2 | 0 | 67 | 95 | 2.2941 | 2.4706 | 88 | 93 | |
| 22 | 2 | 0 | 62 | 95 | 2.2941 | 2.4706 | 89 | 94 | |
| 23 | 2 | 0 | 55 | 95 | 2.2941 | 2.4706 | 90 | 95 | |

***Slika 4.2 – Podaci dobijeni funkcijom midiinfo.***

[1] http://www.guitar-pro.com
[2] http://www.gprotab.net

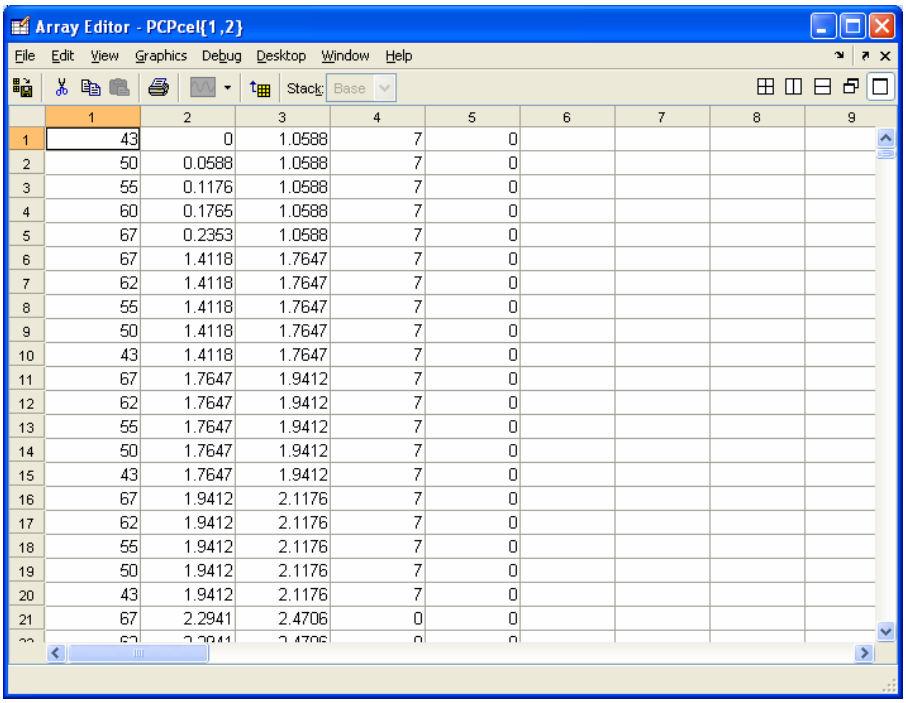
| Note No. | Name | Note No. | Name | Note No. | Name | Note No. | Name |
|---|---|---|---|---|---|---|---|
| 0 | C -2 | 32 | G#0 | 64 | E 3 | 96 | C 6 |
| 1 | C#-2 | 33 | A 0 | 65 | F 3 | 97 | C#6 |
| 2 | D -2 | 34 | A#0 | 66 | F#3 | 98 | D 6 |
| 3 | D#-2 | 35 | B 0 | 67 | G 3 | 99 | D#6 |
| 4 | E -2 | 36 | C 1 | 68 | G#3 | 100 | E 6 |
| 5 | F -2 | 37 | C#1 | 69 | A 3 | 101 | F 6 |
| 6 | F#-2 | 38 | D 1 | 70 | A#3 | 102 | F#6 |
| 7 | G -2 | 39 | D#1 | 71 | B 3 | 103 | G 6 |
| 8 | G#-2 | 40 | E 1 | 72 | C 4 | 104 | G#6 |
| 9 | A -2 | 41 | F 1 | 73 | C#4 | 105 | A 6 |
| 10 | A#-2 | 42 | F#1 | 74 | D 4 | 106 | A#6 |
| 11 | B -2 | 43 | G 1 | 75 | D#4 | 107 | B 6 |
| 12 | C -1 | 44 | G#1 | 76 | E 4 | 108 | C 7 |
| 13 | C#-1 | 45 | A 1 | 77 | F 4 | 109 | C#7 |
| 14 | D -1 | 46 | A#1 | 78 | F#4 | 110 | D 7 |
| 15 | D#-1 | 47 | B 1 | 79 | G 4 | 111 | D#7 |
| 16 | E -1 | 48 | C 2 | 80 | G#4 | 112 | E 7 |
| 17 | F -1 | 49 | C#2 | 81 | A 4 | 113 | F 7 |
| 18 | F#-1 | 50 | D 2 | 82 | A#4 | 114 | F#7 |
| 19 | G -1 | 51 | D#2 | 83 | B 4 | 115 | G 7 |
| 20 | G#-1 | 52 | E 2 | 84 | C 5 | 116 | G#7 |
| 21 | A -1 | 53 | F 2 | 85 | C#5 | 117 | A 7 |
| 22 | A#-1 | 54 | F#2 | 86 | D 5 | 118 | A#7 |
| 23 | B -1 | 55 | G 2 | 87 | D#5 | 119 | B 7 |
| 24 | C 0 | 56 | G#2 | 88 | E 5 | 120 | C 8 |
| 25 | C#0 | 57 | A 2 | 89 | F 5 | 121 | C#8 |
| 26 | D 0 | 58 | A#2 | 90 | F#5 | 122 | D 8 |
| 27 | D#0 | 59 | B 2 | 91 | G 5 | 123 | D#8 |
| 28 | E 0 | 60 | C 3 | 92 | G#5 | 124 | E 8 |
| 29 | F 0 | 61 | C#3 | 93 | A 5 | 125 | F 8 |
| 30 | F#0 | 62 | D 3 | 94 | A#5 | 126 | F#8 |
| 31 | G 0 | 63 | D#3 | 95 | B 5 | 127 | G 8 |

***Slika 4.3 – MIDI notacija visine tona.***

Harmonijska analiza simboličkih podataka u ovom radu je ostvarena funkcijom **MIDIakordi.m** (prilog), i predstavlja realizaciju bloka 1 sa slike 4.1. Prvo se iz *Guitar Pro* fajla ekstraktuje samo MIDI traka gitare. Pošto najniži ton akorda sviranog na gitari u većini slučajeva predstavlja osnovni ton akorda (MIDI fajlovi su za potrebe ovog rada unaprijed obrađeni tako da je ispunjen ovaj uslov), u prvom koraku se identifikuje osnovni ton akorda (u MIDI notaciji [12], Slika 4.3) a u drugom koraku se ispituje postojanje intervala snižene terce, računatog u odnosu na osnovni ton, što ukazuje na molski karakter akorda. Rastojanje osnovni ton – mala terca se dobije za one dvije MIDI note za koje vrijedi:

$$(nota1 - nota2) \bmod 12 = 3. \tag{4.1}$$

Na ovaj način je dobijena matrica koja sadrži početak, kraj i labelu svakog akorda u MIDI fajlu (Slika 4.4).




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 43 | 0 | 1.0588 | 7 | 0 | | | | |
| 2 | 50 | 0.0588 | 1.0588 | 7 | 0 | | | | |
| 3 | 55 | 0.1176 | 1.0588 | 7 | 0 | | | | |
| 4 | 60 | 0.1765 | 1.0588 | 7 | 0 | | | | |
| 5 | 67 | 0.2353 | 1.0588 | 7 | 0 | | | | |
| 6 | 67 | 1.4118 | 1.7647 | 7 | 0 | | | | |
| 7 | 62 | 1.4118 | 1.7647 | 7 | 0 | | | | |
| 8 | 55 | 1.4118 | 1.7647 | 7 | 0 | | | | |
| 9 | 50 | 1.4118 | 1.7647 | 7 | 0 | | | | |
| 10 | 43 | 1.4118 | 1.7647 | 7 | 0 | | | | |
| 11 | 67 | 1.7647 | 1.9412 | 7 | 0 | | | | |
| 12 | 62 | 1.7647 | 1.9412 | 7 | 0 | | | | |
| 13 | 55 | 1.7647 | 1.9412 | 7 | 0 | | | | |
| 14 | 50 | 1.7647 | 1.9412 | 7 | 0 | | | | |
| 15 | 43 | 1.7647 | 1.9412 | 7 | 0 | | | | |
| 16 | 67 | 1.9412 | 2.1176 | 7 | 0 | | | | |
| 17 | 62 | 1.9412 | 2.1176 | 7 | 0 | | | | |
| 18 | 55 | 1.9412 | 2.1176 | 7 | 0 | | | | |
| 19 | 50 | 1.9412 | 2.1176 | 7 | 0 | | | | |
| 20 | 43 | 1.9412 | 2.1176 | 7 | 0 | | | | |
| 21 | 67 | 2.2941 | 2.4706 | 0 | 0 | | | | |

***Slika 4.4 - Matrica sa akordima. Druga i treća kolona daju početak i kraj (u sekundama) svakog tona (kolona 1) u okviru akorda. Broj u četvrtoj koloni predstavlja osnovni ton akorda a broj 12 u petoj koloni označava postojanje male terce.***

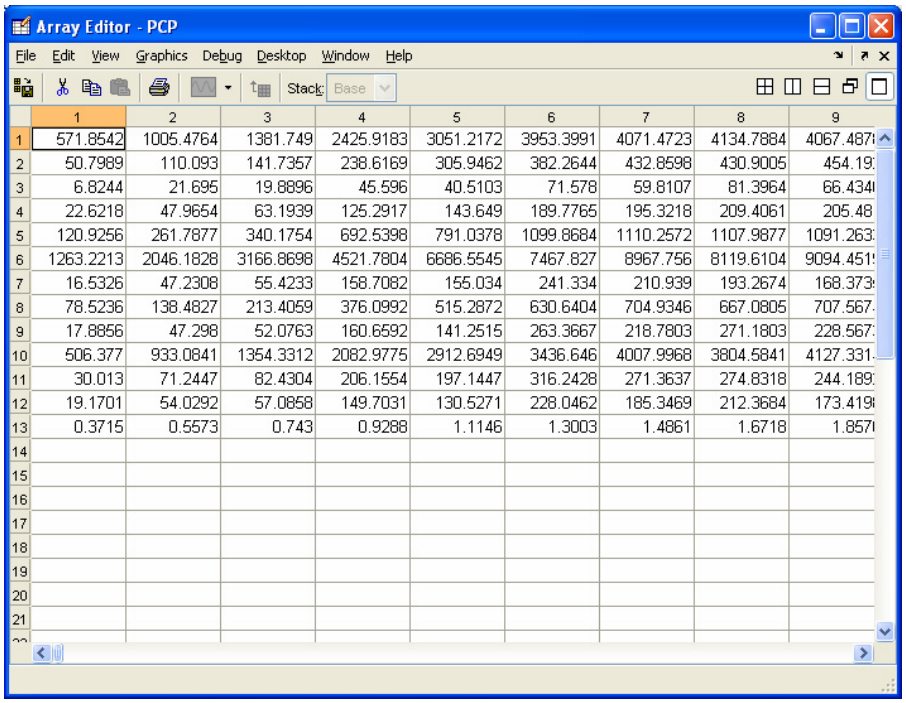
## 4.2 Ekstrakcija PCP obilježja iz audio fajlova

Blok 2 sa Slike 4.1 je u ovom radu realizovan funkcijom **PCP.m** (prilog). MIDI fajlovi se najprije konvertuju u 16-bitni mono *wav*, frekvencije odmjeravanja 11025Hz, pomoću programa *MID Converter*[1)] a nakon toga se na signal primjenjuje kratkotrajna *Fourier*ova transformacija, sa dužinom frejma od $N = 8192$ odmjeraka i korakom od 2048 odmjeraka.

Korištenjem jednačina (3.10) i (3.11) izračunati su PCP vektori za svaki audio fajl iz trening skupa podataka i s tim obilježjima se išlo u obučavanje HMM-a. Za referentnu frekvenciju iz jednačine (3.10) uzeta je vrijednost $f_{ref} = 2,04395Hz$ koja odgovara tonu $C_{-3}$ (znak minus u indeksu oktave ukazuje na to da ova frekvencija nije u čujnom opsegu, jer ljudsko uho ne čuje frekvencije ispod 20 Hz).

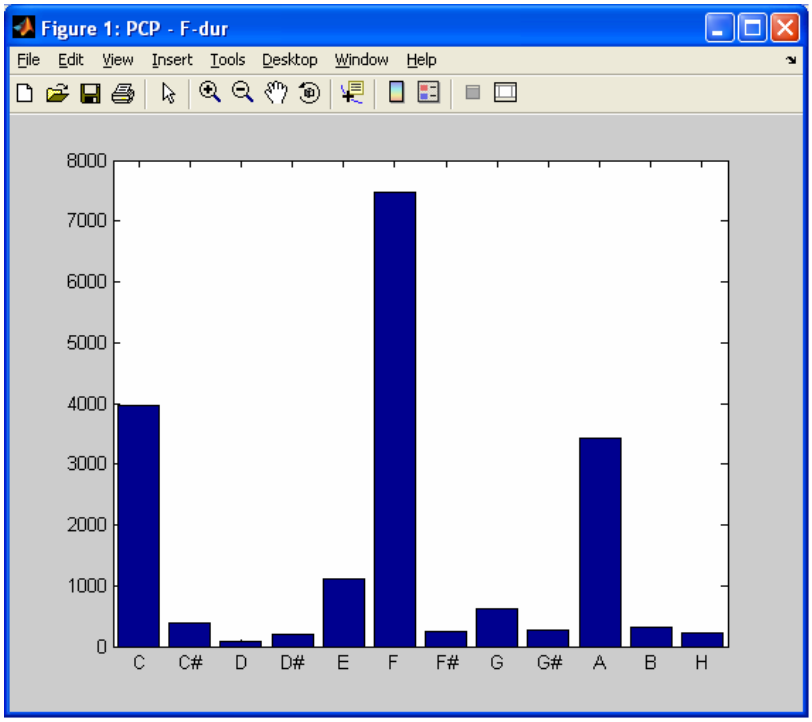
Na slici 4.5 dat je primjer PCP matrice pjesme *"Hey Jude"*, a na slici 4.6 dat je grafički prikaz jednog PCP vektora, izračunatog u trenutku $t = 1,3s$ (kolona 6), koji odgovara akordu F-dur. Lako je uočiti tri dominantna pika na pozicijama F, A i C. Ton F je osnovni ton akorda F-dur, dok tonovi A i C predstavljaju intervale velike terce i kvinte, respektivno. Osnovni ton, velika terca i kvinta čine durski trozvuk.

[1] http://midconverter.com




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 571.8542 | 1005.4764 | 1381.749 | 2425.9183 | 3051.2172 | 3953.3991 | 4071.4723 | 4134.7884 | 4067.487 |
| 2 | 50.7989 | 110.093 | 141.7357 | 238.6169 | 305.9462 | 382.2644 | 432.8598 | 430.9005 | 454.19 |
| 3 | 6.8244 | 21.695 | 19.8896 | 45.596 | 40.5103 | 71.578 | 59.8107 | 81.3964 | 66.434 |
| 4 | 22.6218 | 47.9654 | 63.1939 | 125.2917 | 143.649 | 189.7765 | 195.3218 | 209.4061 | 205.48 |
| 5 | 120.9256 | 261.7877 | 340.1754 | 692.5398 | 791.0378 | 1099.8684 | 1110.2572 | 1107.9877 | 1091.263 |
| 6 | 1263.2213 | 2046.1828 | 3166.8698 | 4521.7804 | 6686.5545 | 7467.827 | 8967.756 | 8119.6104 | 9094.451 |
| 7 | 16.5326 | 47.2308 | 55.4233 | 158.7082 | 155.034 | 241.334 | 210.939 | 193.2674 | 168.373 |
| 8 | 78.5236 | 138.4827 | 213.4059 | 376.0992 | 515.2872 | 630.6404 | 704.9346 | 667.0805 | 707.567 |
| 9 | 17.8856 | 47.298 | 52.0763 | 160.6592 | 141.2515 | 263.3667 | 218.7803 | 271.1803 | 228.567 |
| 10 | 506.377 | 933.0841 | 1354.3312 | 2082.9775 | 2912.6949 | 3436.646 | 4007.9968 | 3804.5841 | 4127.331 |
| 11 | 30.013 | 71.2447 | 82.4304 | 206.1554 | 197.1447 | 316.2428 | 271.3637 | 274.8318 | 244.189 |
| 12 | 19.1701 | 54.0292 | 57.0858 | 149.7031 | 130.5271 | 228.0462 | 185.3469 | 212.3684 | 173.419 |
| 13 | 0.3715 | 0.5573 | 0.743 | 0.9288 | 1.1146 | 1.3003 | 1.4861 | 1.6718 | 1.857 |
| 14 | | | | | | | | | |
| 15 | | | | | | | | | |
| 16 | | | | | | | | | |
| 17 | | | | | | | | | |
| 18 | | | | | | | | | |
| 19 | | | | | | | | | |
| 20 | | | | | | | | | |
| 21 | | | | | | | | | |

*Slika 4.5 – Primjer PCP matrice. U vrsti 13 je dat vremenski trenutak u kojem je računat PCP (centar prozora STFT-a).*




*Slika 4.6 – Grafički prikaz PCP vektora F-dur ("Hey Jude", t=1,3s).*

## 4.3 Inicijalizacija HMM-a

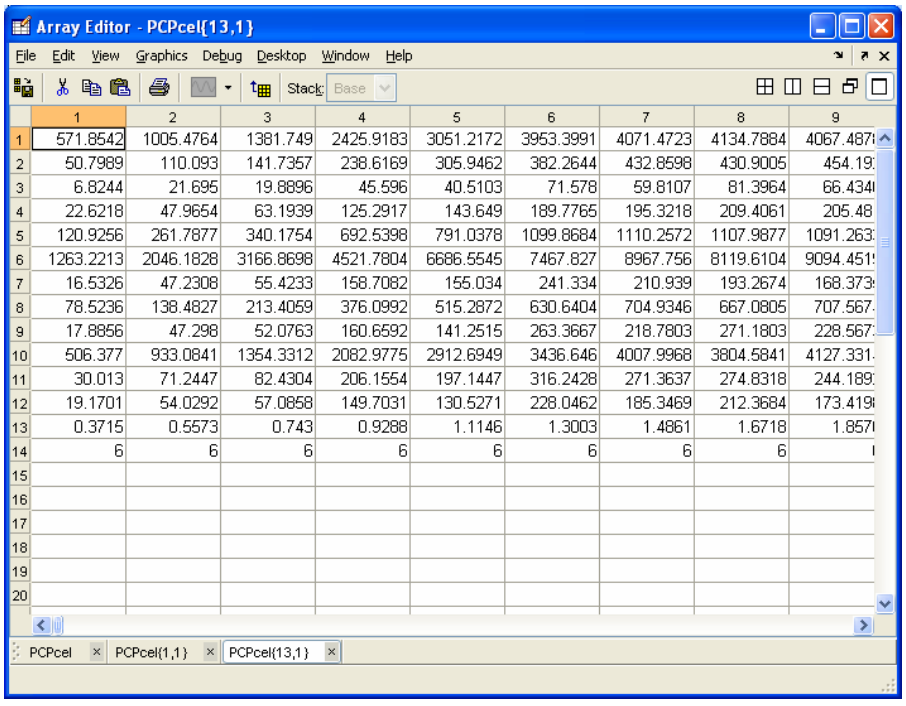
Podaci dobijeni pozivanjem funkcija **MIDIakordi.m** i **PCP.m** za 21 pjesmu koja se posmatra, smještaju se u ćelijsku strukturu PCPcel. Generisanje ćelijske strukture kao i kodovanje akorda ostvareno je u ovom radu u okviru skripte **HMMinic.m** (prilog). Kodovanje akorda dobijenih harmonijskom analizom MIDI-a ostvareno je sabiranjem elemenata četvrte i pete kolone matrice akorda (Slika 4.4). Broj u četvrtoj koloni predstavlja MIDI notaciju osnovnog tona a broj 12 u petoj koloni ukazuje na postojanje male terce. Ako je u petoj koloni broj 0, akord je durski i njegova oznaka je (*MIDI notacija osnovnog tona*) + 1, a ako je u petoj koloni broj 12, akord je molski i njegova oznaka će biti (*MIDI notacija osnovnog tona* + 12) + 1. Na osnovu informacija o akordima iz simboličkih podataka, izvršeno je označavanje PCP vektora upisivanjem labele u 14 vrstu PCP matrice svake pjesme (slika 4.6).

| Akord | C | C# | D | D# | E | F | F# | G | G# | A | B | H |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Oznaka | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

***Tabela 4.1 – Kodovanje durskih akorda.***

| Akord | Cm | C#m | Dm | D#m | Em | Fm | F#m | Gm | G#m | Am | Bm | Hm |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Oznaka | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |

***Tabela 4.2 – Kodovanje molskih akorda.***




***Slika 4.7 – Matrica označenih PCP vektora.***

## 4.4 HMM obuka i prepoznavanje

U okviru implementacije sistema za automatsko prepoznavanje akorda korišten je *Hidden Markov Model Toolbox for MATLAB – HMM Toolbox* [13]. Za obučavanje modela je korištena funkcija **gausshmm_train_observed,** koja ima oblik:

*[initState, transmat, mu, Sigma] = gausshmm_train_observed(obsData, hiddenData, nstates).*

Ulazne veličine su matrica opservacija *obsData*, vektor sa labelama skrivenih stanja *hiddenData* i broj stanja modela *nstates*. Izlazne veličine su vektor vjerovatnoća početnog stanja *initState*, matrica vjerovatnoća prelaza *transmat*, matrica srednjih vrijednosti *mu* i kovarijansna matrica *Sigma*.

Za prepoznavanje sekvence akorda korištene su funkcije **mixgauss_prob** i **viterbi_path** iz *HMM Toolbox*a. Prva funkcija prvo izračuna vjerovatnoću opservacija na osnovu zadatih parametara Gausove raspodjele, dok druga funkcija primjenjuje *Viterbijev* algoritam. Funkcije su oblika:

*vjerobs = mixgauss_prob(data, mu, Sigma)*

*sekvenca_akorda = viterbi_path(initState, transmat, vjerobs)*

Obučavanje skrivenog Markovljevog modela podacima iz trening skupa kao i prepoznavanje sekvence akorda implementirano je u ovom radu u obliku skripte **HMMoip.m** (prilog).

# 5. REZULTATI PREPOZNAVANJA AKORDA

Testiranje sistema za automatsko prepoznavanje akorda izvršeno je na svakoj pojedinačnoj pjesmi od njih 21, u smislu da se u svakom koraku odabere različita pjesma za prepoznavanje a ostalih dvadeset se koristi za obučavanje. Ovakav postupak daje dobar uvid u korelaciju između trening skupa i test fajla, odnosno oslikava zavisnost rezultata prepoznavanja sekvence akorda od izbora podataka za obučavanje.

Program za automatsko generisanje labela akorda u okviru trening skupa je testiran na pjesmi *Act Naturally*. Poređenjem sekvence akorda dobijene iz MIDI fajla sa ručno obilježenom sekvencom (*ground – truth*), od ukupno 804 frejma samo njih 17 je bilo pogrešno obilježeno, odnosno dobijena je tačnost od 97,9 %. Ovoliki procenat tačno obilježenih akorda je bio očekivan, s obzirom da harmonijska analiza simboličkih fajlova nije zasnovana na statističkim metodama već na jednom egzaktnom algoritmu, koji identifikuje svaku pojedinu notu u MIDI fajlu i na osnovu toga vrši obilježavanje labela trening skupa.

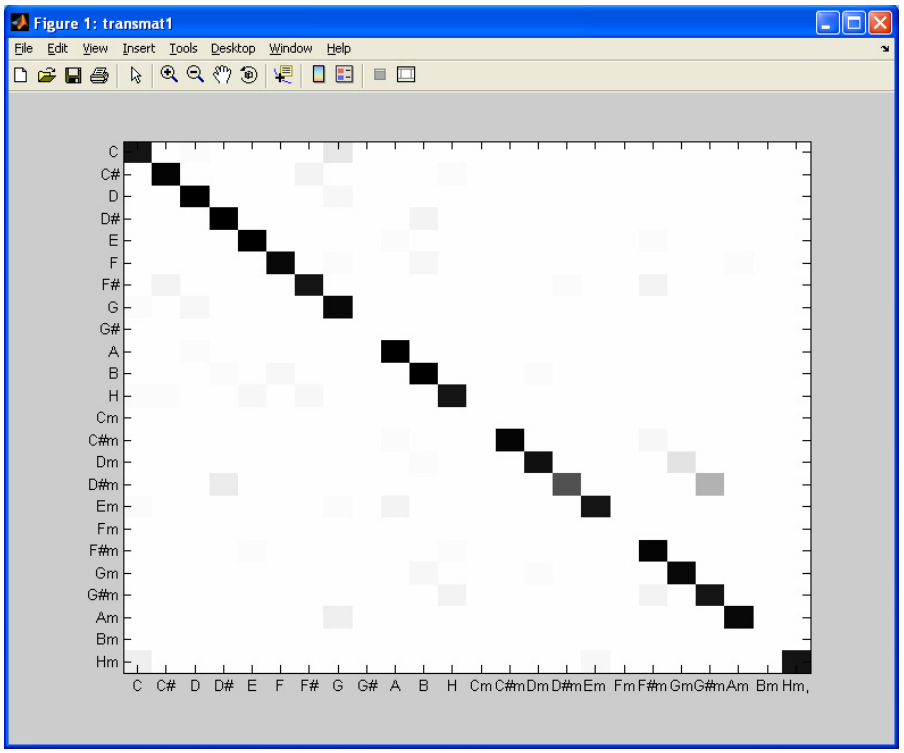
## 5.1 Primjer prepoznavanja akorda pri dobro odabranom trening skupu

Ako se za testiranje sistema odabere pjesma *Hey Jude*, a ostalih 20 pjesama za obučavanje modela, HMM će generisati sekvencu akorda koja u poređenju sa sekvencom dobijenom iz MIDI fajla daje tačnost od 80,4176 %. Pjesma *Act Naturally* izdvojena za testiranje takođe daje veoma visoku tačnost od 81,0945 %.

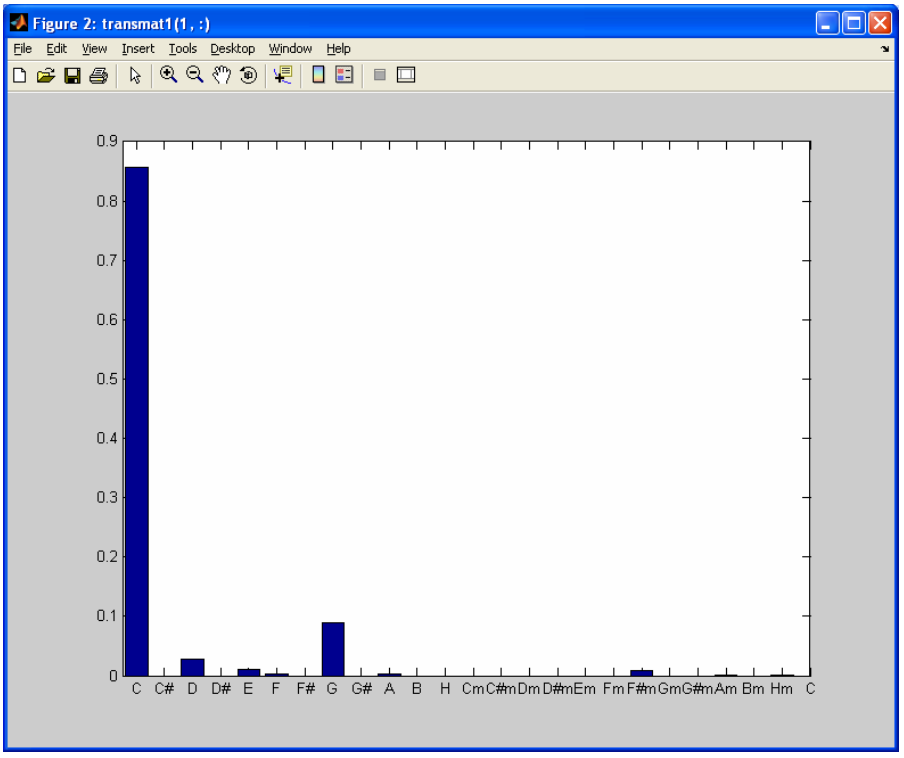
Objašnjenje ovih rezultata moguće je dobiti analiziranjem parametara skrivenog Markovljevog modela. Na slici 5.1 prikazana je matrica vjerovatnoća prelaza, estimirana iz trening skupa podataka. Lako je uočiti izrazitu dijagonalnost u strukturi matrice. To se objašnjava činjenicom da akordi uglavnom traju znatno duže od jednog frejma pa stanje sistema ostaje nepromijenjeno duž nekoliko frejmova. Drugim riječima, prilično je velika vjerovatnoća da će sistem zadržati posjećeno skriveno stanje.

Takođe, moguće je uočiti pravilnost u akordskoj progresiji zasnovanoj na muzičkoj teoriji [14]. Ako se posmatraju vjerovatnoće prelaza akorda C-dur (slika 5.2), kao što je i pomenuto, najveća vjerovatnoća je da će model ostati u istom stanju odnosno u akordu C-dur. S druge strane postoji velika vjerovatnoća prelaza na akord G-dur, koji predstavlja njegovu kvintu, kao i na D-dur, E-dur, A-dur, F-dur i F-mol.

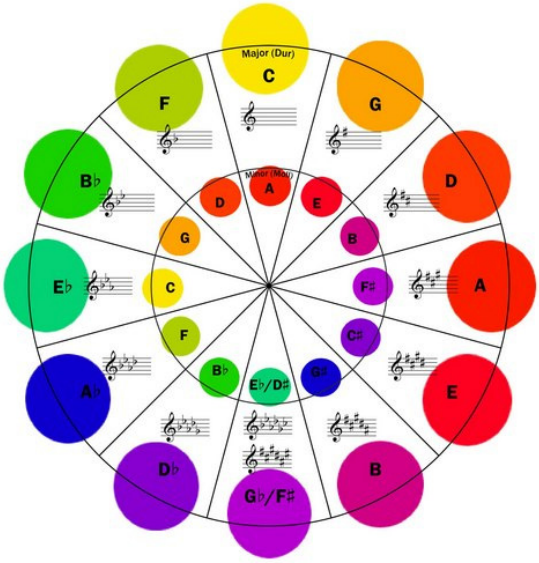
Poznato je da akordska progresija popularne zapadnjačke muzike često podliježe zakonitostima kvintnog kruga [15]. Posmatrajući sliku 5.3 može se uočiti da počevši od C, krečući se po kvintnom krugu dobijamo niz C – G – D – A – E, koji se poklapa sa akordima sa Slike 5.2. Akordi F-dur i F-mol takođe stoje u kvintnom odnosu sa akordom C-dur.




*Slika 5.1 – Matrica vjerovatnoća prelaza (24x24).*



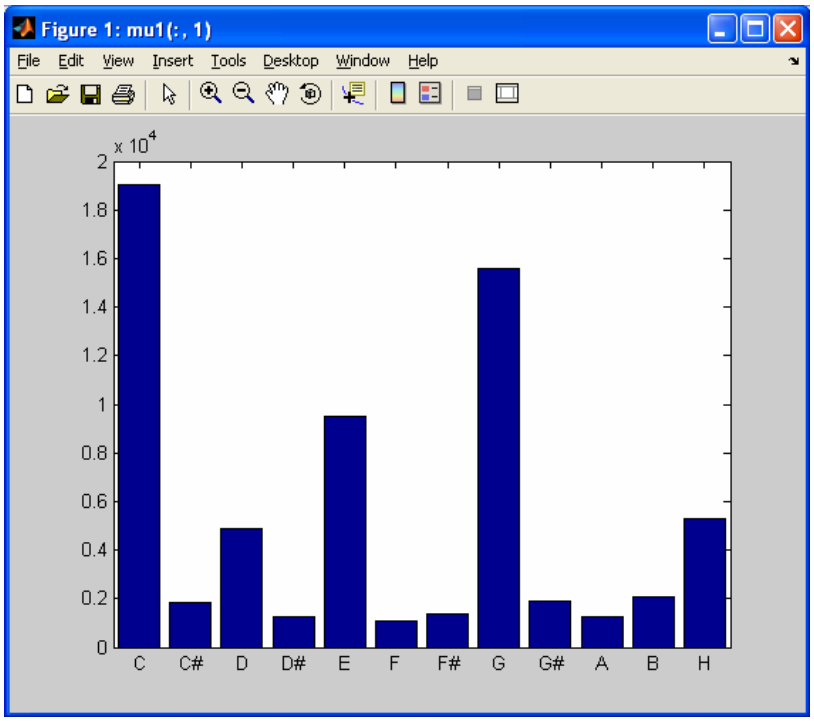

*Slika 5.2 – Vjerovatnoće prelaza – C-dur.*



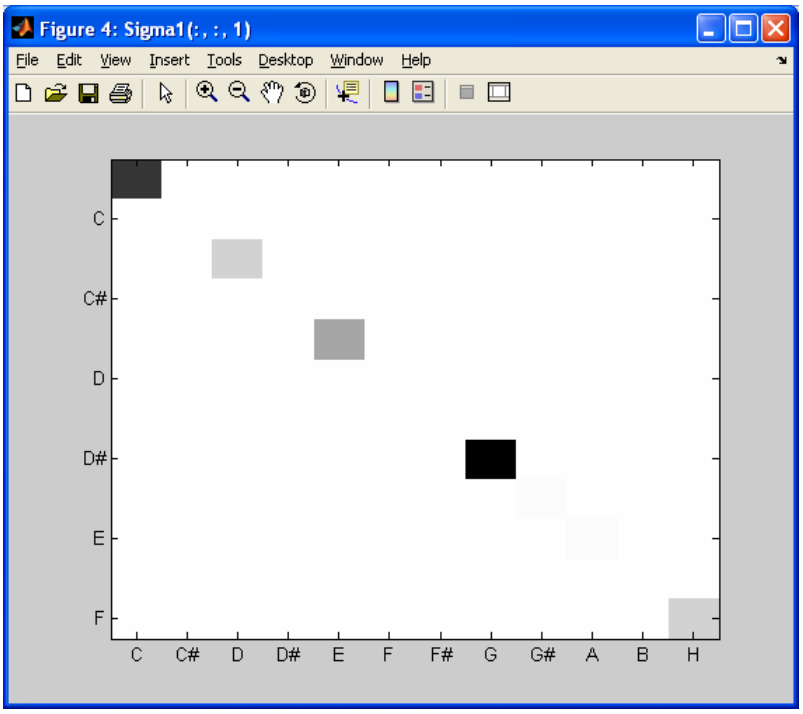

*Slika 5.3 – Kvintni (kvartni) krug.*

Slike 5.4 i 5.5 prikazuju parametre raspodjele opservacija, estimirane iz trening skupa podataka za akord C-dur. Očigledno je da prosječni PCP vektor sa slike 5.4, dobijen iz matrice srednjih vrijednosti, ima tri dominantna pika na tonovima C, E i G, koji čine akord C-dur, što je i bilo za očekivati.

Kovarijansna matrica za akord C-dur, sa Slike 5.5, je također dosljedna sa onim što se očekuje poznavajući muzičku teoriju. Tonovi C, E i G su strogo korelisani sami sa sobom, za razliku od ostalih tonova koji ne ulaze u sastav akorda C-dur.




*Slika 5.4 – Estimirani prosječni PCP vektor za akord C-dur.*



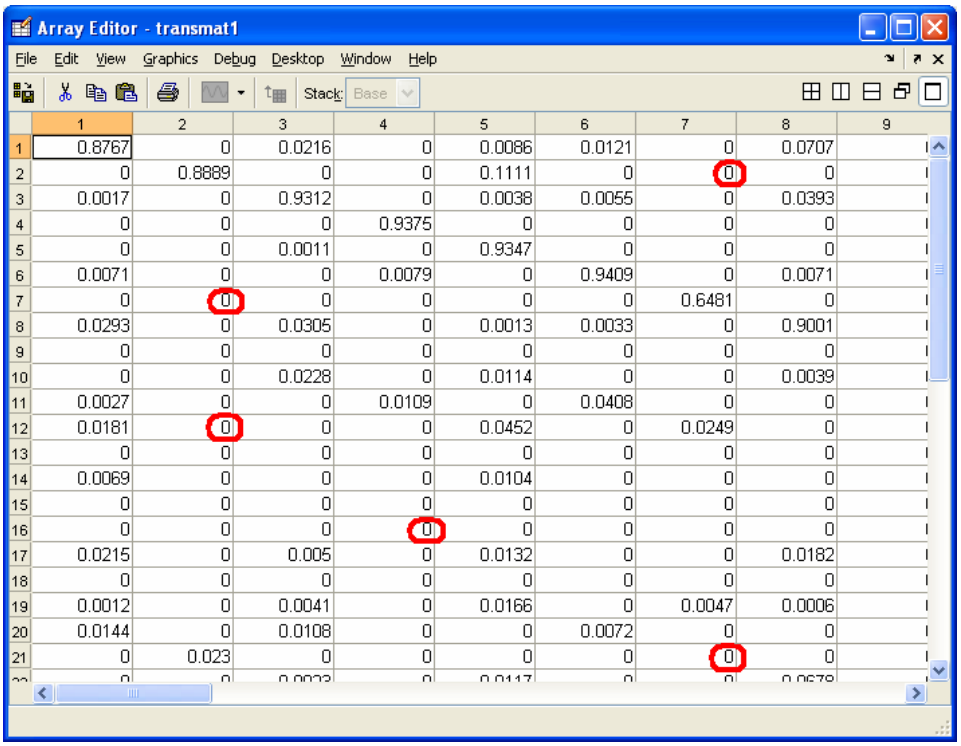

*Slika 5.5 – Kovarijansna matrica za akord C-dur.*

## 5.2 Primjer prepoznavanja akorda pri loše odabranom trening skupu

U ovom slučaju testiranje je izvršeno na pjesmi *Yellow Submarine*, za koju se unaprijed pretpostavljalo da će dati loš rezultat u pogledu prepoznavanja sekvence akorda. Naime, u programu *GuitarPro* promijenjeno je štimovanje gitare i sve žice su spuštene za pola stepena, a zatim je sa takvim štimom ekstraktovan MIDI fajl i iz njega generisan *wav*. Ovakvim postupkom promijenjen je i tonalitet pjesme a akordi su postali netipični u poređenju sa harmonijskom progresijom većine pjesama *Beatles*a. I zaista, model je prepoznao sekvencu akorda sa tačnošću od samo 9,5361 %.

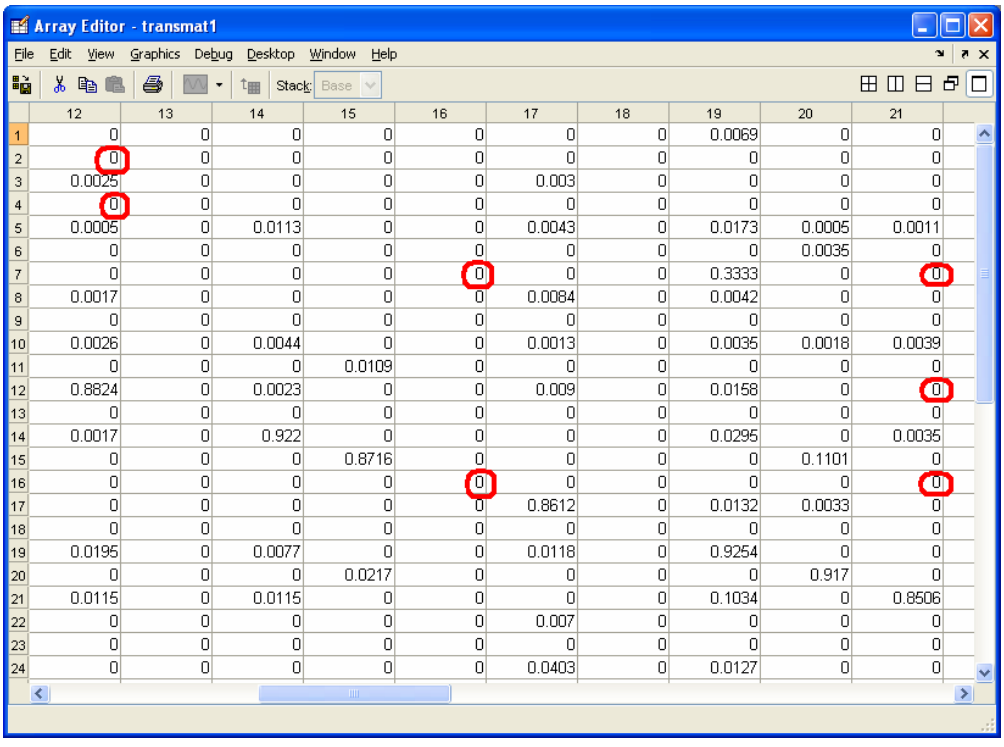
Posmatranjem prelaza između akorda koji se javljaju u pjesmi (dobijenih iz MIDI fajla) i matrice vjerovatnoća prelaza *transmat* obučenog HMM-a, može se objasniti zašto je ovako loš rezultat bio očekivan. Harmonijskom analizom simboličkih podataka utvrđeno je da su u pjesmi zastupljeni sljedeći akordi: $C^{\#}$-dur, $D^{\#}$-dur, $F^{\#}$-dur, H-dur, $D^{\#}$-mol i $G^{\#}$-mol. Od ukupno 19 prelaza koji se javljaju u pjesmi, čak 12 njih u matrici prelaza ima vjerovatnoću ravnu nuli (slike 5.6 i 5.7). Akord $D^{\#}$-mol iz pjesme se uopšte ne pojavljuje u trening skupu. U matrici srednjih vrijednosti, na poziciji tog akorda su nule (Slika 5.8).

Očigledno je da u ovom primjeru trening skup ne odgovara sekvenci akorda koju treba prepoznati tj. trening skup podataka ne nosi u sebi dovoljno informacija o prelazima koji se mogu javiti u pjesmi kojom je testiran sistem. Da bi se rezultat poboljšao potrebno je izabrati drugi skup podataka za obučavanje koji će u svom sastavu uključiti i "netipične" akorde test pjesme, kao i moguće prelaze i kombinacije tih akorda. Ovakav izbor će na kraju dovesti do toga da matrica prelaza nema nultih članova na mjestima koja su potrebna za tačno prepoznavanje akorda.



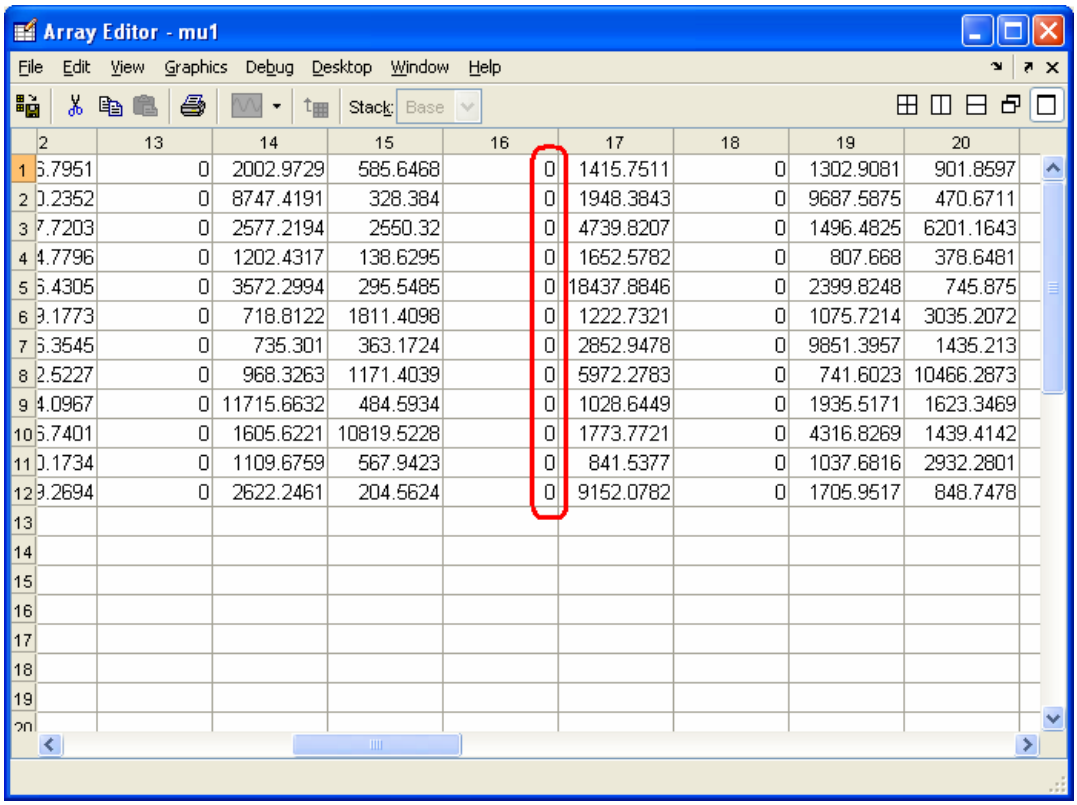

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.8767 | 0 | 0.0216 | 0 | 0.0086 | 0.0121 | 0 | 0.0707 | |
| 2 | 0 | 0.8889 | 0 | 0 | 0.1111 | 0 | 0 | 0 | |
| 3 | 0.0017 | 0 | 0.9312 | 0 | 0.0038 | 0.0055 | 0 | 0.0393 | |
| 4 | 0 | 0 | 0 | 0.9375 | 0 | 0 | 0 | 0 | |
| 5 | 0 | 0 | 0.0011 | 0 | 0.9347 | 0 | 0 | 0 | |
| 6 | 0.0071 | 0 | 0 | 0.0079 | 0 | 0.9409 | 0 | 0.0071 | |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0.6481 | 0 | |
| 8 | 0.0293 | 0 | 0.0305 | 0 | 0.0013 | 0.0033 | 0 | 0.9001 | |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 10 | 0 | 0 | 0.0228 | 0 | 0.0114 | 0 | 0 | 0.0039 | |
| 11 | 0.0027 | 0 | 0 | 0.0109 | 0 | 0.0408 | 0 | 0 | |
| 12 | 0.0181 | 0 | 0 | 0 | 0.0452 | 0 | 0.0249 | 0 | |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 14 | 0.0069 | 0 | 0 | 0 | 0.0104 | 0 | 0 | 0 | |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 17 | 0.0215 | 0 | 0.005 | 0 | 0.0132 | 0 | 0 | 0.0182 | |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 19 | 0.0012 | 0 | 0.0041 | 0 | 0.0166 | 0 | 0.0047 | 0.0006 | |
| 20 | 0.0144 | 0 | 0.0108 | 0 | 0 | 0.0072 | 0 | 0 | |
| 21 | 0 | 0.023 | 0 | 0 | 0 | 0 | 0 | 0 | |

*Slika 5.6 – Prelazi sa vjerovatnoćom nula.*




| | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0069 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0.0025 | 0 | 0 | 0 | 0 | 0.003 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0.0005 | 0 | 0.0113 | 0 | 0 | 0.0043 | 0 | 0.0173 | 0.0005 | 0.0011 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0035 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.3333 | 0 | 0 |
| 8 | 0.0017 | 0 | 0 | 0 | 0 | 0.0084 | 0 | 0.0042 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0.0026 | 0 | 0.0044 | 0 | 0 | 0.0013 | 0 | 0.0035 | 0.0018 | 0.0039 |
| 11 | 0 | 0 | 0 | 0.0109 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0.8824 | 0 | 0.0023 | 0 | 0 | 0.009 | 0 | 0.0158 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0.0017 | 0 | 0.922 | 0 | 0 | 0 | 0 | 0.0295 | 0 | 0.0035 |
| 15 | 0 | 0 | 0 | 0.8716 | 0 | 0 | 0 | 0 | 0.1101 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0.8612 | 0 | 0.0132 | 0.0033 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 0.0195 | 0 | 0.0077 | 0 | 0 | 0.0118 | 0 | 0.9254 | 0 | 0 |
| 20 | 0 | 0 | 0 | 0.0217 | 0 | 0 | 0 | 0 | 0.917 | 0 |
| 21 | 0.0115 | 0 | 0.0115 | 0 | 0 | 0 | 0 | 0.1034 | 0 | 0.8506 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0.007 | 0 | 0 | 0 | 0 |
| 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 24 | 0 | 0 | 0 | 0 | 0 | 0.0403 | 0 | 0.0127 | 0 | 0 |

*Slika 5.7 – Prelazi sa vjerovatnoćom nula (nastavak).*




| | 2 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 6.7951 | 0 | 2002.9729 | 585.6468 | 0 | 1415.7511 | 0 | 1302.9081 | 901.8597 |
| 2 | 0.2352 | 0 | 8747.4191 | 328.384 | 0 | 1948.3843 | 0 | 9687.5875 | 470.6711 |
| 3 | 7.7203 | 0 | 2577.2194 | 2550.32 | 0 | 4739.8207 | 0 | 1496.4825 | 6201.1643 |
| 4 | 4.7796 | 0 | 1202.4317 | 138.6295 | 0 | 1652.5782 | 0 | 807.668 | 378.6481 |
| 5 | 6.4305 | 0 | 3572.2994 | 295.5485 | 0 | 18437.8846 | 0 | 2399.8248 | 745.875 |
| 6 | 9.1773 | 0 | 718.8122 | 1811.4098 | 0 | 1222.7321 | 0 | 1075.7214 | 3035.2072 |
| 7 | 6.3545 | 0 | 735.301 | 363.1724 | 0 | 2852.9478 | 0 | 9851.3957 | 1435.213 |
| 8 | 2.5227 | 0 | 968.3263 | 1171.4039 | 0 | 5972.2783 | 0 | 741.6023 | 10466.2873 |
| 9 | 4.0967 | 0 | 11715.6632 | 484.5934 | 0 | 1028.6449 | 0 | 1935.5171 | 1623.3469 |
| 10 | 6.7401 | 0 | 1605.6221 | 10819.5228 | 0 | 1773.7721 | 0 | 4316.8269 | 1439.4142 |
| 11 | 0.1734 | 0 | 1109.6759 | 567.9423 | 0 | 841.5377 | 0 | 1037.6816 | 2932.2801 |
| 12 | 9.2694 | 0 | 2622.2461 | 204.5624 | 0 | 9152.0782 | 0 | 1705.9517 | 848.7478 |
| 13 | | | | | | | | | |
| 14 | | | | | | | | | |
| 15 | | | | | | | | | |
| 16 | | | | | | | | | |
| 17 | | | | | | | | | |
| 18 | | | | | | | | | |
| 19 | | | | | | | | | |
| 20 | | | | | | | | | |

***Slika 5.8 – Matrica srednjih vrijednosti - $D^{\#}$-mol.***

## 5.3 Eksperimentalni rezultati

Kao što je već spomenuto, testiranje modela izvršeno je na svakoj pjesmi iz skupa koga čini 21 pjesma *Beatles*a. Labele akorda i PCP vektori za svaku pjesmu, dobijeni u ovom radu pomoću funkcija **MIDIakordi.m** i **PCP.m**, smješteni su u okviru ćelijske strukture PCPcel. U svakom koraku odabrana je različita pjesma za prepoznavanje, a sa ostalim pjesmama se išlo u obučavanje modela.

Računanje procenata uspješnosti prepoznavanja akorda izvršeno je poređenjem sekvence akorda dobijene *Viterbijevim* algoritmom sa sekvencom dobijenom iz simboličkih podataka, i takođe je uključeno u okviru skripte **HMMoip.m.** Rezultati testiranja sistema na pjesmama iz posmatranog skupa, prikazani su u tabeli 5.1.

Prilično dobri rezultati, pored pjesama *Act Naturally* i *Hey Jude*, dobijeni su testiranjem na pjesmama *Baby It's You* (77,6952 %), *Don't Let Me Down* (73,6739 %), *All I've Got To Do* (66,6667 %), *Please Please Me* (66,6250 %), *Across the Universe* (66,5552 %), *A Hard Day's Night* (65,8135 %) i *Help* (60,5416 %). Sve ove pjesme u velikoj mjeri sadrže najčešće svirane akorde na gitari (poput C-dur, G-dur, D-dur, F-dur, E-dur, A-dur i H-dur od durskih i E-mol, A-mol, D-mol, H-mol i G-mol od molskih akorda), koji se u najvećem broju javljaju kod pjesama *Beatles*a. Pošto je model dobro obučen sa tim akordima, otuda je i veliki procenat uspješnosti prepoznavanja dobijen kod ovih pjesama.

Ostale pjesme sadrže u većoj mjeri "netipične" akorde, a neke od njih i žanrovski odstupaju od većine pjesama *Beatles*a. U okviru trening skupa podataka nije bilo dovoljnog broja pojavljivanja ovih akorda, kao i prelaza između njih, pa model nije dovoljno obučen da bi dao dobar rezultat prepoznavanja.

| Pjesma | Rezultat [%] |
|---|---|
| *A Hard Day's Night* | 65,8135 |
| *A taste of honey* | 28,6378 |
| *Across the Universe* | 66,5552 |
| *Act Naturally* | 81,0945 |
| *All I've Got To Do* | 66,6667 |
| *All my loving* | 56,9252 |
| *All You Need Is Love* | 39,0273 |
| *And I Love Her* | 26,2087 |
| *And Your Bird Can Sing* | 21,7042 |
| *Baby It's You* | 77,6952 |
| *Don't Let Me Down* | 73,6739 |
| *Help* | 60,5416 |
| *Hey Jude* | 80,4176 |
| *ObLaDi ObLaDa* | 4,6615 |
| *Please Please Me* | 66,6250 |
| *Roll Over Beethoven* | 29,2453 |
| *Tell Me Why* | 47,8955 |
| *Ticket to Ride* | 40,1648 |
| *While My Guitar Gently Weeps* | 17,9974 |
| *Yellow Submarine* | 9,5361 |
| *You've Really Got A Hold On Me* | 21,9842 |

***Tabela 5.1 – Rezultat testiranja izvršenog na svakoj pjesmi ponaosob.***

# 6. ZAKLJUČAK

Oblast pretraživanja muzičkih informacija (*Music Information Retrieval*), kao interdisciplinarna nauka, u posljednje vrijeme privlači sve veći broj istraživača. Ovo nije iznenađujuće s obzirom na mogućnosti koje MIR aplikacije pružaju kako u akademskom, edukativnom domenu tako i u domenu zabave. Sve do unazad par godina, sa stanovišta muzikologije, računari nisu imali veliku primjenu u analizi muzičkog zapisa. Korišteni su prosto za snimanje, čuvanje i reprodukciju i nudili su pojedine alate u pogledu produkcije i obrade. Međutim, sada postaje moguće obučiti računar da prepozna osnovne muzičke elemente, kao što su ritam, melodija, harmonija, tempo i sl.

U ovom radu pokazano je kako se uz pomoć računara može napraviti sistem za automatsko prepoznavanje akorda, korištenjem skrivenih Markovljevih modela. Akordi se tretiraju kao skrivena stanja modela dok se za obilježja koriste 12-dimenzionalni PCP vektori. Korišteni su simbolički podaci (MIDI) kako za generisanje labela tako i za kreiranje audio fajlova. Akcenat je stavljen na automatsko generisanje labela trening skupa, koje se koriste za obučavanje modela i iz kojih se, zajedno sa audio informacijama, direktno estimiraju parametri HMM-a.

Nakon obučavanja modela, audio fajl nepoznate sekvence akorda se dovodi na ulaz sistema i primjenjuje se *Viterbijev* algoritam koji pronalazi najvjerovatniji put kroz sistem, odnosno najvjerovatniju sekvencu.

Rezultati dobijeni testiranjem na pjesmama *Beatles*a pokazuju da je moguće dobiti visoku tačnost prepoznavanja akorda, pod uslovom da se odabere adekvatan skup podataka za obučavanje. Izbor trening skupa kao i procjena količine informacija potrebnih za obučavanje modela predstavljaju možda i najzahtjevniji dio cjelokupnog problema.

U budućem radu bi trebalo obratiti pažnju na sljedeće stvari:

- Proširiti analizu na mnogo veći skup podataka i ići na obučavanje modela sa minimalno stotinjak pjesama.
- Posvetiti vrijeme što tačnijem označavanju originalnih audio fajlova i sa njima ići u obučavanje modela.
- Zbog velikog uticaja harmonijskog sadržaja trening skupa na rezultat prepoznavanja, probati odabrati trening skup tako da pjesme odgovaraju istom žanru muzike a ne istom izvođaču.
- Povećati broj stanja modela uključivanjem drugih tipova akorda.
- Probati modelovati raspodjelu vidljivih stanja sa mješavinom Gausovih raspodjela.
- Povećati rezoluciju PCP vektora tako da oni budu 24-dimenzionalni, kako bi se uključile i male varijacije tonaliteta, odnosno frekvencija tonova.

Sistem koji će vršiti analizu muzičkih informacija na jednom višem nivou apstrakcije (sa stanovišta muzikologije, psihologije, pa čak i estetike) i koji će dati potpunu transkripciju kompleksnog muzičkog djela, predstavlja jedan od konačnih ciljeva istraživanja u oblasti automatskog izdvajanja muzičkih informacija.

# 7. PRILOG

## MATLAB skripte i funkcije:

### ***PCP.m***

```
%Funkcija koja računa STFT i PCP importovanog audio fajla,
%frekvencije odmjeravanja Fs, dodijeljenog varijabli Y:
function PCP=PCP(Y,Fs)
Fm=2.04395;
N=8192;

%Računanje STFT:
[X,F,T] = spectrogram(Y,hamming(N),6144,N,Fs);

%računanje PCP:
[frekvencija vrijeme]=size(X);
for k=1:frekvencija
p(k)=rem(round(12*log2((k/N)*(Fs/Fm))),12);
end

for n=1:vrijeme
   for pitch=0:11
      PCP(pitch+1,n)=0;
   for k=1:frekvencija
      if p(k)==pitch
         PCP(pitch+1,n)=PCP(pitch+1,n)+(abs(X(k,n)))^2;
      end
   end
   end
end
PCP(13,:)=T;
```

### ***MIDIakordi.m***

```
function gitara=MIDIakordi(ime)

%Funkcija koja određuje akorde u MIDI fajlu. Svakom pojedinačnom PCP
%vektoru dodijeljen je jedan akord (skriveno stanje).

midiime=[ime '.mid'];
midi=readmidi(midiime);
[note,endtime] = midiInfo(midi,0);

gitara=note(:,[3 5 6]);      %Uzimamo samo kolone 'note number',
                    %'start time' i 'end time'.

[vrstagit kolonagit]=size(gitara);
matricakraj=gitara(:,3);

k=1;
while k<=vrstagit
```

```
trenkraj=matricakraj(k);
indeksi=find(matricakraj==trenkraj);
[vrsteindeksi koloneindeksi]=size(indeksi);
osnovni=gitara(indeksi(1),1);
for i=2:vrsteindeksi
   osnovni=min(osnovni,gitara(indeksi(i),1));
end

for i=1:vrsteindeksi
   gitara(indeksi(i),4)=rem(osnovni,12);
end
k=k+vrsteindeksi;
end

%Rastojanje 'osnovni ton-mala terca': rem((X1-X2),12)==3.

for i=1:vrstagit
   gitara(1,5)=0;
end

k=1;
while k<=vrstagit
trenkraj=matricakraj(k);
indeksi=find(matricakraj==trenkraj);
[vrsteindeksi koloneindeksi]=size(indeksi);
for i=1:vrsteindeksi
   if rem(((gitara(indeksi(i),1))-(gitara(indeksi(i),4))),12)==3
      for i=1:vrsteindeksi
      gitara(indeksi(i),5)=12;
      end
   end
end
k=k+vrsteindeksi;
end
```

## ***HMMoip.m***

```
%Obučavanje HMM-a i prepoznavanje sekvence akorda. Svaki put uzimamo
%različit skup podataka (fajlova) za obučavanje i računamo procenat
%uspješnosti prepoznavanja za fajl s kojim nismo obučavali model.
%Na kraju dobijene rezultate (procente) usrednjimo.

%(Prije pokretanja skripte potrebno je pokrenuti HMMinic.m
%ili učitati fajl PCPcel21.mat)
%-------------------------------------------------------------------

%Izdvajanje pjesme za prepoznavanje:
%-----------------------------------
[vrstaPCP kolonaPCP]=size(PCPcel);
brojfajlova=vrstaPCP-1;
for j=1:vrstaPCP
   PCPnep=PCPcel(j,:);          %PCP vektori nepoznate pjesme.
   PCPobuka=PCPcel;
   PCPobuka(j,:)=[];            %Uklanjanje vektora nepoznate pjesme
                                %iz trening seta.
```

```
%Obučavanje HMM-a:
%-----------------
for i=1:brojfajlova
   [brvrsta brkolona]=size(PCPobuka{i,1});
   for k=1:brkolona
      obsData{i}(:,k)=PCPobuka{i,1}(1:12,k);
      hiddenData{i}(k)=PCPobuka{i,1}(14,k);
   end
end

[initState1, transmat1, mu1, Sigma1] = gausshmm_train_observed(obsData, hiddenData, ...
                    24,'cov_type','diag');


%Prepoznavanje sekvence akorda:
%------------------------------
vjerobs=mixgauss_prob(PCPnep{1,1}(1:12,:), mu1, Sigma1);
sekvenca_akorda = viterbi_path(initState1, transmat1, vjerobs);


%Računanje procenta pogođenih frejmova (akorda):
%-----------------------------------------------
clc
pogodjeno=find(sekvenca_akorda==PCPnep{1,1}(14,:));
procenat(j)=((length(pogodjeno))/(length(sekvenca_akorda)))*100;
end

%Usrednjeni procenat:
%--------------------
procenat
suma=sum(procenat);
srprocenat=suma/length(procenat)
```

## ***HMMinic.m***

```
close all, clear all,clc

%Importovanje audio fajlova i formiranje æelijskog polja PCPcel.
%Vrste odgovaraju pojedinim fajlovima. U prvoj æeliji vrste su smješteni
% PCP vektori a u drugoj i treæoj akordi za svaki vektor.
%-------------------------------------------------------------
nazivipjesama=input('Unesite ime txt fajla sa naslovima pjesama: ','s');
disp(' ')
brojfajlova=input('Unesite broj audio fajlova koje želite importovati: ');
str=[nazivipjesama '.txt'];
fid = fopen(str);
celija = textscan(fid, '%s', 'delimiter', '.');
naslovi=celija{1};
for i=1:brojfajlova
[Yime Fs]=wavread(naslovi{i});
PCPcel{i,1}=PCP(Yime,Fs);
PCPcel{i,2}=MIDIakordi(naslovi{i});
PCPcel{i,3}=unique(PCPcel{i,2}(:,[3 4 5]),'rows');
end

%-------------------------------------------------------------------
%Kodovaæemo akorde na sljedeæi naèin:
```

```
% C=1 C#=2 D=3 D#=4 E=5 F=6 F#=7 G=8 G#=9 A=10 B=11 H=12; Cm=13      C#m=14
% Dm=15 D#m=16 Em=17 Fm=18 F#m=19 Gm=20 G#m=21 Am=22 Bm=23 Hm=24
%---------------------------------------------------------------------

for i=1:brojfajlova
   [brvrsta brkolona]=size(PCPcel{i,1});
   for k=1:brkolona
      PCPt=PCPcel{i,1}(13,k);
      krajakorda=find(PCPcel{i,3}(:,1)>=PCPt);
      indeks=0;
      if length(krajakorda)~=0
      indeks=krajakorda(1);
      end
      if indeks==0

PCPcel{i,1}(14,k)=PCPcel{i,3}(length(PCPcel{i,3}),2)+PCPcel{i,3}(length(PCPcel{i,3}),3)+1;
      else
      PCPcel{i,1}(14,k)=PCPcel{i,3}(indeks,2)+PCPcel{i,3}(indeks,3)+1;
      end
   end
end
```

***Funkcije pomoću kojih su generisane slike u MATLAB-u:***

```
T=mat2gray(transmat1, [1 0]);
imagesc(T),colormap(gray);

figure,bar(transmat1(1,:))

figure,plot(mu1(:,1))

S=mat2gray(Sigma1(:,:,1), [Smax 0]);
figure,imagesc(S),colormap(gray);
```

## Lista oznaka i skraćenica:

CD – *Compact Disc*
DFT – *Discrete Fourier Transform*
EM – *Expectation Maximization*
FFT – *Fast Fourier Transform*
HMM – *Hidden Markov Model*
MATLAB – *MATrix LABoratory*
MIDI – *Musical Instruments Digital Interface*
MIR – *Music Information Retrieval*
PCP – *Pitch Class Profile*
STFT – *Short Time Fourier Transform*

# LITERATURA


[1] L.R. Rabiner, "A tutorial on Hidden Markov models and selected applications in speech recognition", *Proceedings of the IEEE*, vol. 77, no.2, pp. 257-286, 1989.

[2] R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification*, 2nd edition, John Wiley and Sons, 2001.

[3] K. Lee, M. Slaney, "Automatic chord recognition from audio using an HMM with supervised learning", *in Proceedings of the International Symposium on Music Information Retrieval*, 2006.

[4] A. Sheh, D.P. Ellis, "Chord segmentation and recognition using EM-trained hidden Markov models", *in Proceedings of the International Symposium on Music Information Retrieval*, 2003.

[5] http://ccrma.stanford.edu/~jos/mus423h/Real_Time_Chord_Recognition_Musical.html
posjećeno: decembar 2007.

[6] http://en.wikipedia.org/wiki/Pitch_classes
posjećeno: decembar 2007.

[7] L.R. Rabiner, R.W. Schafer, *Digital processing of speech signals*, Prentice Hall, 1978.

[8] http://en.wikipedia.org/wiki/Equal_temperament
posjećeno: decembar 2007.

[9] Z. Babić, *Analogni i digitalni filtri*, skripta, Elektrotehnički fakultet Banjaluka, 2007.

[10] J. Bilmes, *What HMMs can do*, Dept of EE, University of Washington, 2002.

[11] http://www.kenschutte.com/midi
posjećeno: decembar 2007.

[12] http://www.sengpielaudio.com/calculator-notenames.htm
posjećeno: decembar 2007.

[13] http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html
posjećeno: decembar 2007.

[14] http://en.wikipedia.org/wiki/Music_theory
posjećeno: decembar 2007.

[15] http://en.wikipedia.org/wiki/Circle_of_fifths
posjećeno: decembar 2007.

# SADRŽAJ



*Napomena: Uz rad je priložen CD.