SVEUČILIŠTE U ZAGREBU
**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

DIPLOMSKI RAD br. 537

# Estimacija frekvencije osnovnog tona pomoću valićne i Fourierove transformacije

Nader Salameh

Zagreb, lipanj 2012.

***ZAHVALE***

*Zahvaljujem svom mentoru prof. dr. sc. Davoru Petrinoviću na strpljenju i pomoći tokom cijelog studija. Posebno se zahvaljujem Ivi "Jovanotti" Jarić koja mi je svojim stručnim znanjem pomogla oko svih glazbenih pitanja.*

*Najviše bi se želio zahvaliti roditeljima koji su mi omogućili ovaj studij.*

## Sadržaj

# Uvod

U okviru ovog rada potrebno je istražiti postupke estimacije osnovnog tona pomoću valićne i Fourierove transformacije. Detekcija osnovnog tona služila bi za transkripciju zvučnog signala. Transkripcija zvučnog signala može se definirati kao slušanje glazbenog djela te zapisivanje notnog zapisa. Ukoliko se promatra monofoni signal, tj. maksimalno jedan ton istovremeno, problem je prilično jednostavan. Situacija se jako komplicira kod polifonih signala te se za ovakve primjene još uvijek razvijaju različiti sustavi.

Automatska transkripcija glazbe zasigurno bi pronašla svoje mjesto u glazbenim krugovima. Neki od potencijalnih primjena su:

- Sustav gdje se odsvirana melodija uspoređuje s gotovim notnim zapisom.
- Ukoliko glazba postoji samo kao audio zapis, postupak prevođenja u notni zapis bio bi kudikamo jednostavniji.
- Čin kompozicije bi se ubrzao jer bi kompozitor mogao veći dio svoje koncentracije usmjeriti na glazbu umjesto na pisanje.

Sam sustav mogao bi se podijeliti na više dijelova. Prvi dio svakako bi bilo prebacivanje signala u frekvencijsku domenu, jer je najbliža fizička veličina koja opisuje ton upravo frekvencija. Idealno bi bilo promatrati kako se kroz vrijeme pojavljuju frekvencije (tj. tonovi) u signalu. Kontinuirana valićna i kratkotrajna Fourierova transformacija dat će reprezentaciju signala u vremensko-frekvencijskoj ravnini. Koje su mane i prednosti jedne i druge transformacije te konačno koja je bolja, treba vidjeti na primjerima. Nakon što se signal prebacio u vremensko-frekvencijsku domenu potrebno je pomoću te interpretacije detektirati frekvencije tj. tonove koji se pojavljuju u signalu. To će ujedno biti i drugi korak analize. Načine detekcije potrebno je tek ustvrditi te odabrati najbolji. Završni korak, pronalazak je osnovnih frekvencija signala kako bi se transkripcija mogla realizirati u potpunosti. Kao i kod detekcije, postoji više metoda a ono što preostaje je odrediti koja najviše odgovara potrebama.

# 1. Kratkotrajna Fourierova transformacija

## 1.1. Motivacija

Fourierova transformacija daje idealan uvid u frekvencijski sadržaj analiziranog signala. Naravno, signal nad kojim se vrši Fourierova transformacija mora zadovoljavati određena svojstva. Dovoljni uvjeti za postojanje Fourierove transformacije su:

1) Snažni Dirichletovi uvjeti
   - Signal koji se promatra mora imati konačan broj minimuma i maksimuma na bilo kojem intervalu $t$.
   - Signal koji se promatra mora imati konačan broj diskontinuiteta na bilo kojem konačnom intervalu $t$.
2) Slabi Dirichletovi uvjeti
   - Signal mora biti apsolutno integrabilan

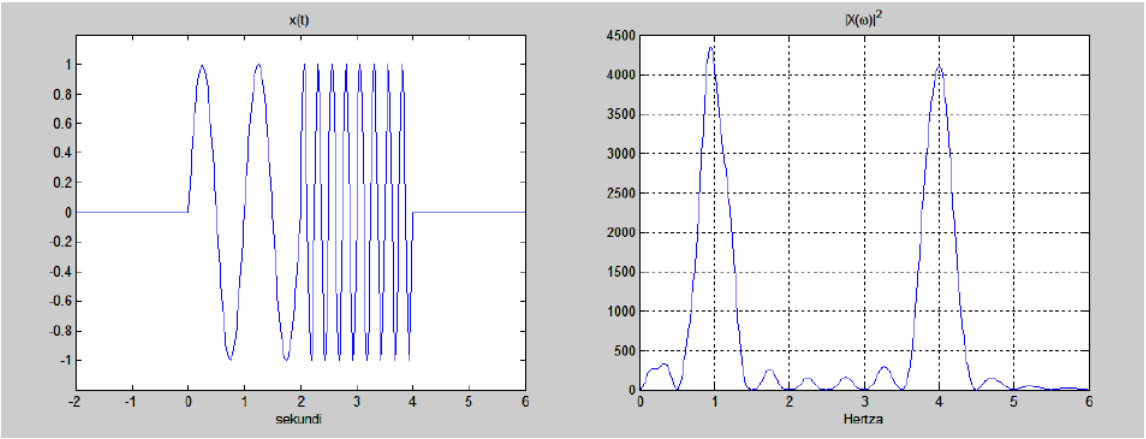
Nakon što je signal zadovoljio uvjete slijedi integracija umnoška signala te kompleksne harmonijske funkcije tj. :

$$\int_{-\infty}^{\infty} x(t)e^{-j\omega t} = X(\omega) \qquad (1)$$

$X(\omega)$ zapravo označava mjeru sličnosti između signala $x(t)$ te kompleksne harmonijske funkcije $e^{j\omega t}$.

Kao što je već spomenuto, Fourierovom transformacijom dobiva se uvid u frekvencijski sadržaj analiziranog signala. Frekvencijski sadržaj zapravo označava harmonijske funkcije koje se pojavljuju u tom signalu. No, te harmonijske funkcije nisu lokalizirane u vremenu. Drugim riječima jedan od manjka Fourierove transformacije je taj što ona ne daje informaciju o vremenu u kojem se pojedina harmonijska funkcija pojavila[3]. Ovo nije problem kod stacionarnih signala, no sigurno su potrebna poboljšanja kada je riječ o nestacionarnim signalima.

Pretpostavimo sljedeću situaciju: razvija se sustav koji za sinusoidu od 1 Hz radi određenu akciju A, dok za sinusoidu od 4 Hz radi akciju B. Sljedeća slika pokazuje vremenski signal koji se sastoji od obje sinusoide ali u različitim vremenskim trenutcima.




Slika 1 - Signal sastavljen od dvije sinusoide

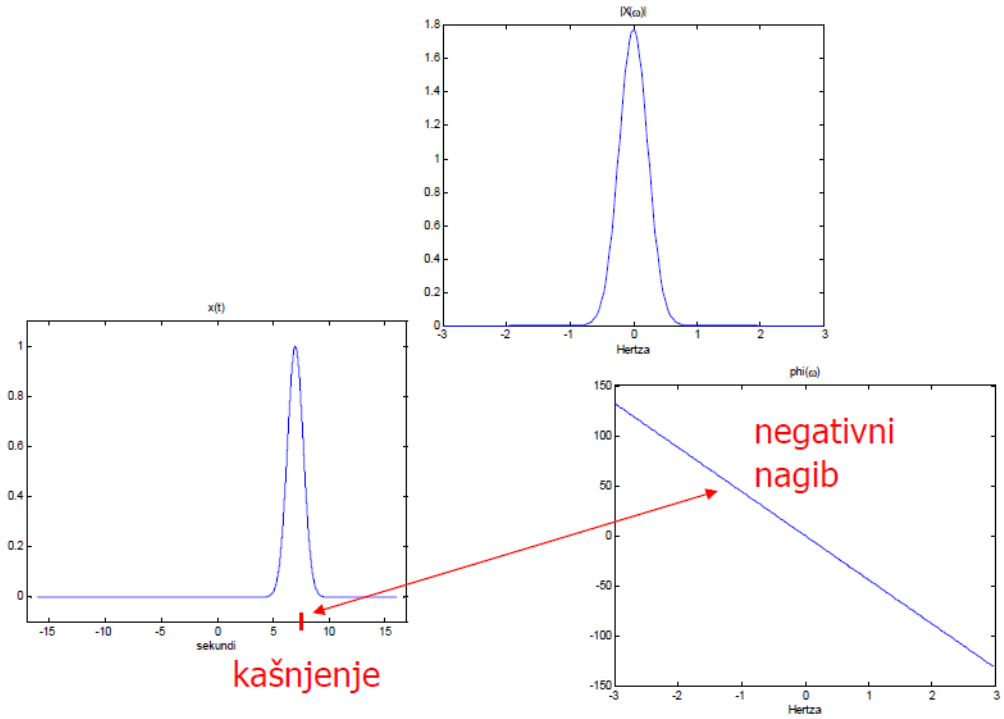
Kao što se vidi sa slike, Fourierovom transformacijom može se ili promatrati frekvencijske odnose, ili promatrati vremenske odnose. No niti jedan prikaz nije zadovoljavajući, jer se ne zna kada treba poduzeti akciju A tj. akciju B.

Međutim vremenski odnosi su ugrađeni u frekvencijsku karakteristiku, ali nisu uvijek eksplicitno vidljivi. Poznato je da osim amplitudne frekvencijske karakteristike postoji i fazna frekvencijska karakteristika, koja sadržava informaciju o vremenskim odnosima. Ono što je potrebno napraviti je derivirati faznu karakteristiku kako bi se dobilo grupno kašnjenje $T(\omega)$. Matematički to izgleda ovako:

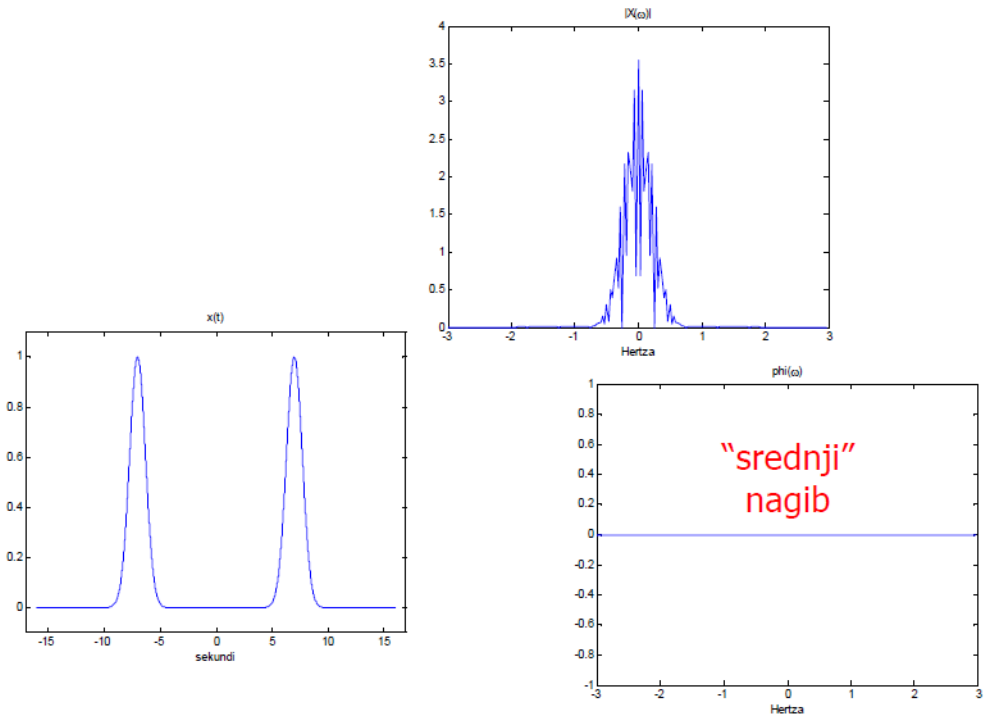
$$X(\omega) = A(\omega)e^{j\varphi(\omega)},\ T(\omega) = -\frac{d\varphi(\omega)}{d\omega}, \qquad (2)$$

gdje je $A(\omega)$ amplitudno frekvencijska, a $\varphi(\omega)$ fazno frekvencijska karakteristika.

Sljedeća slika pokazuje signal u vremenu, amplitudnu frekvenciju karakteristiku signala te faznu frekvencijsku karakteristiku. Može se primijetiti kako nagib fazne karakteristike ovisi o kašnjenju signala. Što signal više kasni nagib postaje sve veći. Čini se kako je za ovu situaciju Fourierova transformacija dovoljna, no što je sa situacijom gdje se pojavljuju 2 signala? Odgovor se nalazi na slici broj 3.




Slika 2 – Signal s vremenskim pomakom




Slika 3 - Signal s dva vremenska pomaka

Slično kao i u prijašnjem primjeru, vremenska domena sadrži signal koji kasni određeni iznos, ali mu se pridodaje drugi signal koji je uranio za isti iznos kašnjenja prvog signala. Amplitudna karakteristika trenutno nije od značaja, ali se vidi da je došlo do promjene kod fazne frekvencijske karakteristike. Za razliku od prvog primjera, faza u ovom slučaju nema nikakav nagib. Ako se izostavi vremenska domena, može se doći do (krive) informacije, tj. da su signali bez pomaka u vremenu.

Promatrajući ova dva primjera zaključuje se kako grupno kašnjenje ima jasan smisao ukoliko se frekvencijski sadržaj analiziranog signala događa u jednoj vremenskoj točki. Ukoliko se identični frekvencijski sadržaj analiziranog signala događa u više vremenskih točaka, grupno kašnjenje neće biti dobra mjera zbog svoje višeznačnosti.

Upravo radi ovakvih potreba razvila se nova transformacija koja će istovremeno promatrati vremenske i frekvencijske odnose signala. Transformacija je dvodimenzionalna i naziva se kratkotrajna (vremenski kratka) Fourierova transformacija (eng. *Short Time Fourier Transform,* STFT).

## 1.2. STFT

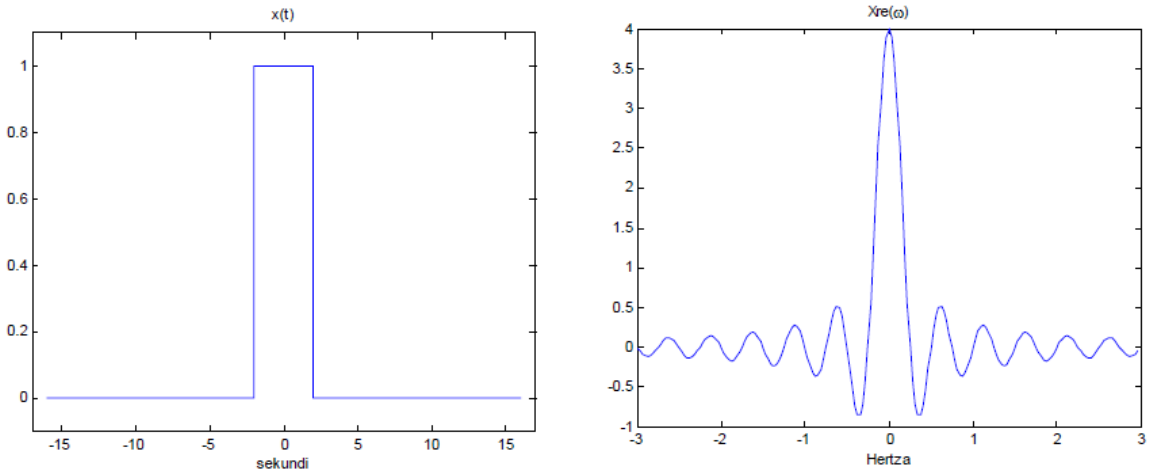
Razlika između obične i vremenski kratke Fourierove transformacije je u tome što kompleksnoj harmonijskoj funkciji $e^{j\omega t}$ dodajemo lokalni analizirajući otvor $g(t)$. To je mali vremenski otvor s određenim željenim karakteristikama u obje domene. Osim toga vremenski otvor je pomičan u vremenu te stoga sadrži i vremenski pomak $\tau$.

$$X(\tau,\omega)=\int_{-\infty}^{\infty} x(t)g^{*}(t-\tau)e^{-j\omega t}dt \qquad (3)$$

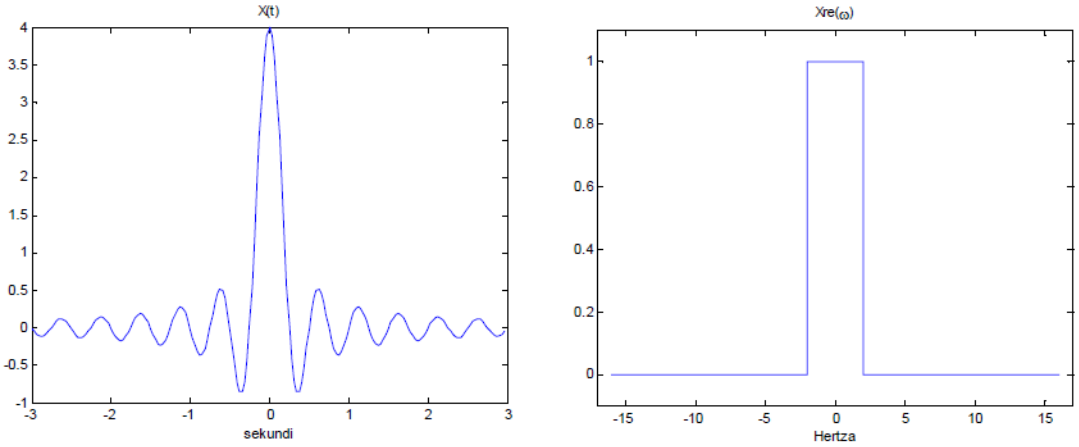
Funkcija koja se treba transformirati množi se sa vremenskom funkcijom koja je različita od nule samo u kratkom vremenskom periodu. Računa se Fourierova transformacija (jednodimenzionalna funkcija) dobivenog umnoška, dok se vremenski otvor pomiče po vremenskoj osi, dobivajući pritom dvodimenzionalnu reprezentaciju početnog signala.

Jasno je kako vremenski kratka Fourierova transformacija nije uvijek identična. Njezino rješenje uvelike ovisi o izboru vremenskog otvora tj. funkcije $g(t)$. Kako izabrati odgovarajuću funkciju $g(t)$?

Veliki problem kod izbora funkcije stvaraju veze između frekvencijske i vremenske domene. Naime, ukoliko je signal vremenski ograničen ne može biti frekvencijski ograničen. Također vrijedi i obrnuta relacija tj. frekvencijski ograničen signal ne može biti vremenski ograničen.



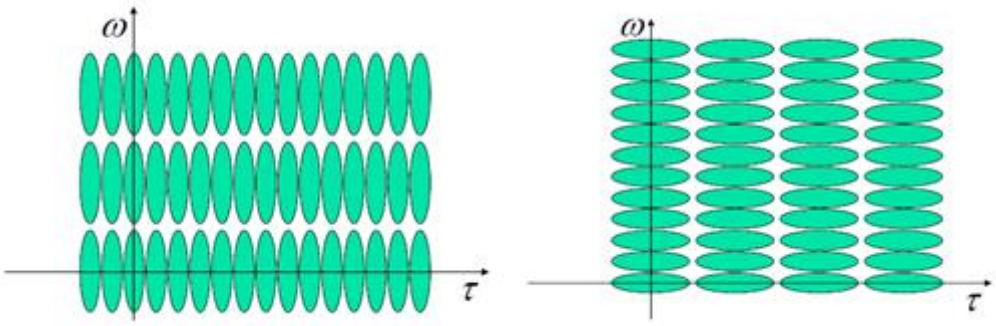

Slika 4 - Signal ograničen u vremenskoj domeni




Slika 5 - Signal ograničen u frekvencijskoj domeni

Kao primjer će poslužiti prethodne dvije slike. Ukoliko odaberemo konačni pravokutni signal u vremenu, dobiti ćemo beskonačnu *sinc* funkciju. Vrijedi i obrnuta relacija. Dobro rješenje je koristiti Gaussovu funkciju. Iako nije konačna u vremenu, veliki dio njezine energije nalazi se u relativno malom pojasu. Također Fourierova transformacija Gaussove funkcije dat će opet Gaussovu funkciju. Vremenski kratka Fourierova transformacija koja koristi Gaussovu funkciju $g_\alpha(t)$ za težinsku tj. vremensku funkciju naziva se još i Gaborova transformacija.

Još jedan problem, koji se javlja uslijed ovisnosti vremenske i frekvencijske domene, je problem razlučivosti u vremensko-frekvencijskoj ravnini. Poznata je recipročna veza između ove dvije domene. Naime, ukoliko je potrebna fina razlučivost u vremenu koristit ćemo što manji vremenski otvor (vremenski pojas) kako bi se točno znalo u kojem trenutku se pojavila određena harmonijska funkcija. Problem je u tome što kratak vremenski pojas znači široki frekvencijski pojas. Dakle, ukoliko je potrebna fina rezolucija u vremenskoj domeni, dobiva se loša rezolucija u frekvencijskoj domeni. Naravno, vrijedi i obrnuta situacija.



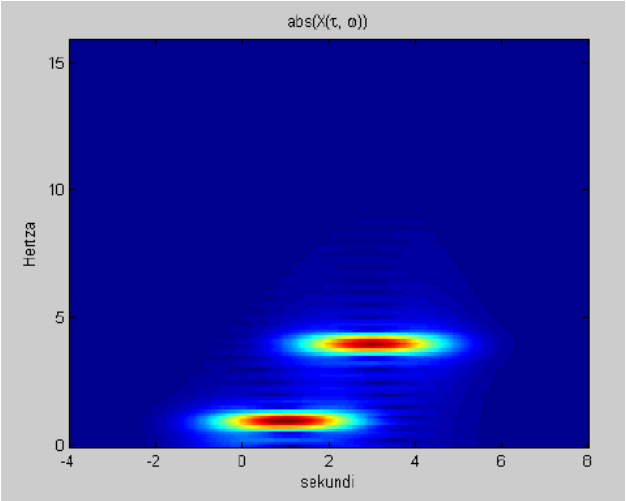

Slika 6 - Kompromis između dobre frekvencijske i vremenske rezolucije

Ovaj kompromis, koji međusobno isključuje dobru lokalizaciju u vremenu i frekvenciji, objašnjava se Heisenbergovim principom neodređenosti. On govori (u verziji prilagođenoj ovom slučaju) kako je razlučivost u vremensko-frekvencijskoj domeni određena širinom vremenskog otvora $\sigma_g$ i širinom frekvencijskog otvora $\sigma_G$:

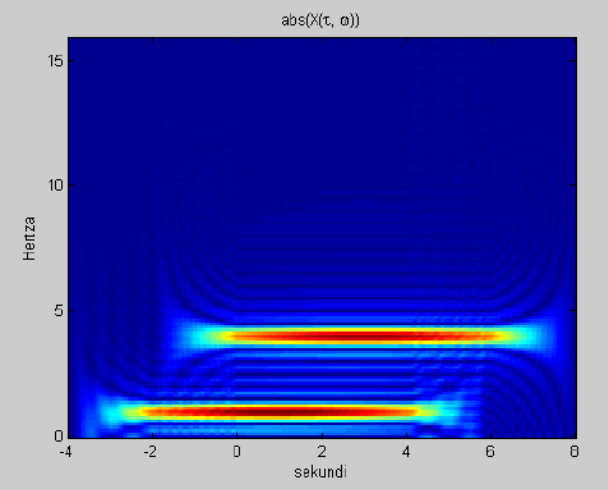
$$\sigma_g \sigma_G \geq c\,. \tag{4}$$

Kako bi se postigla što preciznija lokalizacija u vremensko-frekvencijskoj domeni, potrebno je odabrati funkciju za vremenski otvor kod koje će konstanta $c$ biti najmanja. Funkcija s najmanjom vrijednošću konstante $c$ upravo je Gaussova funkcija, kod koje ona iznosi ½.

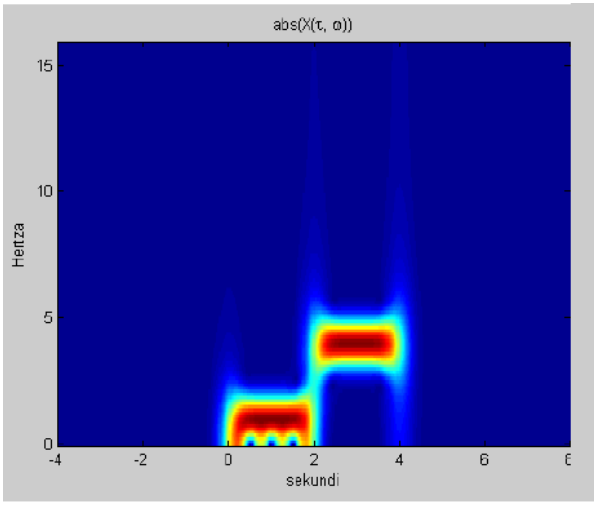
Kako bi se pokazao utjecaj širine vremenskog otvora na lokalizacijsku karakteristiku funkcije otvora u vremenskoj i frekvencijskoj domeni, dana su tri primjera s različitim širima vremenskog otvora.



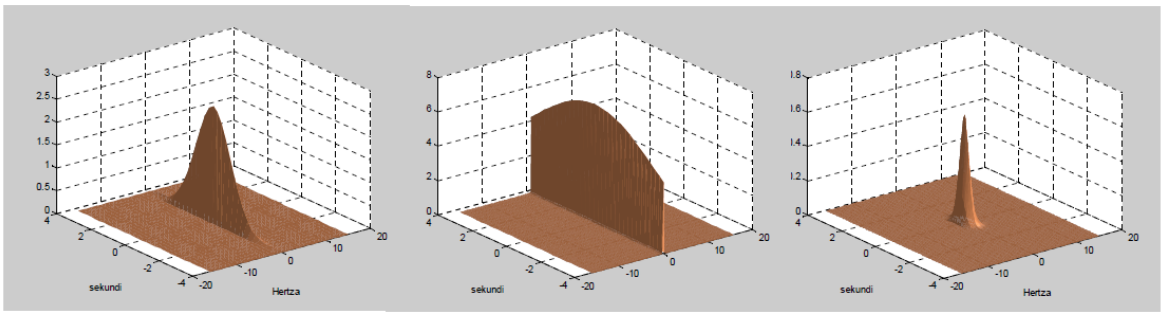

Slika 7 - Podjednako dobra lokalizacija u obje domene




Slika 8 - Transformacija s boljom lokalizacijom u frekvencijskoj domeni




Slika 9 - Transformacija s boljom lokalizacijom u vremenskoj domeni




Slika 10 - Tri različite širine vremenskog otvora

Za prvu širinu vremenskog otvora primjećujemo kako je postignut kompromis između frekvencijske i vremenske domene. Za obje domene razlučivost je prihvatljiva. Sljedeći vremenski otvor uopće ne izgleda popu Gaussove funkcije. Funkcija je preširoka i može se naslutiti kako će vremenska razlučivost biti jako mala, što se i vidi na drugoj slici. Dobra strana drugog primjera je velika frekvencijska razlučivost, tako da se detaljnije može saznati koje frekvencije su prisutne u spektru. Zadnji primjer pokazuje usku Gaussovu funkciju, koja će rezultirati odličnom preciznosti u vremenu. Širina u smjeru apscise je najmanja. Problem se, dakako, javlja u frekvencijskoj domeni gdje je razlučivost pregruba i neprihvatljiva. Nijedan od ova tri slučaja nije najbolji, već je na korisniku da izabere omjer koji će mu najviše odgovarati i zadovoljiti njegove potrebe.

# 2. Kontinuirana valićna transformacija

## 2.1. Što je CWT?

Poput Fourierove transformacije, kontinuirana valićna transformacija (eng. *continuous wavelet transform*; CWT) koristi skalarne umnoške kako bi izmjerila podudaranje signala i analizirajuće funkcije. U Fourierovoj transformaciji, analizirajuće funkcija je kompleksna eksponencijala $e^{j\omega t}$. Rezultat je funkcija jedne varijable $\omega$. U STFT-u, analizirajuća funkcija je opet kompleksna eksponencijala ali ovaj put ograničena vremenskim otvorom $w(t)e^{j\omega t}$. Rezultat je funkcija dvije varijable. Koeficijenti STFT-a, $F(\omega,\tau)$, predstavljaju podudaranje signala i sinusoide s kutnom frekvencijom $\omega$ u intervalu određene duljine centriranog na točki $T$ 5[1].

Kod CWT-a, analizirajuća funkcija je valić $\psi$. CWT uspoređuje signal sa pomaknutom i zbijenom ili rastegnutom verzijom valića. Rastezanje ili kompresija funkcije je kolektivno referencirano kao ekspanzija ili skaliranje i odgovara fizičkom pojmu skale. Uspoređujući signal s valićem s različitim skalama i pozicijama, dobiva se funkcija dvije varijable. Dvodimenzionalna reprezentacija jednodimenzionalnog signala je redundantna. Ako su vrijednosti valića kompleksne, onda je CWT funkcija skale i položaja s kompleksnim vrijednostima. Ako je pak signal realan CWT je funkcija s realnim vrijednostima. Za parametar skale , $a > 0$ i položaj $b$, CWT je:

$$C(a,b;f(t),\psi(t)) = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{a}}\psi^*\left(\frac{t-b}{a}\right)dt \qquad (5)$$

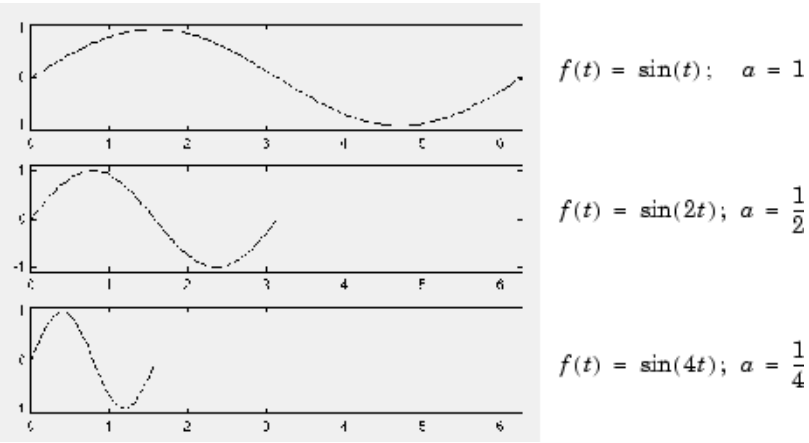
gdje * označava kompleksnu konjugaciju. Ne samo da vrijednosti skale i položaja utječu na CWT koeficijente, nego i odabir valića također utječe na njih. Konstantno mijenjajući vrijednosti skale, tj. parametar $a$, i poziciju, tj. parametar $b$, dolazimo do CWT koeficijenata $C(a,b)$. Naravno, zbog jednostavnosti se izostavlja ovisnost CWT koeficijenata o funkciji i analizirajućem valiću.

Postoji mnogo različitih prihvatljivih valića koji mogu biti upotrjebljeni u CWT-u. Možda se čini kako veliki izbor može biti zbunjujući, no to je zapravo snaga valićne analize. Ovisno o tome kakva obilježja signala se žele detektirati, slobodno se može izabrati valić koji olakšava detekciju tog obilježja. Na primjer, ako se želi detektirati nagli diskontinuitet u signalu, može se izabrati jedan valić. S druge strane, ako je interesantnije naći oscilacije s glatkim prijelazom, slobodno se može izabrati valić koji će više zadovoljiti kod takvog ponašanja signala. Neki od najpoznatijih tipova valićnih funkcija su Haarov valić, Morletov valić te sombrero.

### 2.1.1. Skala

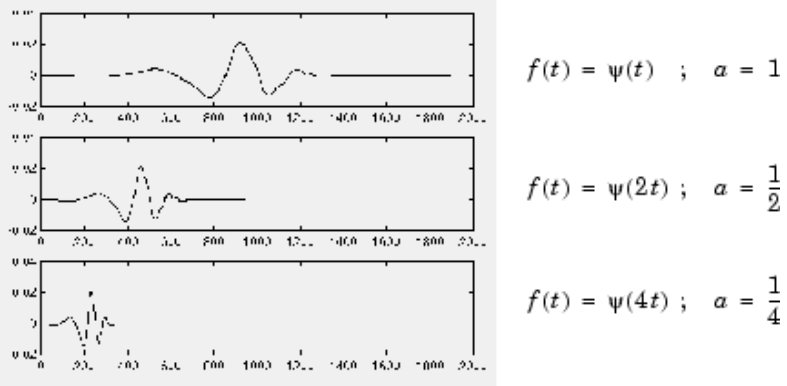
Slično frekvenciji, skala je još jedna korisna osobina slike i signala. Na primjer, podaci o temperaturi se mogu analizirati ovisno o različitim skalama. Može se promatrati mjesečna promjena, finija, dnevna promjena ili pak, grublja, godišnja promjena temperature. U nekim slučajevima zanimljivije su promjene koje se događaju kroz duže vrijeme tj. grublju skalu jer se na manjim skalama te promjene ne vide. Također, nekada se događa i suprotni primjer, gdje se određena obilježja vide samo na finijim skalama. Postoje i obilježja koje su nepromjenjiva ovisno o skali kao što je, na primjer, prepoznavanje ljudi. Neovisno o tome gleda li se veliki portret ili mala fotografija, prepoznat će se osoba koja je poznata.

Kako bi se objasnio pojam proširivanja i kompresije uvest će se faktor skale, obično nazivan $a$. Faktor skale je sam po sebi prirodan pozitivan broj, $a>0$. Na donjem primjeru pokazat će se kako se sinusni signal mijenja ovisno o faktoru skale.




Slika 11 - Sinusni signal za tri različite skale $a$

Ista stvar se događa i s valićima. Što je manji faktor skale, to je zbijeniji valić.




Slika 12 - Valić za tri različite skale $a$

Iz prethodnih slika, jasno je vidljivo kako je faktor skale $a$ u (inverznoj) vezi s kružnom frekvencijom $\omega$. Ta relacija između skale i frekvencije vrijedi općenito za sve signale. Osim što je reprezentacija pomoću vremena i skale novi način za gledanje signala, ujedno je i vrlo prirodan način za razmatranje mnogih podataka proizašlih iz mnogih prirodnih fenomena.

Kao primjer se može uzeti površina stijena koja se nalazi na obali mora. Oblik površine je rezultat dugogodišnjeg razaranja mora velikim te malim valovima. Ako se površina stijene zamisli kao jednodimenzionalni signal, onda je razumno zamisliti da je to signal koji se sastoji od komponenti različitih skala. Velika udubljenja u signalu biti će rezultat velike skale tj. utjecaja velikih valova, dok će finiji dio predstavljati utjecaj manjih valova.

Slika 13 - Stijena i njezin površinski presjek nastao većim i manjim valovima

Očito je da postoji veza između frekvencije i skale. Kao što je već ranije pokazano, veće skale rezultiraju izduženijim valićima. Što je više valić raširen, duži je dio signala koji se uspoređuje s valićem pa su stoga i obilježja signala, mjerena valićnim koeficijentima, grublja. Dakle, možemo zaključiti da vrijede sljedeće relacije:

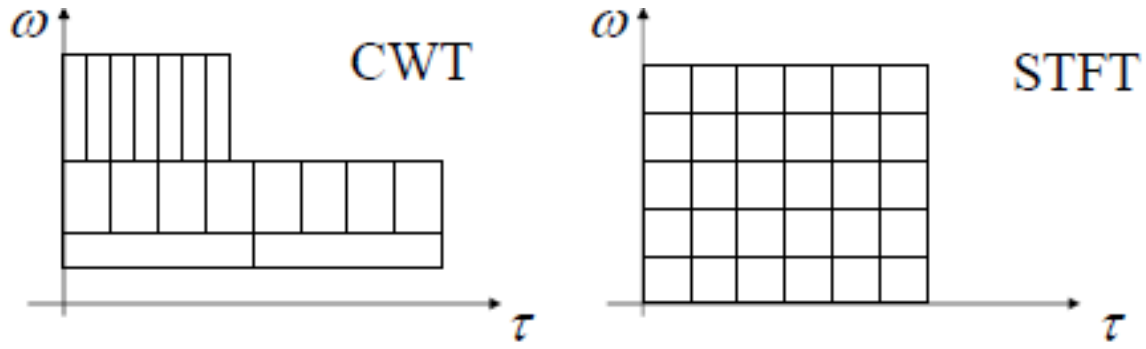
1) Mala skala $a$ => komprimirani valić => brzo promjenjujući detalji => velika frekvencija $\omega$.
2) Velika skala $a$ => izduženi valić => sporo mijenjajuća, grublja obilježja => mala frekvencija $\omega$.

Iako postoji općenita veza između skala i frekvencija, nema precizno definiranog odnosa. Obično korisnici upoznati s Fourierovom analizom, žele definirati preslikavanje između valića za danu skalu sa specifičnim periodom uzrokovanja te frekvencijom u Hz. To se može raditi samo u općenitom slučaju, a više o tome može se naći u sljedećim poglavljima.

### 2.1.2. CWT kao transformacija vremenskim otvorom

Kratkotrajna Fourierova transformacija, STFT, je definirana tako da se uzimaju vremenski otvori signala, kako bi se stvorila lokalna frekvencijska analiza. Nedostatak tome je konstantna veličina vremenskog otvora. Kao što je objašnjeno ranije, korisnik mora birati svoju veličinu vremenskog otvora. Ukoliko odabere duži vremenski otvor, popravit će frekvencijsku rezoluciju ali na štetu vremenske. Isto tako, ukoliko odabere kratki vremenski otvor popravit će vremensku rezoluciju ali će pokvariti frekvencijsku.

Valićna transformacija reprezentira novi korak u razmišljanju: vremenski otvor gdje će njegova veličina biti promjenjiva tj. varijabla. Kod ovakvog slučaja, dobiva se situacija gdje korisnik sam bira gdje će mu trebati bolja vremenska a gdje bolja frekvencijska rezolucija.




Slika 14 - Rezolucije kod CWT-a i STFT-a

### 2.1.3. CWT kao tehnika filtriranja

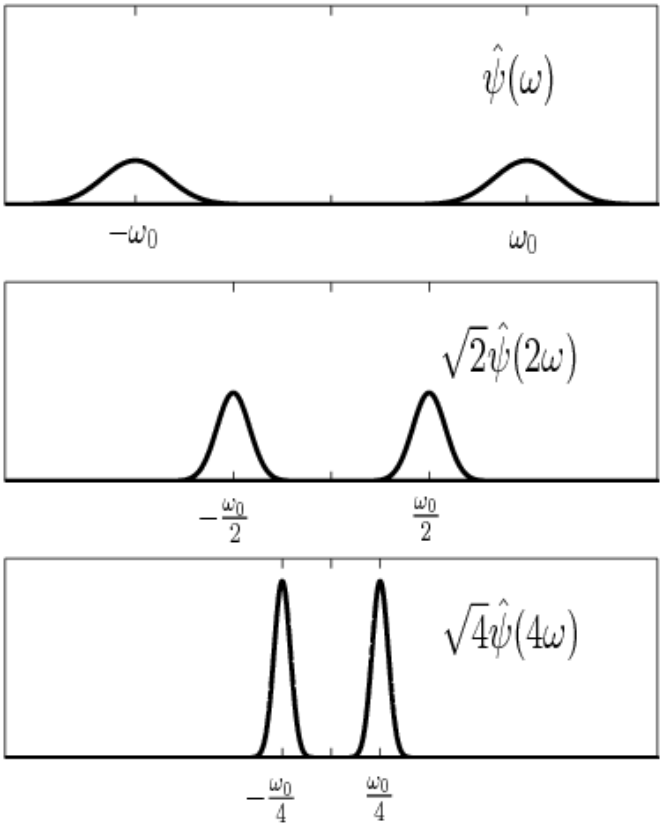
Kontinuirana valićna transformacija računa skalarni umnožak signala (funkcija $f(t)$) sa proširenom verzijom analizirajućeg valića, $\psi(t)$. Kao što je već objašnjeno, definicija CWT-a je :

$$C(a,b;f(t),\psi(t)) = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{a}}\psi^*\left(\frac{t-b}{a}\right)dt \qquad (6)$$

Isto tako, CWT se može interpretirati kao frekvencijsko filtriranje signala, ako preoblikujemo definiciju CWT-a u inverznu Fourierovu transformaciju.

$$C(a,b;f(t),\psi(t)) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\hat{f}(\omega)\sqrt{a}(\hat{\psi}(a\omega))^* e^{j\omega b}d\omega \qquad (7)$$

Gdje su $\hat{f}(\omega)$ i $\hat{\psi}(\omega)$ Fourierove transformacije signala i valića. Iz prethodnih jednadžbi vidimo da širenje valića u vremenu uzrokuje, njegovo smanjenje u frekvencijskoj domeni. Osim smanjenja u frekvencijskoj domeni, centar frekvencije se približava nižim frekvencijama. Sljedeća slika pokazat će promijene valića ako faktor skale mijenja vrijednost u 1,2 i 4.




Slika 15 – Promjena valića za faktor skale 1, 2 i 4

Ovdje se CWT prikazuje kao pojasno-propusna filtracija ulaznog signala. CWT koeficijenti na nižim skalama predstavljaju energiju ulaznog signala na višim frekvencijama, dok CWT koeficijenti na višim skalama pokazuju energiju ulaznog signala na nižim frekvencijama. Doduše za razliku od Fourierovog pojasno-propusnog filtriranja, širina pojasa za propuštanje kod CWT-a je obrnuto proporcionalan skali. Širina CWT filtara smanjuje se s povećanjem skale. Ova činjenica slijedi iz veze između vremenske i frekvencijske podrške signalu: što je šira podrška signalu u vremenu, to je uža u frekvenciji. Obrnuti slučaj također vrijedi.

U valićnoj transformaciji, skala, ili operacija proširenja je definirana kako bi očuvala energiju. Kako bi se očuvala energija dok se smanjuje frekvencijski pojas, potrebno je povećati vrh nivoa energije. Faktor kvalitete (eng. *Quality factor* ili Q-faktor) filtra je omjer njegovog vrha energije i frekvencijskog pojasa. Zbog činjenice da se kod smanjenja ili povećavanja frekvencijskog pojasa valića povećava ili smanjuje vrh njegove energije, valići su obično spominjani kao filtri s konstantnim Q-faktorom.

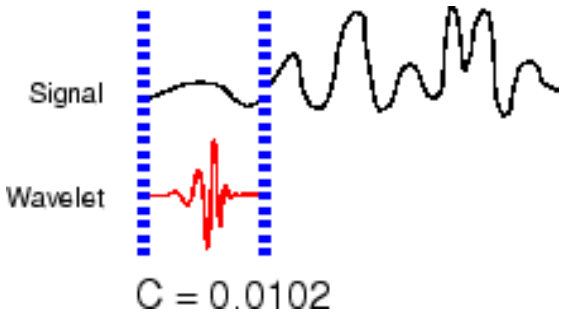
## 2.2. Implementacija CWT-a u Matlabu

Računanje kontinuirane valićne transformacije možemo podijeliti u 5 jednostavnih koraka[2]:

1) Usporediti valić s dijelom na početku originalnog signala

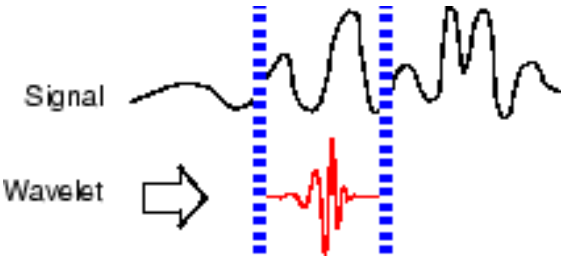
2) Izračunati broj $C$, koji predstavlja sličnost između valića i trenutnog dijela signala. Što je veći broj $C$ (u apsolutnoj vrijednosti) veća je podudarnost. To slijedi iz činjenice da su CWT koeficijenti računani pomoću skalarnog umnoška. Ako su energija signala i energija valića jednaki jedan, $C$ se može interpretirati kao koeficijent korelacije. No, općenito energija signala nije jedan i CWT koeficijenti nisu direktno povezani sa korelacijskim koeficijentima.

   Kao što je bilo opisano u definiciji kontinuirane valićne transformacije, CWT koeficijenti eksplicitno ovise o analizirajućem valiću. Stoga su CWT koeficijenti različiti za isti signal, ako koristimo različite valiće.



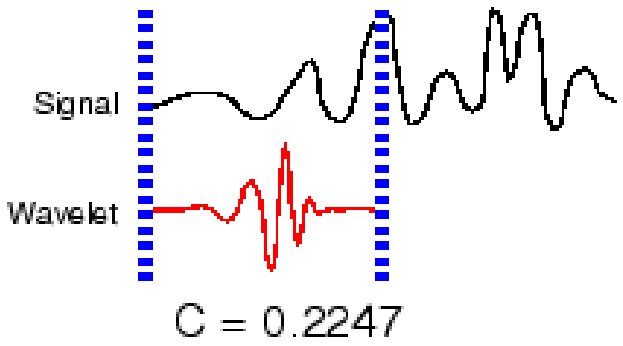

Slika 16 – Usporedba valića i prvog segmenta signala

3) Pomaknuti valić udesno i ponoviti korake 1 i 2 dok se ne pokrije cijeli signal.




Slika 17 - Usporedba valića i drugog segmenta signala

4) Raširiti (istegnuti) valić i ponoviti korake 1, 2 i 3.




Slika 18 - Usporedba proširenog valića i signala

5) Ponoviti korake od 1 do 4 za sve skale.

### 2.2.1. Interpretacija CWT koeficijenata

Zbog činjenice da je CWT redundantna transformacija i zbog toga što CWT koeficijenti ovise o valiću, ponekad je teško interpretirati rezultate transformacije.

Kako bi se shvatio način interpretacije CWT koeficijenata u Matlabu, najbolje je početi sa jednostavnim signalom za analizu, te analizirajućim valićem s jednostavnom strukturom.

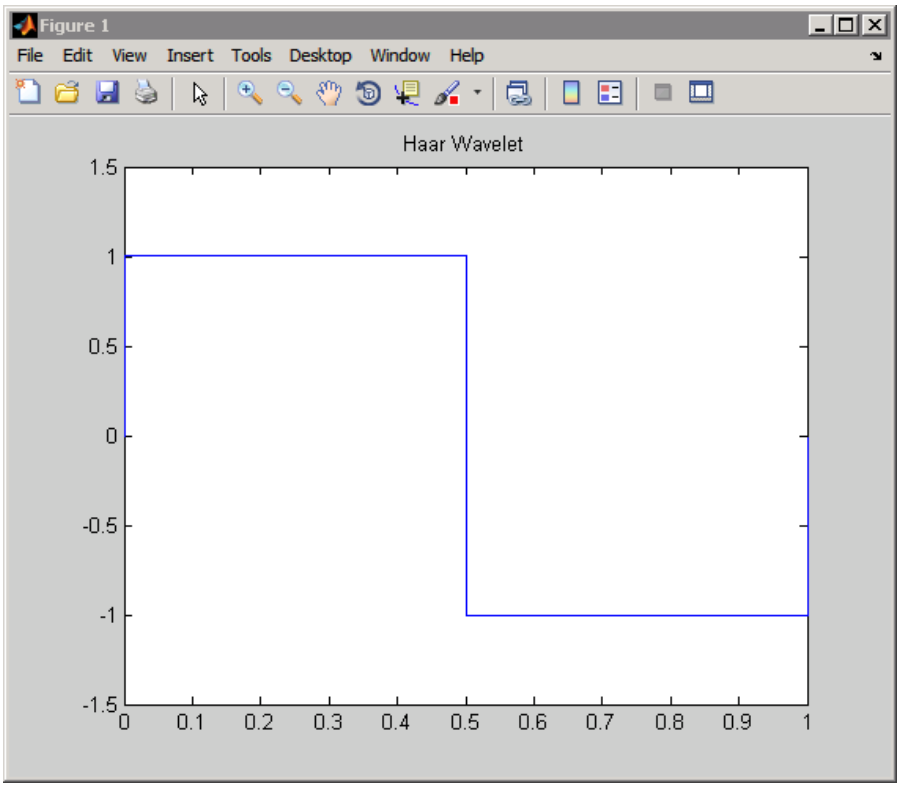
Jedno od obilježja koje valići dobro detektiraju su diskontinuiteti ili singularitet. Nagle promjene u signalu rezultiraju CWT koeficijentima s visokim apsolutnim vrijednostima.

Za primjer će se uzeti signal koji ima pomaknuti impuls koji se pojavljuje na točki 500.

```
x = zeros(1000,1);
x(500) = 1;
```

Za valić će se izabrati Haarov valić.

```
[~,psi,xval] = wavefun('haar',10);
plot(xval,psi); axis([0 1 -1.5 1.5]);
title('Haar Wavelet');
```




Slika 19 - Haarov valić

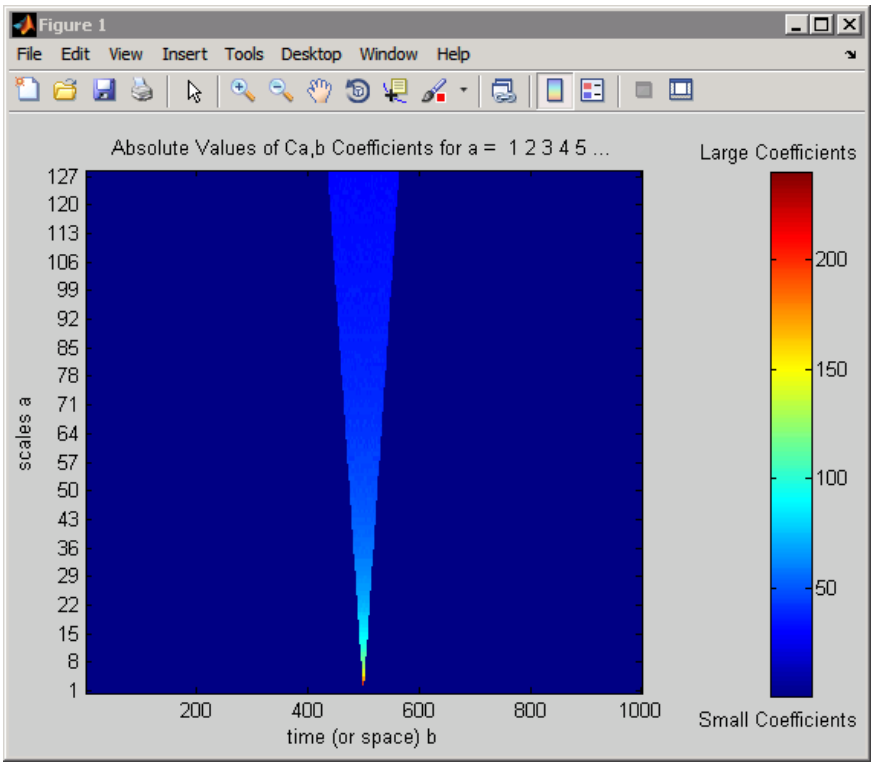
Za izračunavanje CWT-a pomoću Haarovog valića za skale od 1 do 128 treba se upisati sljedeće:

```
CWTcoeffs = cwt(x,1:128,'haar');
```

*CWTcoeffs* je matrica 128 sa 1000. Svaki redak matrice sadrži CWT koeficijente za jednu skalu. Broj redova je 128 jer je i broj skala od 1 do 128. Broj stupaca u matrici ovisi o duljini ulaznog signala.

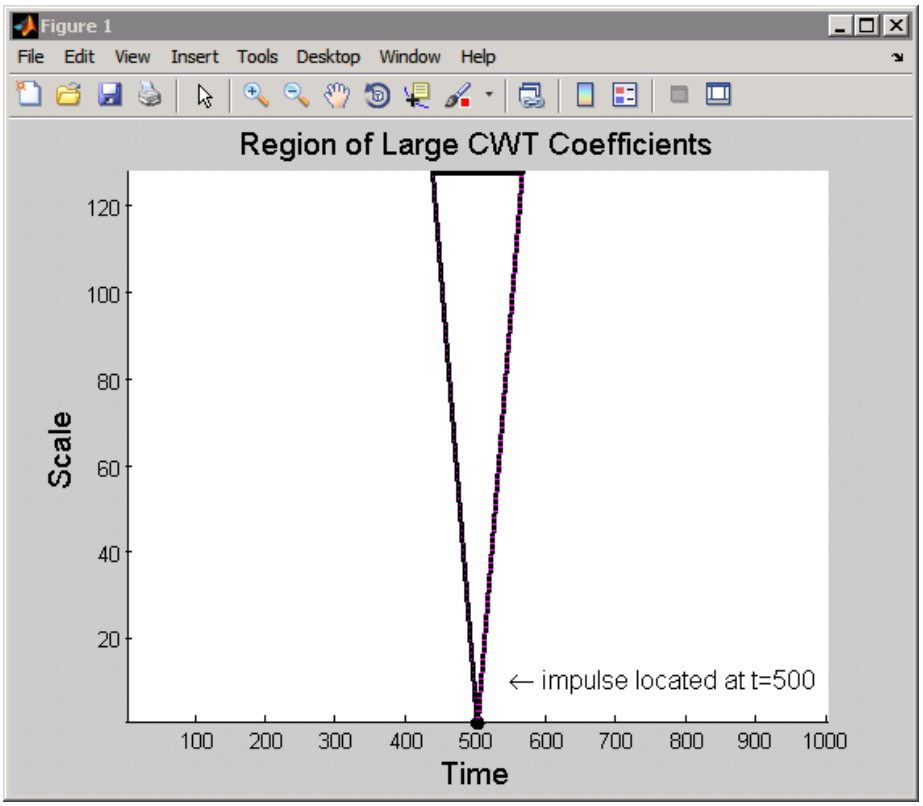
Kako je CWT jednodimenzionalnog signala funkcija skale i vremena (tj. pozicije), za skiciranje će biti potrebno pozicionirati vrijeme na x-os, skala će biti postavljena na y-osi dok će se magnituda tj. veličina CWT koeficijenata ogledati u bojama na ravnini vremena i skale. Što je žarkija (toplija) boja to je apsolutna vrijednost CWT koeficijenata veća.

```
cwt(x,1:128,'haar','plot');
colormap jet; colorbar;
```




Slika 20 – Apsolutne vrijednosti CWT koeficijenata

Ako se prouči CWT pomaknutog impulsa, može se vidjeti da je grupa velikih CWT koeficijenata koncentrirana u maloj regiji. Nalazi se oko točke 500 smještena na malim skalama. Što se skala povećava grupa velikih CWT koeficijenata postaje šira (koeficijenti se sukladno smanjuju), ali je i dalje centrirana oko točke 500. Granica ovog polja (regije) ima oblik kao na slici ispod.




Slika 21 - Konus utjecaja točke $t = 500$

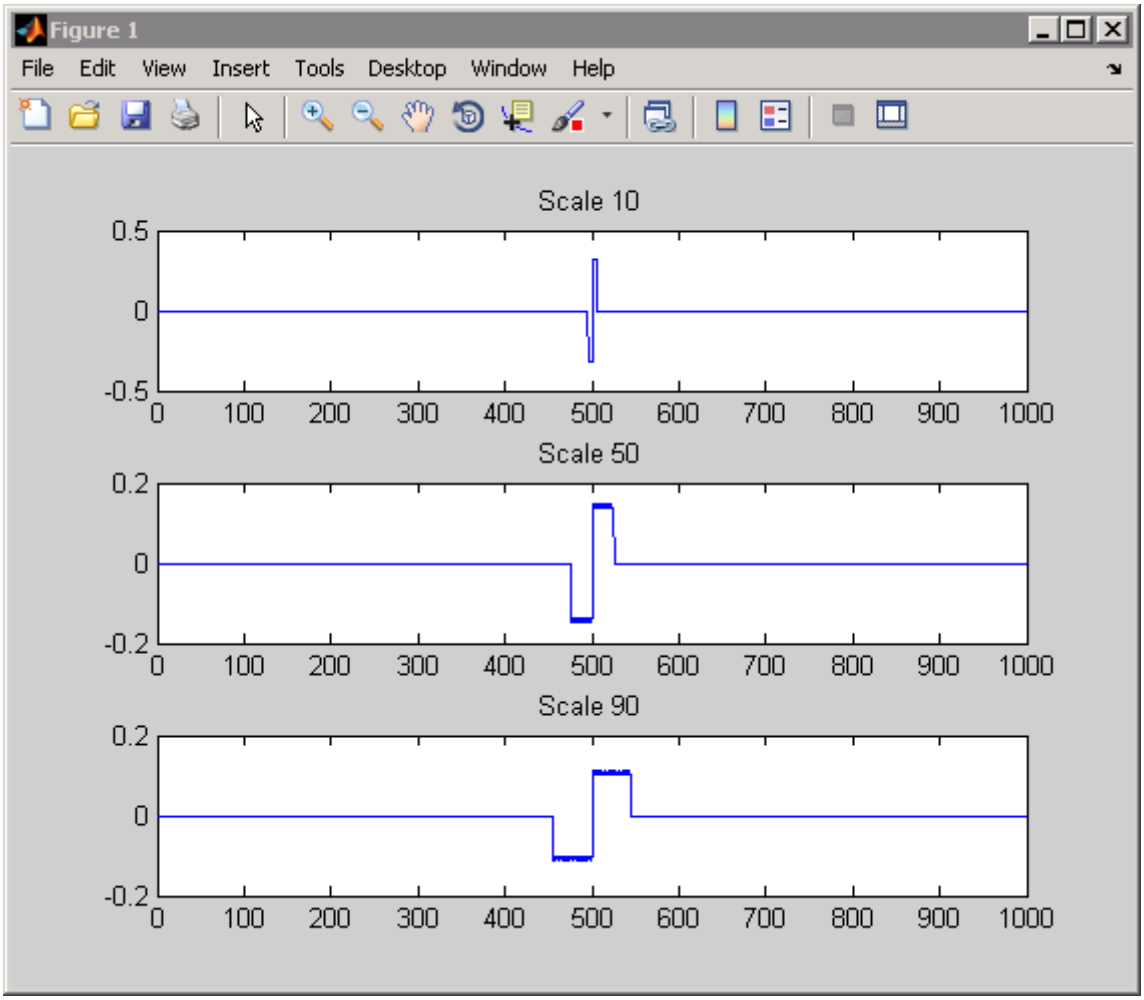
To polje se često označava kao konus utjecaja točke $t = 500$ za Haarov valić. Za danu točku, konus utjecaja pokazuje koji CWT koeficijenti su pod utjecajem signala u toj točki.

Kako bi se bolje shvatio konus utjecaja, pretpostavit će se da je valić podržan na intervalu $[-C, C]$. Pomičući valić za $b$ i skalirajući ga za $a$, rezultira valićem podržanom na intervalu $[-Ca + b, Ca + b]$. Za jednostavan primjer pomaknutog impulsa, $\delta(t - \tau)$, CWT koeficijenti su jedino na intervalu oko $\tau$ različiti od nule te jednaki podršci valića na svakoj skali. To se može vidjeti razmatrajući formalni izraz CWT-a za pomaknuti impuls.

$$C(a, b; \delta(t-\tau), \psi(t)) = \int_{-\infty}^{\infty} \delta(t-\tau) \frac{1}{\sqrt{a}} \psi^*\left(\frac{t-b}{a}\right) dt = \frac{1}{\sqrt{a}} \psi^*\left(\frac{\tau-b}{a}\right) \tag{8}$$

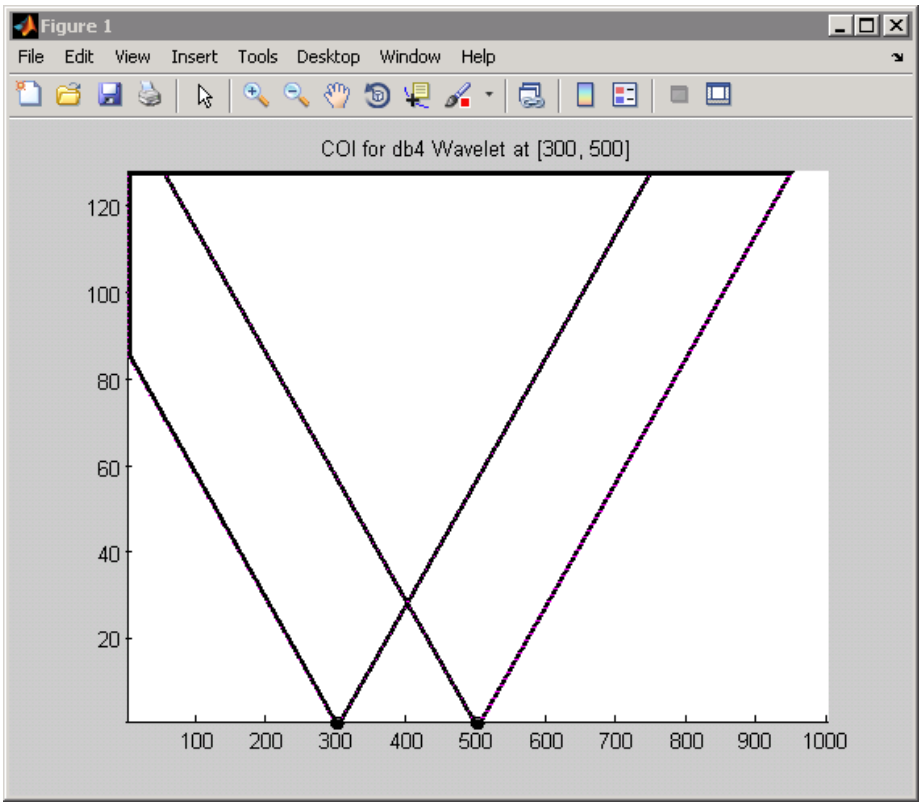
Za signal jednak impulsu, CWT koeficijenti su jednaki konjugiranoj, vremenski obrnutoj i skaliranoj valićnoj funkciji, kao funkcija pomaknutog parametra $b$. To se može vidjeti iscrtavanjem CWT koeficijenata za nekoliko skala.

```
subplot(311)
plot(CWTcoeffs(10,:)); title('Scale 10');
subplot(312)
plot(CWTcoeffs(50,:)); title('Scale 50');
subplot(313)
plot(CWTcoeffs(90,:)); title('Scale 90');
```




Slika 22 – CWT koeficijenti impulsa za različite skale

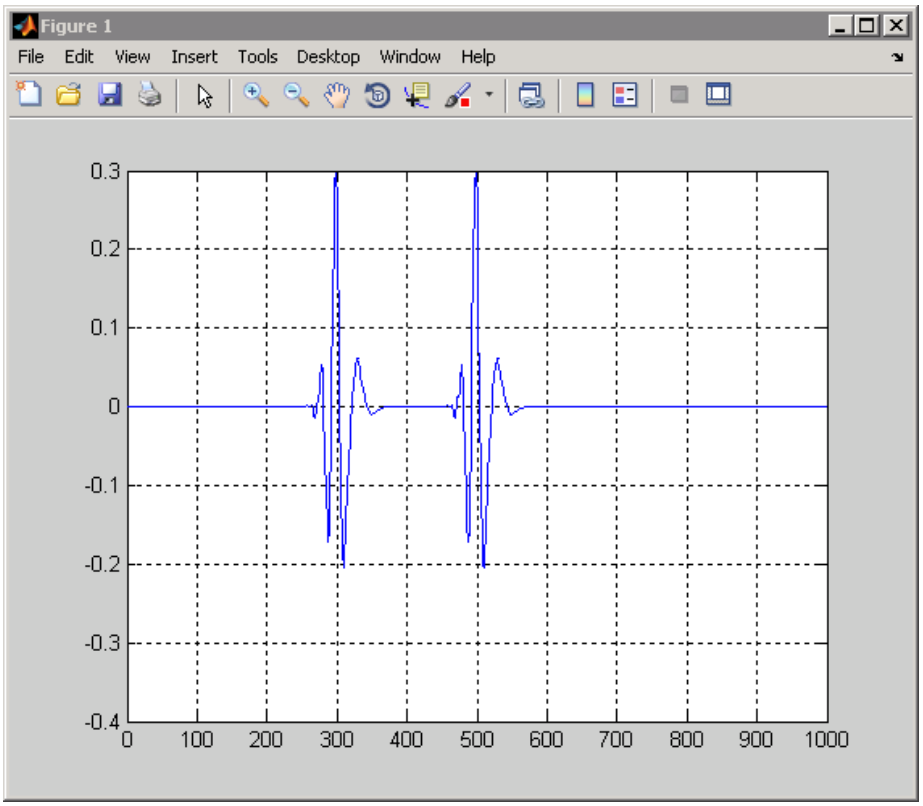
Sljedeći primjer pokazat će superpoziciju dva pomaknuta impulsa, delta($t$-300)+delta($t$-500). U ovom slučaju koristit će se Daubechies'ov *extremal phase* valić s 4 nultočke, db4. Sljedeća slika prikazuje konus utjecaja za točke 300 i 500 upotrjebljujući db4 valić.




Slika 23 - Konus utjecaja točaka $t = 300$ i $t = 500$

Promatranjem točke 400 za skalu 20 može se vidjeti da nijedan konus ne prelazi točku 400. Za očekivati je, dakle, da će CWT koeficijenti biti nula za tu točku i skalu. Signal je različit od nule samo za dvije točke, a to su 300 i 500 te nijedan konus utjecaja za te dvije vrijednosti uključuje točku 400 za skalu 20. Ovo se može potvrditi i jednostavnim kodom:

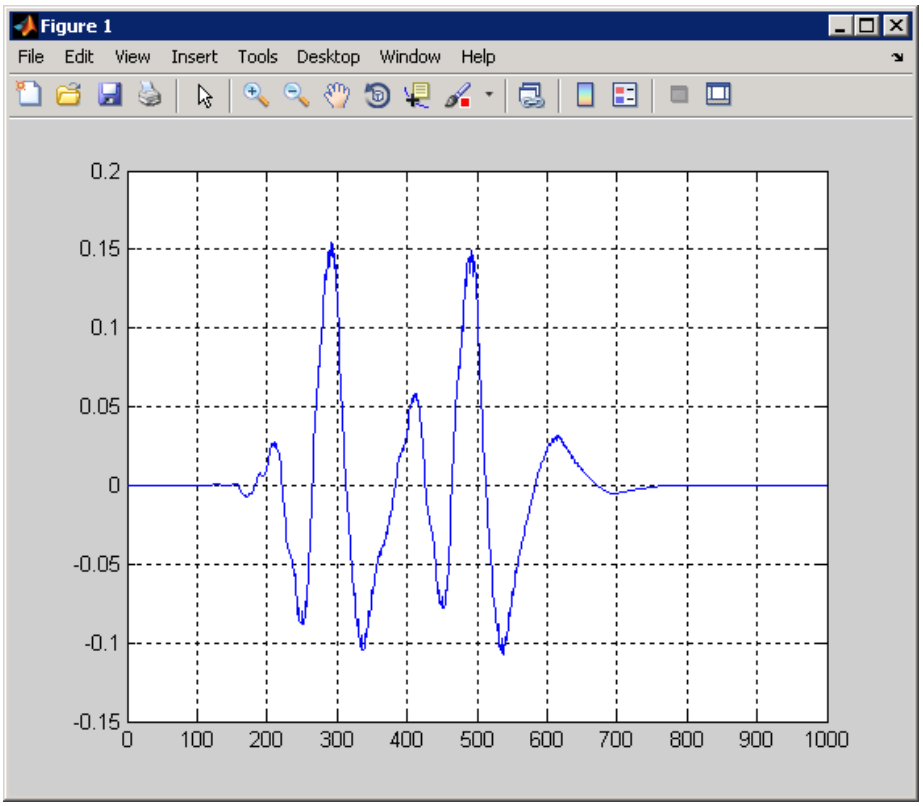
```
x = zeros(1000,1);
x([300 500]) = 1;
CWTcoeffs = cwt(x,1:128,'db4');
plot(CWTcoeffs(20,:)); grid on;
```




Slika 24 - CWT koeficijenti za skalu 20

Situacija sa višim skalama je drugačija. Konusi utjecaja za obje točke (300 i 500), na skali 80, sadrže i točku 400. Iako je signal nula u točki 400, dobiva se CWT koeficijent koji je različit od nule. Razlog tomu je taj što je područje djelovanja valića postalo dovoljno veliko za tu skalu (proširenje valića u vremenu) tako da se za CWT koeficijente upotrebljavaju i točke signala koje su i do 100 točki udaljene od promatrane točke. Sljedeći primjer pojašnjava situaciju:

```
plot(CWTcoeffs(80,:));
grid on;
```

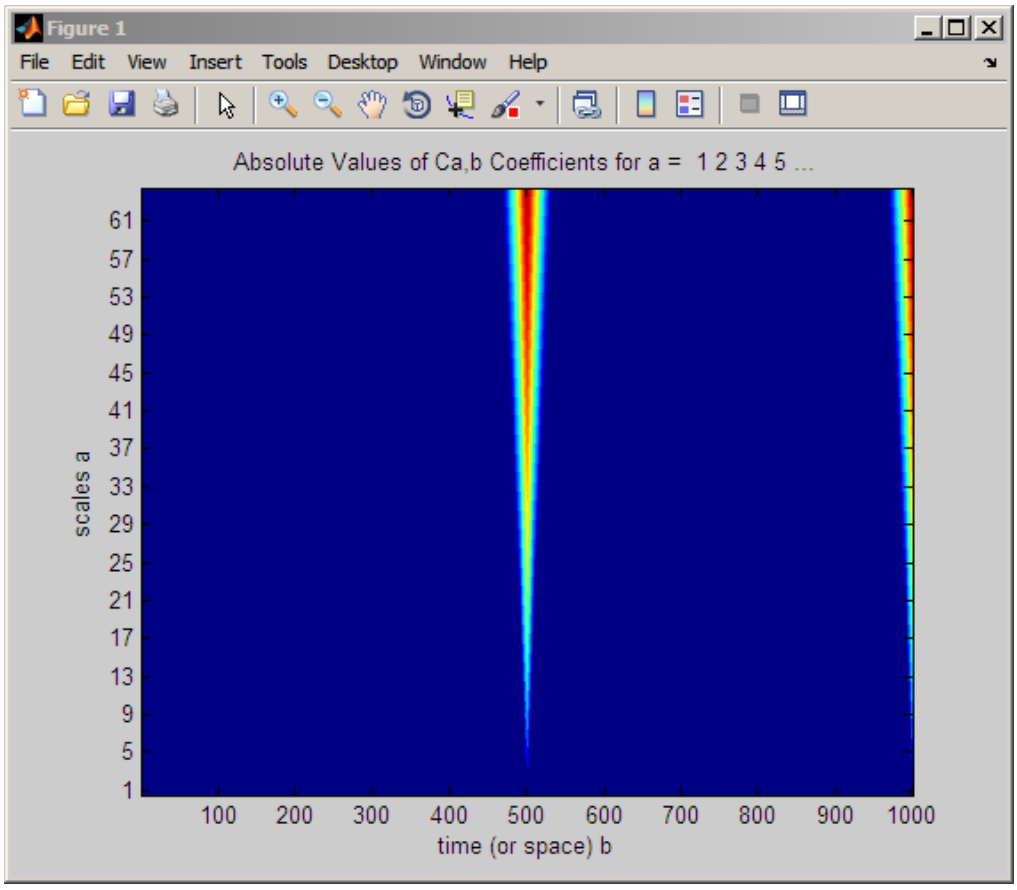



Slika 25 - CWT koeficijenti za skalu 80

U prethodnim primjerima vidi se da CWT koeficijenti postaju veći u blizini nagle promjene u signalu. Ta sposobnost detektiranja diskontinuiteta je snaga valićne transformacije. Primjeri također pokazuju kako CWT koeficijenti najbolje lokaliziraju diskontinuitet na malim skalama.

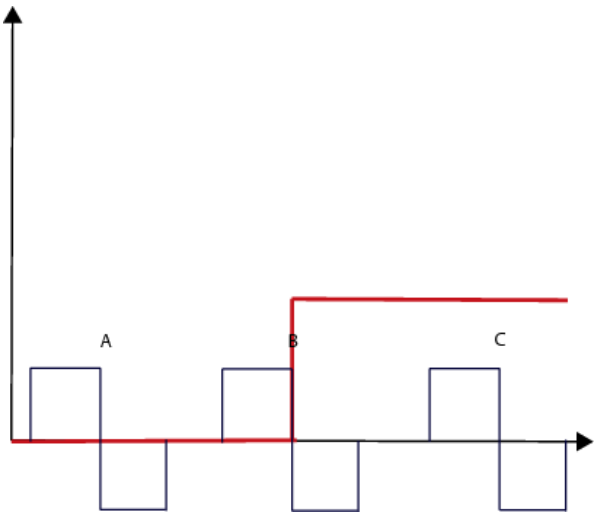
Kako bi se demonstriralo zašto je valićna transformacija toliko vješta u detekciji naglih promjena u signalu, koristit će se odskočni (eng. *step*) signal.

```
x = [zeros(500,1); ones(500,1)];
CWTcoeffs = cwt(x,1:64,'haar','plot'); colormap jet;
```




Slika 26 - CWT transformacija odskočnog (*step*) signala

Slično kao i kod pomaknutog impulsa, nagla promjena u pomaknutoj odskočnoj funkciji rezultira velikim CWT koeficijentima kod diskontinuiteta. Sljedeća slika objašnjava zašto.



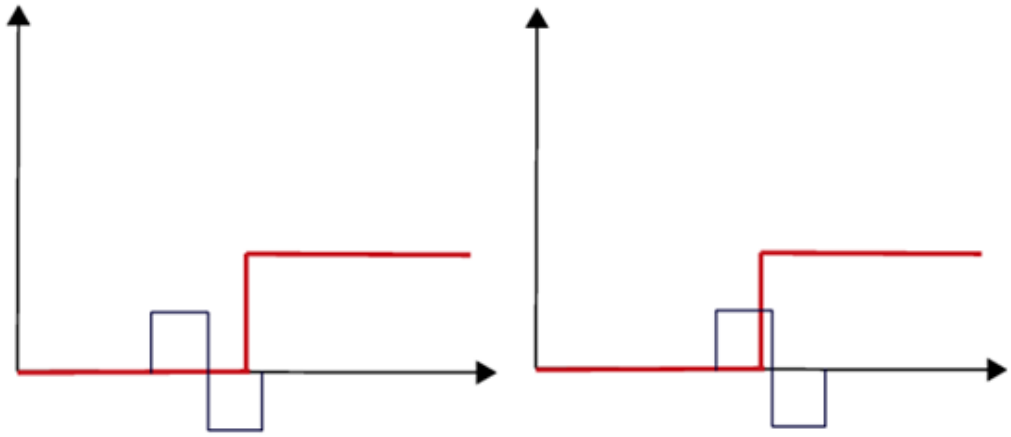

Slika 27 – Usporedba valića s odskočnim (*step*) signalom

Na prethodnoj slici, crveno je označena pomaknuta odskočna (*step*) funkcija. Crne funkcije označene s A,B i C prikazuju Haarov valić za istu skalu samo na drugim pozicijama. Može se vidjeti da su CWT koeficijenti oko pozicije A jednaki nuli. Signal je nula u toj okolini i stoga je valićna transformacija isto tako nula jer se svaki valić integrira u nulu.

Valić centriran oko pozicije B preklapa se s oba dijela odskočne funkcije. Negativni dio Haarovog valića preklapa se s područjem funkcije koja je jednaka jedinici. CWT koeficijenti su negativni jer je umnožak Haarovog valića i odskočne funkcije negativna konstanta. Integriranje preko tog područja daje negativni broj.

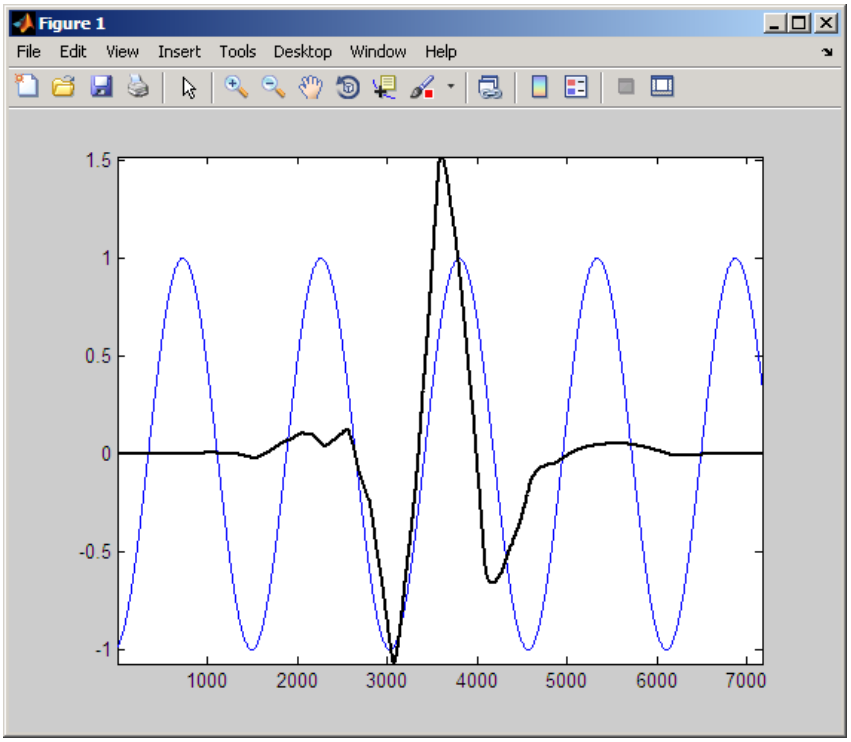
Kod valića centriranog oko točke C, CWT koeficijenti su isto nula. Odskočna funkcija tamo je jednaka jedan. Umnožak funkcije i valića jednak je samom valiću. Integracija bilo kojeg valića daje nulu, a to svojstvo valića zove se nulti moment.

Treba primijetiti da je kod pozicije B, Haarov valić pomaknut u nenegativni dio funkcije sa ½ svoje širine. Čim se valić pomakne svojim negativnim dijelom u pozitivni dio odskočne funkcije CWT koeficijenti postaju različiti od nule. CWT koeficijenti tako rastu (apsolutna vrijednost) i dosežu maksimum upravo na poziciji B. Cijeli negativni dio valića se u tom trenutku preklapa s jedinicom odskočne funkcije dok se cijeli pozitivni dio valića preklapa s nulom. Čim se i najmanji dio pozitivnog dijela valića pomakne pod jedinicu odskočne funkcije apsolutna vrijednost CWT koeficijenta počinje se smanjivati. Razlog tome je što se sad negativni dio integrala povećao za onaj pozitivni dio valića, te tako smanjuje ukupnu apsolutnu vrijednost. Obje ove situacije prikazane su slikom ispod.



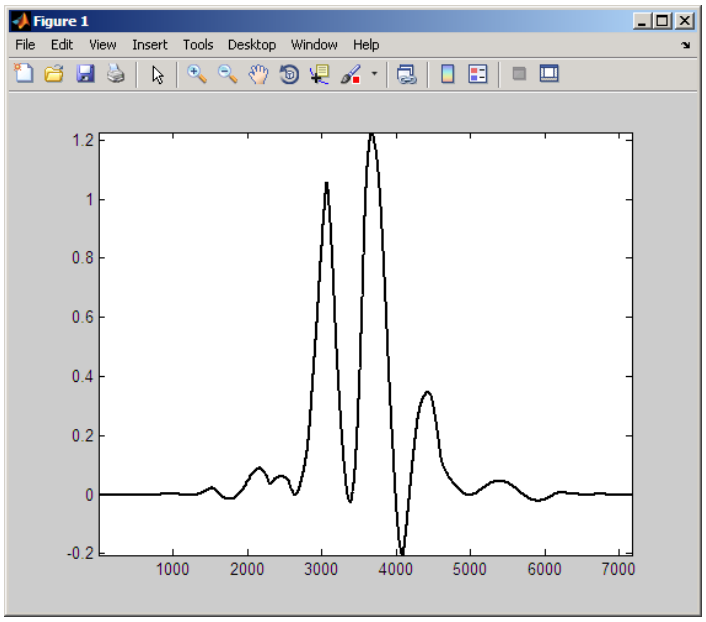
Slika 28 - Pomicanje valića po signalu

Objašnjeno je kako se detektiraju diskontinuiteti u signalu. Potrebno je vidjeti drugu stranu tj. kako se pomoću CWT-a reprezentiraju glatki signali. Kao primjer glatkog signala uzet će se sinusoida jer je česta pojava u prirodi. Za ovu primjenu koristit će se sym4 valić koji će biti superponiran sinusnom signalu.



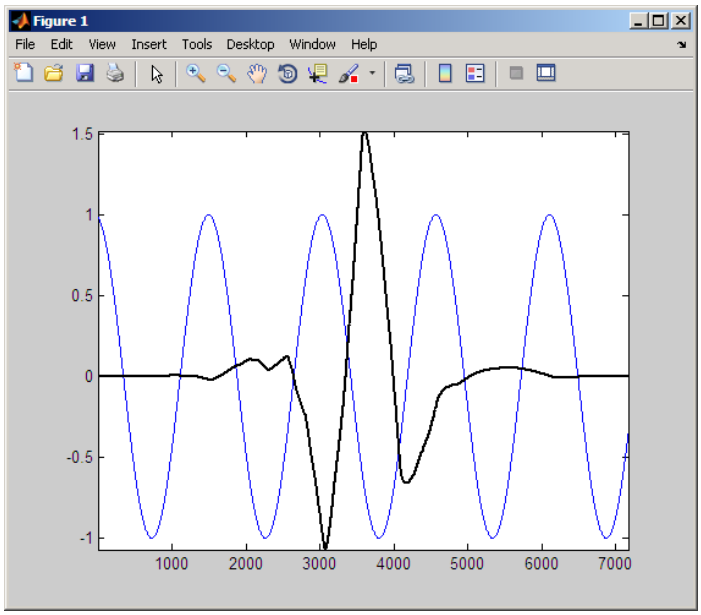

Slika 29 - Sinusni signal i valić

Potrebno je prisjetiti se kako su CWT koeficijenti dobiveni računanjem integrala umnoška signala sa pomaknutim i skaliranim analizirajućim valićem. Sljedeća slika pokazuje umnožak signala sa analizirajućim valićem iz prethodne slike.



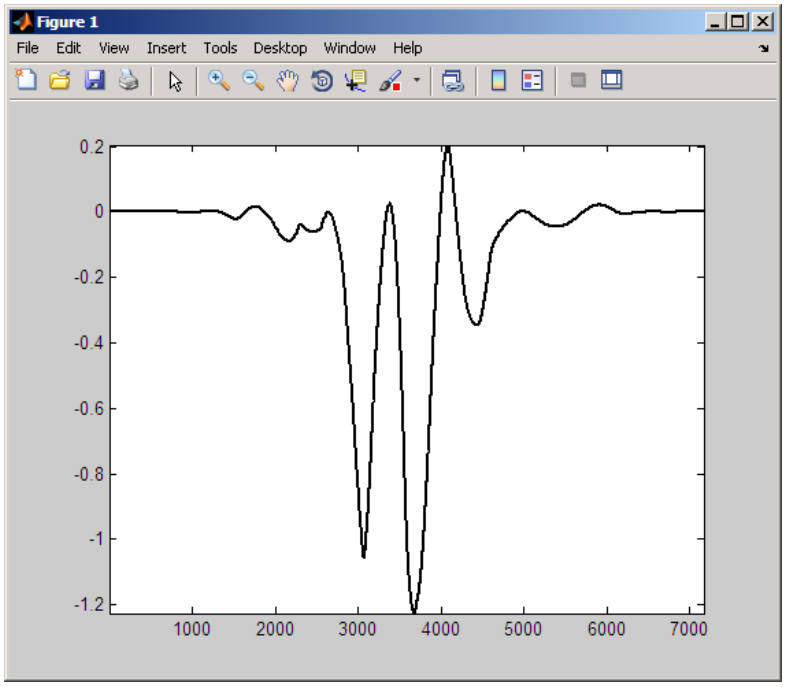

Slika 30 - Umnožak signala sa analizirajućim valićem

Primjećuje se da integracijom ovog signala (umnoška) nastaju pozitivni CWT koeficijenti. Ovo se događa jer oscilacije u valiću približno odgovaraju periodu sinusnog vala. Valić je u fazi sa sinusnim signalom. Negativni dijelovi valića se poklapaju sa negativnim dijelom sinusnog vala, dok se pozitivni dijelovi valića poklapaju sa pozitivnim dijelom sinusa. Rezultat je pozitivan CWT koeficijent.



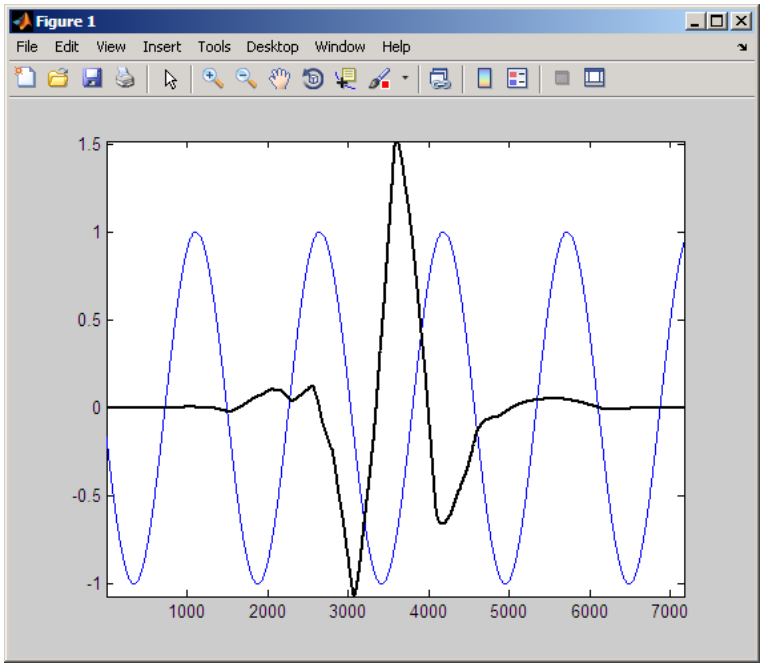

Slika 31 - Valić pomaknut za pola perioda

Prethodna slika pokazuje situaciju kada se valić pomakne za pola perioda. Može se primijetiti da se situacija samo obrnula. Ovaj put se negativni dio valića poklapa sa pozitivnim dijelom sinusa, dok se pozitivni dio valića poklapa sa negativnim dijelom sinusnog vala. Za pretpostaviti je da će se dobiti negativan umnožak koji će po apsolutnoj vrijednosti biti jednak prethodnom slučaju. To se može provjeriti i sljedećom slikom.



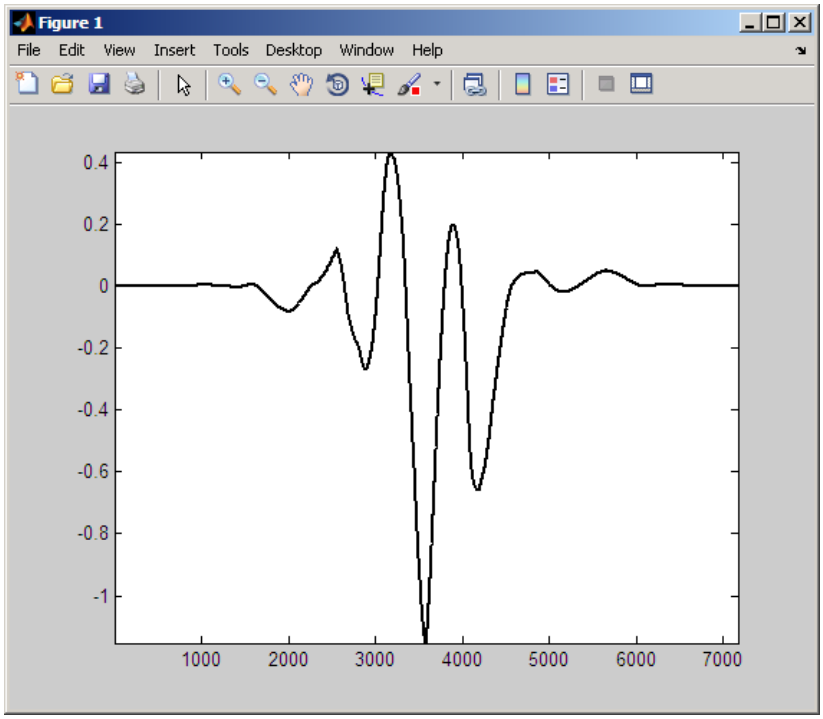

Slika 32 - Umnožak pomaknutog valića i signala

Vrijedi pogledati i treći primjer gdje se valić pomakne samo za ¼ perioda.




Slika 33 - Valić pomaknut za četvrtinu perioda

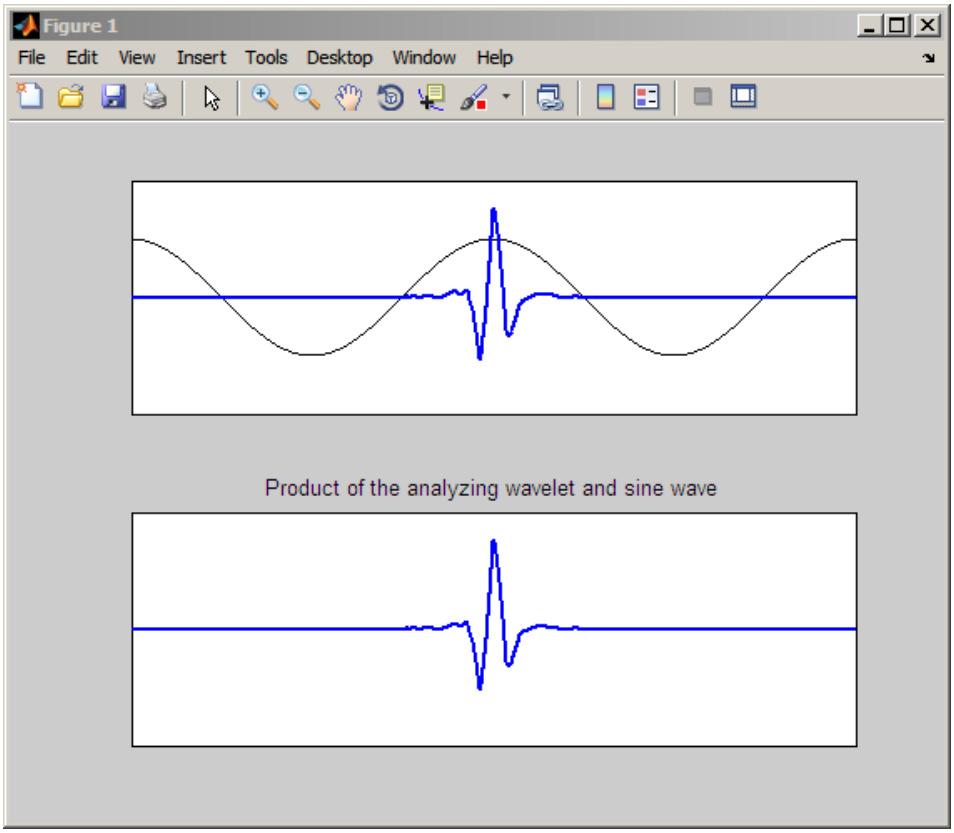
Produkt ova dva signala prikazan je na sljedećoj slici.




Slika 34 - Umnožak pomaknutog valića i signala

Integriranje ovog umnoška stvara CWT koeficijent koji je mnogo manji u apsolutnoj vrijednosti od oba prijašnja primjera. Ovo se događa zbog toga što je negativni dio valića približno poravnat s pozitivnim dijelom sinusnog vala. Također pozitivni dio valića je isto tako približno poravnat s pozitivnim dijelom sinusnog signala. Rezultirajući umnožak izgleda više kao valić nego umnožak dvaju signala. Kada bi umnožak izgledao isto kao i valić integral bi bio nula zbog ranije opisanog svojstva.

Kod skala gdje je oscilacija valića na jako velikoj ili na jako maloj skali, u odnosu na period sinusnog vala, CWT koeficijenti koji se dobiju su jednaki nuli. Sljedeća slika ilustrirat će slučaj kada su oscilacije valića na mnogo manjoj skali od sinusoide.

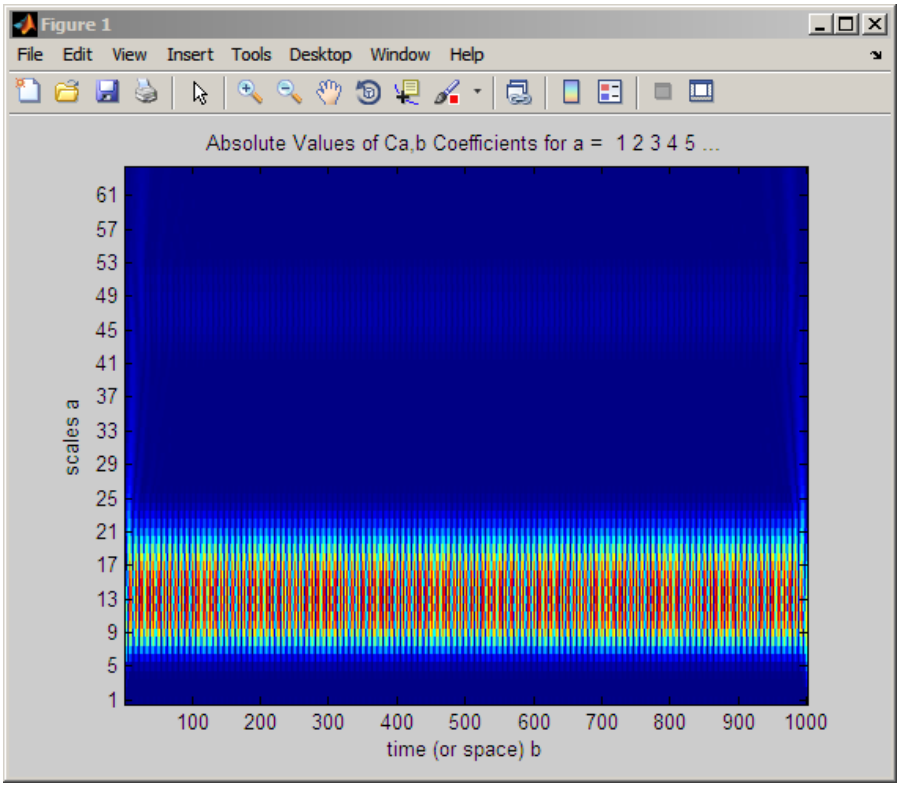



Slika 35 – Oscilacije valića na puno manjoj skali od sinusnih

Može se vidjeti kako je umnožak analizirajućeg valića i sinusnog signala gotovo identičan valiću. Integracijom tog umnoška dobiva se CWT koeficijent, koji je po apsolutnoj vrijednosti blizak nuli.

Sljedeći primjer prikazuje CWT koeficijente sinusnog signala frekvencije 60 Hz, koji je uspoređen sa sym8 valićem.

```
t = linspace(0,1,1000);
x = cos(2*pi*60*t);
CWTcoeffs = cwt(x,1:64,'sym8','plot'); colormap jet;
```




Slika 36 – CWT sinusoide od 60 Hz

Vidi se kako su najveći koeficijenti između skale 9 i 21. Pseudo-frekvencije mogu se naći koristeći naredbu:

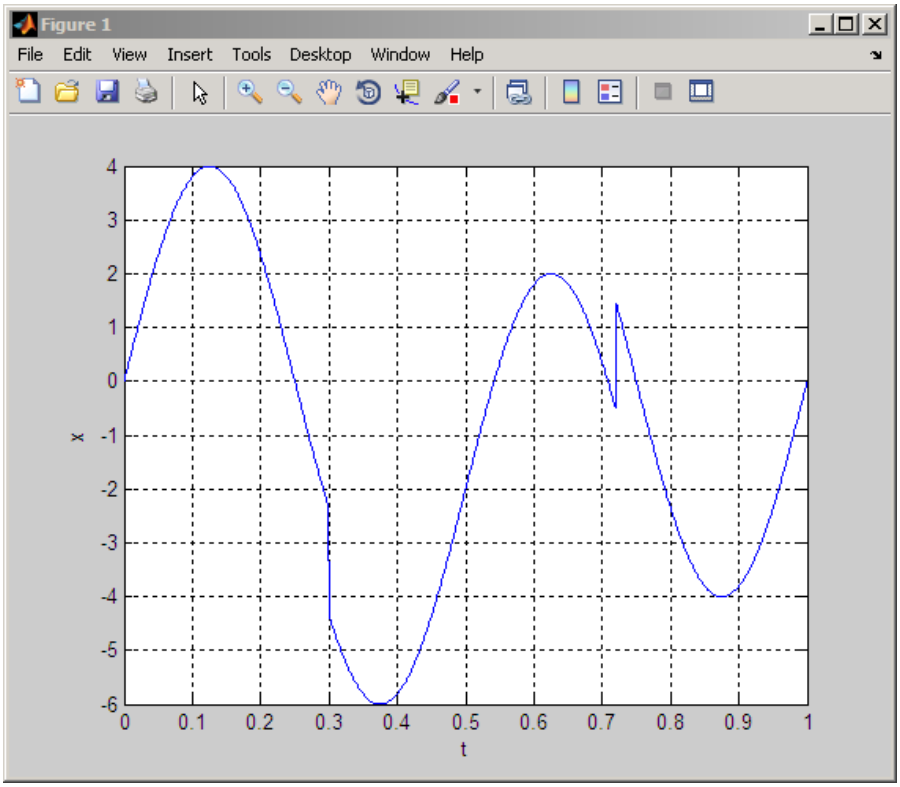
```
freq = scal2frq(9:21,'sym8',1/1000);
```

Treba napomenuti kako su najveći CWT koeficijenti na skalama koje su blizu frekvenciji sinusnog signala. Pomoću sljedećeg koda jasno se može vidjeti sinusni uzorak u CWT koeficijentima.

```
surf(CWTcoeffs); colormap jet;
shading('interp'); view(-60,12);
```

Za kraj će biti predstavljen primjer koji sadrži brze promjene signala ali i glatke oscilacije. Signal je sinus frekvencije 2 Hz s dva ubačena diskontinuiteta.

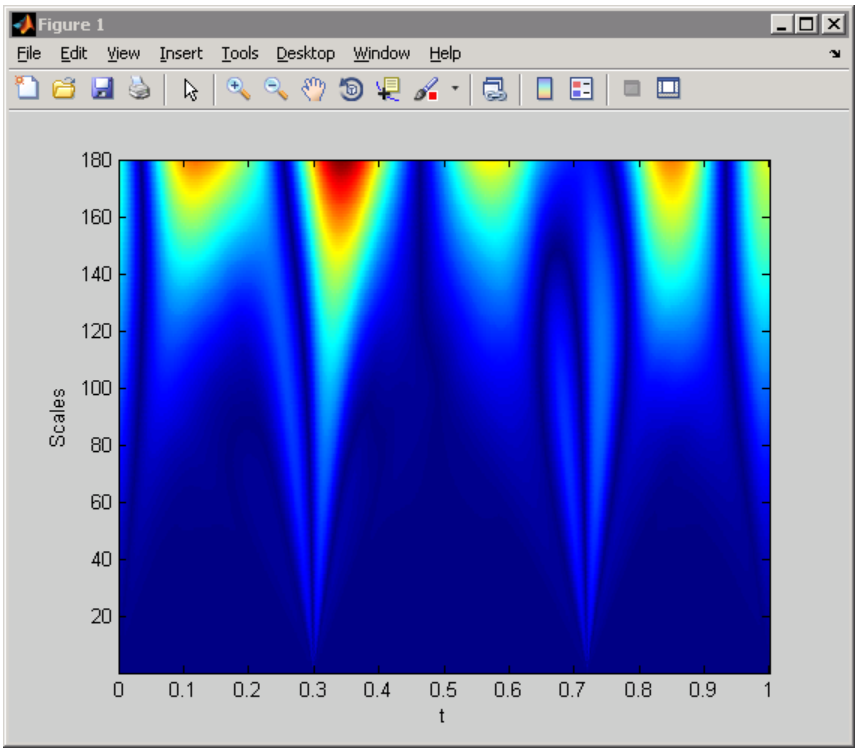
```
N = 1024;
t = linspace(0,1,1024);
x = 4*sin(4*pi*t);
x = x - sign(t - .3) - sign(.72 - t);
plot(t,x); xlabel('t'); ylabel('x');
grid on;
```




Slika 37 - Signal sa sporim i brzim promjenama

Opet će se naći CWT koeficijenti te će biti iscrtani na sljedećoj slici.

```
CWTcoeffs = cwt(x,1:180,'sym4');
imagesc(t,1:180,abs(CWTcoeffs));
colormap jet; axis xy;
xlabel('t'); ylabel('Scales');
```




Slika 38 - CWT signala sa prethodne slike

Na slici se jasno vidi kako je u trenutku $t = 0.3$ te malo nakon trenutka $t = 0.7$ došlo do nekakvih promjena u signalu. Pošto su kod tih promjena izražene jako male skale za zaključiti je da se radi o naglim promjenama signala tj. diskontinuitetima. Također može se vidjeti kako su najveći koeficijenti oko područja velikih skala. To znači da je kod malih frekvencija valića, valić najsličniji signalu. Veliki koeficijenti se nalaze oko točaka $t = 0.1$, 0.3, 0.6 i 0.8. Da se zaključiti da je u tim trenutcima sinusni val imao najveću amplitudu, što znači da su mu u tim trenutcima nastupili dolovi i brjegovi.

# 3. Estimacija frekvencije osnovnog tona u MATLAB-u

## 3.1. Estimacija frekvencije tona STFT-om

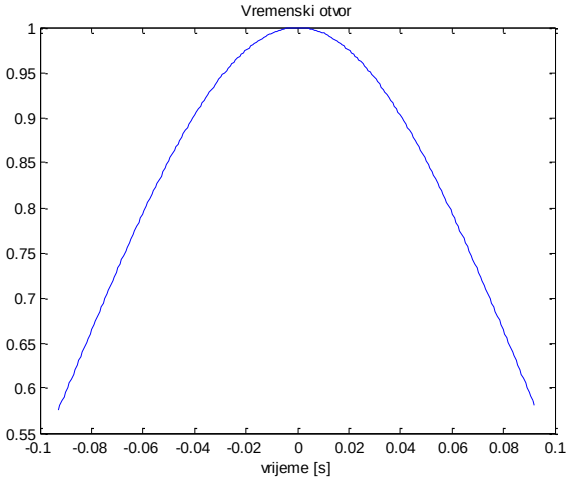
Već je ranije pokazano kako je vremenski kratkotrajna Fourierova transformacija definirana sljedećim izrazom:

$$X(\tau,\omega)=\int_{-\infty}^{\infty}x(t)g^{*}(t-\tau)e^{-j\omega t}dt\,. \tag{9}$$

Za vremenski otvor $g$ izabire se funkcija konačne energije koja ima željena svojstva lokalizacije u vremenskoj i frekvencijskoj domeni. Kako je ranije opisano, jedan dobar izbor za vremenski otvor je Gaussova funkcija. Ona odgovara zbog toga što je jednaka u vremenskoj i frekvencijskoj domeni te ono što je još važnije: produkt efektivnih širina u vremenskoj i frekvencijskoj domeni je minimalan. U ovom radu koristila se jednostavna Gaussova funkcija amplitude $l$ bez pomaka u vremenu:

$$w=e^{\frac{-t^{2}}{l}}, \tag{10}$$

gdje je $t$ polje od 256 vremenskih trenutaka a $l$ broj uzoraka vremenskog otvora tj. 256. Broj od 256 uzoraka uzet je radi bržeg izvođenja brze Fourierove transformacije.




Slika 39 - Vremenski otvor (Gaussova funkcija)

Potrebno je integral STFT-a pretvoriti u oblik pogodan za računanje u MATLAB-u. Eulerovom aproksimacijom, konačan signal trajanja $N$ i funkciju razlaganja jednoliko otipkamo periodom $T$, pa za diskretne pomake $\tau = kT$ aproksimacija glasi:

$$X(kT,\omega) \approx T\sum_{n=0}^{N-1} x(nT)g^{*}(nT-kT)e^{-j\omega nT}. \tag{11}$$

STFT se može izračunati na dva načina. Prvi način je da se umnožak signala i vremenskog otvora shvati kao jedan novi signal, na kojem će se primijeniti Fourierova transformacija.

$$X(kT,\omega) \approx T\sum_{n=0}^{N-1}\left[x(nT)g^{*}(nT-kT)\right]e^{-j\omega nT}. \tag{12}$$

Uzorci se zatim računaju numerički pomoću DFT-a:

$$X(kT,\omega) \approx T\cdot DFT\left[x(nT)g^{*}(nT-kT)\right]. \tag{13}$$

Ovakav način računanja STFT-a korišten je tijekom ovog rada. Ono što je dobro napomenuti je proširenje signala x nulama s lijeve i desne strane. Dodavanje nula omogućuje precizniji FFT u više točaka. Slijedi prikaz jednostavne for petlje koja radi gore opisanu vremenski kratkotrajnu Fourierovu transformaciju.

```
for i=1:ls                      % ls je duljina signala + vrem. otvora
    x1 = x(i:i+lenw-1) .* wg;   % umnožak signala i otvora je novi signal
    X(:, i) = (fft(ifftshift(x1))); % radi se FFT novog signala
end
```

Druga mogućnost računanja STFT-a shvaća signal moduliran eksponencijalnom funkcijom kao novi signal na kojem se primjenjuje konvolucija s preokrenutim vremenskim otvorom. Ovaj pristup ekvivalentan je filtriranju.

$$X(kT,\omega) \approx T\sum_{n=0}^{N-1}\left[x(nT)e^{-j\omega nT}\right]\cdot g^{*}\left[-(kT-nT)\right]. \tag{14}$$

Općenito je implementacija konvolucijom bolja kod kratkih otvora, dok je implementacija FFT-om superiorna kod velikih otvora čiji je broj uzoraka potencija broja 2, što je bio slučaj u ovom radu[4].

## 3.2. Estimacija frekvencije tona CWT-om

Kontinuirana valićna transformacija definirana je već poznatim izrazom :

$$C(a,b;f(t),\psi(t)) = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{a}}\psi^*\left(\frac{t-b}{a}\right)dt \quad (15)$$

.

Kao i kod STFT-a potrebno je izabrati dobru valićnu funkciju psi. Neki od ranije spomenutih valića su:

1) kompleksni Morletov valić: $\psi(t) = C \cdot e^{\frac{-t^2}{2\sigma^2}} e^{-j\omega_0 t}$, (16)

2) realni Morletov valić: $\psi(t) = e^{\frac{-t^2}{2}} \cos(5t)$, (17)

3) sombrero (eng. Mexican hat): $\psi(t) = C(1-t^2)e^{\frac{-t^2}{2}}$, (18)

4) kompleksni Shannonov valić: $\psi(t) = C \cdot \frac{\sin(\omega_b t)}{\omega_b t} e^{-j\omega_0 t}$. (19)

Za numeričko izračunavanje kontinuirane valićne transformacije u MATLAB-u služi funkcija `cwt()` čiji je princip djelovanja objašnjen u poglavlju Implementacija CWT-a u Matlabu. Funkcija je dio *Wavelet Toolboxa*, a opis korištenja je sljedeći:

```
cwt Real or Complex Continuous 1-D wavelet coefficients.
    COEFS = cwt(S,SCALES,'wname') computes the continuous
    wavelet coefficients of the vector S at real, positive
    SCALES, using wavelet whose name is 'wname'.
    The signal S is real, the wavelet can be real or complex.

    COEFS = cwt(S,SCALES,'wname','plot') computes
    and, in addition, plots the continuous wavelet
    transform coefficients.
```

Vektor `S` predstavlja signal nad kojim će se provesti transformacija, vektor `SCALES` sadržava sve skale za koje želimo izračunati CWT, dok je `'wname'` naziv valićne funkcije iz biblioteke.

Podatke o grupama i obiteljima valićnih funkcija raspoloživih u *Wavelet Toolboxu* možemo dobiti funkcijom `waveinfo()`. Podržane su sljedeće grupe:

1) Grubi valići
2) Beskonačno regularni valići
3) Ortogonalni i kompaktno podržani valići
4) Biortogonalni i kompaktno podržani valići
5) Kompleksni valići

U svakoj grupi realizirano je više obitelji valićnih funkcija. Korisnik može definirati i vlastite valićne obitelji pomoću `wavemngr()`.

Diskretne aproksimacije valićnih funkcija možemo dobiti funkcijom `wavefun()` kao u sljedećem primjeru.

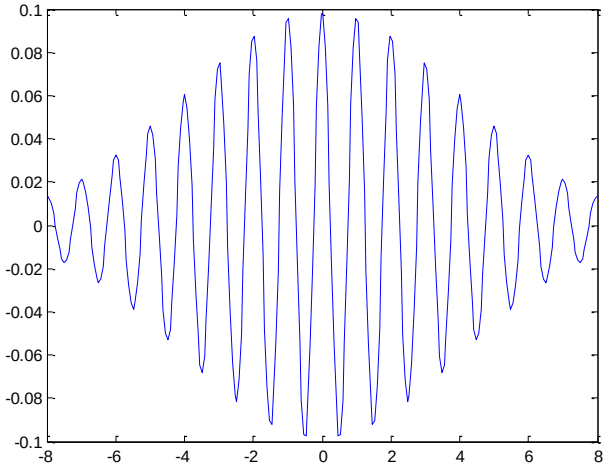
```
[psi, xval] = wavefun('wname'); % 'wname' označava kratki naziv waveleta
figure, plot(xval, psi);        % prikaz wavelet funkcije
```

U ovom radu koristit ćemo kompleksni Morletov valić (`'cmorFb-Fc'`). Ova valićna funkcija je odabrana zbog sličnosti sa STFT funkcijama razlaganja kada je vremenski otvor Gaussov. Razlog zbog kojeg je uzet kompleksni valić je taj što implementacija realnog Morletovog valića u MATLAB-u ne dozvoljava slobodne parametre (vidi izraz za realni Morletov valić). Širina Gaussovog otvora je fiksirana a isto vrijedi i za frekvenciju kosinusne funkcije. Za razliku od realnog, kompleksni valić dozvoljava podešavanje širine Gaussovog otvora te frekvencije kompleksne harmonijske funkcije. Ako se upotrijebi funkcija `waveinfo('cmor')` dobiva se definicija kompleksnog Morletovog valića:

```
Definition: a complex Morlet wavelet is
 cmor(x) = (pi*Fb)^{-0.5}*exp(2*i*pi*Fc*x)*exp(-(x^2)/Fb)
depending on two parameters:
 Fb is a bandwidth parameter
 Fc is a wavelet center frequency
```

Wavelet name cmor"Fb"-"Fc"

Parametri koji određuju širinu i frekvenciju zadaju se uz ime valićne funkcije kao npr. `cmor'1-1', cmor'32-1'` i tako dalje.



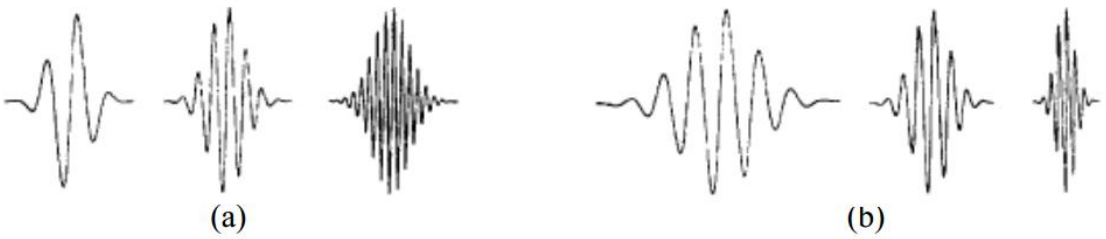

Slika 40 - Kompleksni Morletov valić

## 3.3. Razlika između CWT-a i STFT-a

Kontinuirana valićna transformacija i kratkotrajna Fourierova transformacija imaju mnogo toga zajedničkog. Ipak postoje neke razlike od kojih su najbitnije sljedeće:

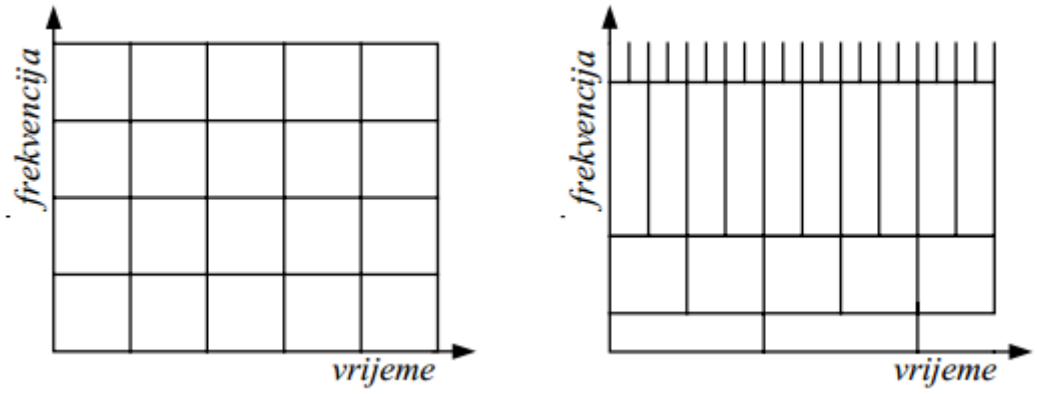
1) kod CWT-a širina vremenskog otvora je promjenjiva ovisno o skali za koju radimo transformaciju
2) nad odsječenom frakcijom signala kod STFT-a se računa Fourierova transformacija dok se kod CWT-a ne računa

Povećavanjem skale kod CWT-a valićna funkcija mijenja se samo u jednom pogledu, a to je da postaje sve šira i šira. Sukladno tome, što se više povećava valićna funkcija to će biti veći i vremenski otvor. Ono što je konstanta kod CWT-a je broj valića u funkciji. S druge strane kod STFT-a širina otvora je konstanta dok je broj valića varijabla. Na sljedećoj slici ovo se jasno vidi [5].




Slika 41 - (a) STFT; (b) CWT

Kao posljedica ove razlike između transformacija javlja se različita podjela vremensko-frekvencijske ravnine. Kod STFT-a razlučivost u ovoj ravnini jednaka je za sve frekvencije. Bez obzira koja frekvencija je u pitanju vremenski otvor je iste širine pa je stoga i frekvencijski pojas uvijek jednake širine. Kod valićne transformacije broj valića uvijek je isti dok se mijenja širina vremenskog otvora. Kako se skale povećavaju raste i veličina vremenskog otvora. To znači da će se na nižim frekvencijama (velike skale) vremenska razlučivost smanjivati, dok će frekvencijska biti detaljnija. Također, kod viših frekvencija (niže skale) vremenski otvor je mali, dakle, vremenska razlučivost je dobra, dok je razlučivost u frekvenciji grublja. Još jednom, vremensko-frekvencijske ravnine izgledaju ovako:

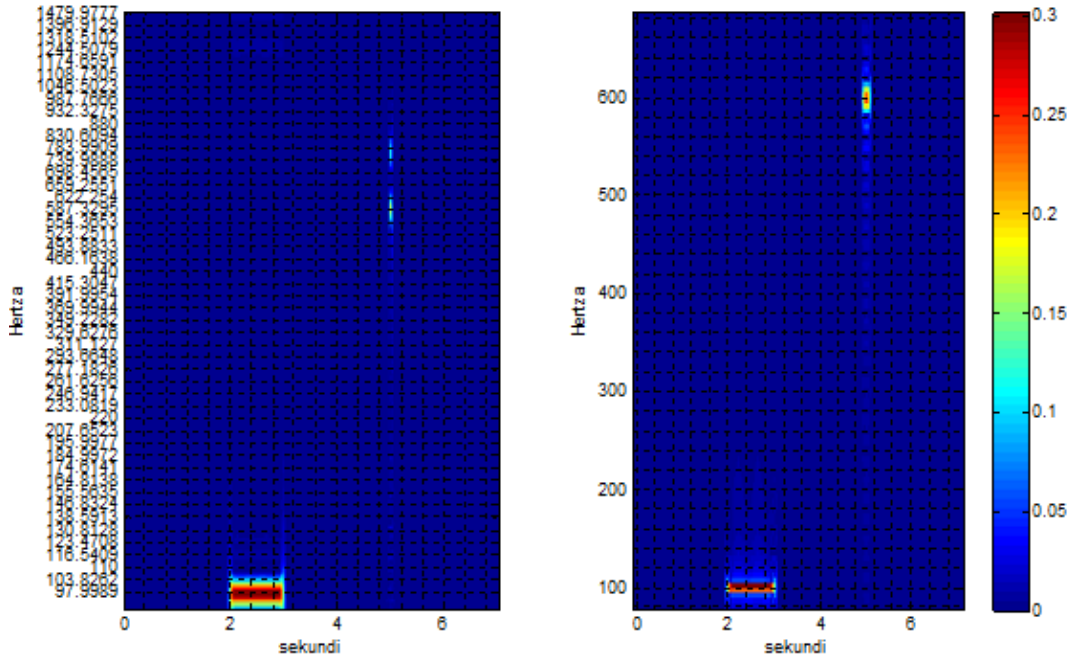



Slika 42 - Vremensko frekvencijske ravnine kod STFT-a i CWT-a

Logično je zapitati se koja transformacija je bolja, tj. prikladnija. Dva su razloga koja podupiru kontinuiranu valićnu transformaciju. Prvi razlog, koji se možda i najbrže nameće, je relativna pogreška. Promotrimo situaciju gdje imamo signal koji je prisutan na širokom pojasu frekvencija te ga je potrebno promotriti kroz vremensko-frekvencijsku ravninu. Pretpostavimo da promatramo dvije frekvencije f1=10 Hz i f2=1000 Hz u vremenu t1=5 s i t2=10 s. Kod traženja frekvencije f1 nema razlike između CWT-a i STFT-a. Obje transformacije jasno će dati do znanja da se radi o frekvenciji f1 jer će najveći koeficijenti biti raspoređeni oko 10 +- 0.5 Hz te oko 5 +- 0.5 s. Ova nesigurnost proizlazi iz nesavršene frekvencijske  i vremenske rezolucije ali nam ne smetaju jer su jako male. Problem dolazi kod frekvencije f2. STFT daje još uvijek dobru rezoluciju u frekvenciji te će pokazati kako se javio signal na frekvenciji 1000 +-0.5 Hz ali će također pokazati da se javio u okolini od

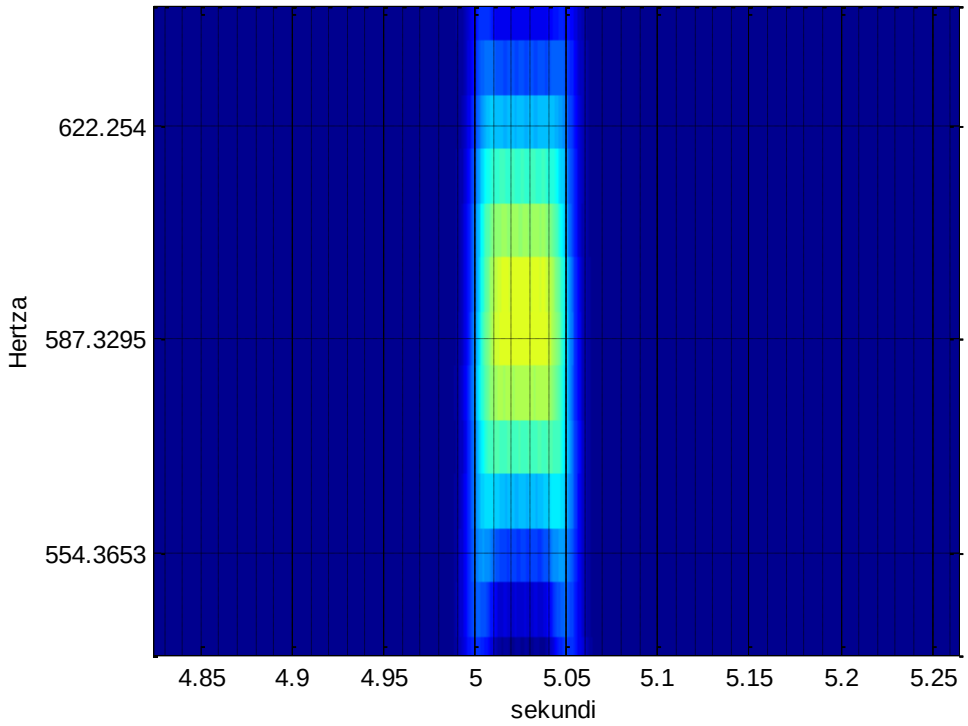
10 +- 0.5 s. Loša strana ovdje je to što je vrijeme jako neprecizno za ovakvu frekvenciju. Ta jedna sekunda nesigurnosti može sadržavati 500 perioda iste frekvencije, a možda nam je upravo od velike važnosti broj perioda frekvencije. S druge strane, CWT će prikazati najveće koeficijente na frekvenciji 1000 +- 50 Hz u vremenu 10 +- 0.005 s. Može se primijetiti značajan pad rezolucije u frekvenciji ali i značajan porast rezolucije u vremenu. Takva rezolucija u vremenu odgovara jer je puno manja od slučaja sa STFT-om. No što je s rezolucijom u frekvenciji? Najveći pogreška koja se može dogoditi je da se u području od 1000 +-50 Hz pretpostavi frekvencija od 950 Hz dok se u stvarnosti pojavila frekvencija od 1050 Hz. Apsolutna pogreška puno je veća nego kod STFT-a no to ne predstavlja veliki problem jer je to relativno mala pogreška od 5 %. Zaključujemo kako u ovakvom slučaju obje transformacije sadrže nesavršenosti, no u odnosu na STFT kod CWT-a su te nesavršenosti prikladnije. Kada je potrebna velika vremenska razlučivost ona se povećava na štetu frekvencijske a kad je potrebna velika frekvencijska razlučivost postiže se na štetu vremenske.

Sličan primjer proveden je u praksi. Signal koji se sastoji od dvije sinusoide analiziran je u vremensko-frekvencijskoj domeni. Jedna sinusoida je frekvencije 100 Hz i traje cijelu sekundu, dok je druga frekvencije 600 Hz te traje dvadesetinu sekunde. Na slici se može vidjeti analiza STFT-om te CWT-om.

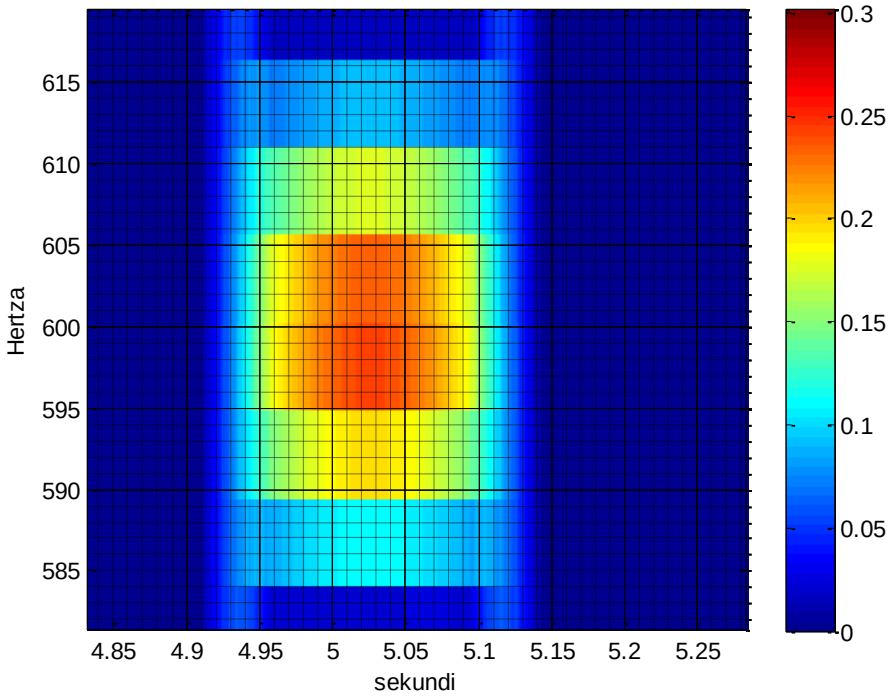



Slika 43 - Transformacije CWT-om te STFT-om

Već sa prve dvije slike jasno je vidljivo kako na frekvenciji od 100 Hz nema velike razlike između CWT-a i STFT-a. Vremenska domena ima dovoljnu razlučivost da točno odredi trajanje signala od druge do treće sekunde. Frekvencijska razlučivost također je zadovoljavajuća te s njom nema problema. Razlika je u višoj frekvenciji od 600 Hz koja ima puno kraće trajanje. Već na prvi pogled može se primijetiti kako je kod STFT-a duže trajanje signala nego kod CWT-a. Kako bi se bolje uočile razlike prikazat će se uvećane slike tog dijela vremensko-frekvencijske ravnine.




Slika 44 - Visoke frekvencije kod CWT-a




Slika 45 - Visoke frekvencije kod STFT-a

Sa prve slike može se zaključiti kako signal počinje negdje oko pete sekunde a završava nakon pet stotinki. Za početak harmonika uzima se svijetloplava boja. Što se tiče vremenske rezolucije može se biti sasvim zadovoljnim jer harmonik u stvarnosti traje samo pet stotinki. Frekvencijska rezolucija dosta je slabija, no ne igra preveliku ulogu. Najsvjetlija boja je između 587 i 622 Hz. Sa sigurnošću se može reći da se tamo nalazi frekvencija koja je od značaja. Ako se zna da je sinusoida frekvencije 600 Hz u najgorem slučaju se dobiva promašaj od 20-ak Hz, što je otprilike 3% pogreške. Prihvatljiva pogreška s obzirom da se dobiva puno točnija i značajnija vremenska rezolucija.

Druga slika pak, najsvjetliji segment ima u području između 595 i 605 Hz. Najgora pogreška u ovom slučaju bi bila 0.08%, što je puno bolje od CWT-a, ali nažalost to nije najtraženija osobina na višim frekvencijama. Vremenska rezolucija, koja je u ovom slučaju puno lošija, igra važniju ulogu. Ukoliko se i uzme najbolji slučaj te se pojava signala percipira samo na žutom dijelu tj. od 4.95 s do 5.1 s, dobiva se puno lošija situacija od one kod CWT-a. Dobiva se trajanje signala od 0.15 s, što je čak tri puta više od stvarnog trajanja signala. Sukladno tomu dokazuje se kako je kod većeg frekvencijskog raspona bolje koristiti kontinuiranu valićnu transformaciju.

Na početku poglavlja spomenuto je kako postoje dva razloga koja stavljaju kontinuiranu valićnu ispred kratkotrajne vremenske Fourierove transformacije. Drugi razlog nije općenit, već je vezan uz prirodne pojave i fenomene među kojima je i tema ovog rada tj. percepcija zvuka. Naime osjećaj visine ovisi o frekvenciji zvuka tj. frekvencijskom spektru kod složenog zvuka. Osim frekvencije, subjektivni osjećaj visine tona ovisi i o intenzitetu ali to trenutno nije od interesa. Većina ljudi može odrediti relativnu visinu tonova (viši , niži) ako su tonovi emitirani jedan za drugim, dok rijetki mogu odrediti pravu visinu tona (osobe s apsolutnim sluhom). Ljudsko uho registrira promjene visine tona logaritamski, pa vrijedi Weber-Fechnerov zakon. Subjektivni osjećaj povećanja visine tona za isti interval dobit će se ako se frekvencija poveća za isti postotak, a ne za jednaku apsolutnu vrijednost (npr. jednako će se doživjeti interval između 100 i 120 Hz kao i između 1000 i 1200 Hz). U oba slučaja postotak povišenja niže frekvencije na višu je isti – 20%, dok je u apsolutnim vrijednostima prvi interval iznosio samo 20 Hz, a drugi 200 Hz. Zbog toga se frekvencijske skale najčešće prikazuju kao logaritamske jer je to najsličnije načinu percepcije visine zvuka kod ljudi.

| Calculation for Equal-Tempered tuning [A3 = 440Hz] | | | | | | | |
|---|---|---|---|---|---|---|---|
| Hertz | | Octave=0 | Octave=1 | Octave=2 | Octave=3 | Octave=4 | Octave=5 |
| 0 | A | 55.000 | 110.000 | 220.000 | 440.000 | 880.000 | 1,760.000 |
| 1 | A#/Bb | 58.270 | 116.541 | 233.082 | 466.164 | 932.328 | 1,864.655 |
| 2 | B | 61.735 | 123.471 | 246.942 | 493.883 | 987.767 | 1,975.533 |
| 3 | C | 65.406 | 130.813 | 261.626 | 523.251 | 1,046.502 | 2,093.005 |
| 4 | C#/Db | 69.296 | 138.591 | 277.183 | 554.365 | 1,108.731 | 2,217.461 |
| 5 | D | 73.416 | 146.832 | 293.665 | 587.330 | 1,174.659 | 2,349.318 |
| 6 | D#/Eb | 77.782 | 155.563 | 311.127 | 622.254 | 1,244.508 | 2,489.016 |
| 7 | E | 82.407 | 164.814 | 329.628 | 659.255 | 1,318.510 | 2,637.020 |
| 8 | F | 87.307 | 174.614 | 349.228 | 698.456 | 1,396.913 | 2,793.826 |
| 9 | F#/Gb | 92.499 | 184.997 | 369.994 | 739.989 | 1,479.978 | 2,959.955 |
| 10 | G | 97.999 | 195.998 | 391.995 | 783.991 | 1,567.982 | 3,135.963 |
| 11 | G#/Ab | 103.826 | 207.652 | 415.305 | 830.609 | 1,661.219 | 3,322.438 |
| 12 | A | 110.000 | 220.000 | 440.000 | 880.000 | 1,760.000 | 3,520.000 |

Slika 46 - Tablica tonova i pripadnih frekvencija

Da bi se dvije bliske frekvencije slušno razlikovale kao različite tonske visine, minimalna razlika među njima mora iznositi oko 2.5 %. Dakle razlikovat će se frekvencije 100 i 102.5 Hz dok će sve manje frekvencije od 102.5 Hz biti smatrane kao frekvencija od 100 Hz.

Ovaj način "rada" ljudskog uha puno pomaže kod CWT-a. Već se nekoliko puta spomenulo kako je na niskim frekvencijama rezolucija (frekvencijska) jako dobra tj. mogu se detektirati male promjene u frekvenciji. Dakle, moguće je razlikovati ton A i A# u nultoj oktavi iako je razlika samo 8 Hz. Što se više kreće prema području viših frekvencija rezolucija postaje lošija te je sada nemoguće detektirati promjenu od 8 Hz. No ako naše uho nije u stanju razlikovati takvu malu promjenu na visokim frekvencijama nema razloga da je CWT precizno detektira, ukoliko će prepoznati ton koji se pojavio. Sa slike se jasno može vidjeti kako u višim oktavama razlika između dva tona može biti i do 200 Hz. Zaključuje se kako je drugi razlog zbog kojega valićna transformacija više odgovara nego STFT, njezina sličnost s ljudskim (prirodnim) načinom percepcije zvuka.

## 3.4. Detekcija tonova

Nakon što se napravi CWT ili STFT dolazi se do problema detekcije tonova. Koeficijenti prikazani u vremensko-frekvencijskoj ravnini lako se mogu detektirati ukoliko to radi čovjek, no problem dolazi kada se želi da aplikacija automatski sama zapisuje tonove. U ovom radu analiziralo se nekoliko mogućih rješenja detekcije tonova, neki s više neki s manje uspjeha.

Prvi pristup detekciji tonova bio je dakako najjednostavniji. Ideja je bila da se prije transformacije koeficijenata odredi prag koji koeficijent mora proći kako bi se detektirao ton. Pragovi bi se nalazili empirijski na temelju subjektivne ocjene promatrača. Prednost ove ideje je dakako njezina brzina i jednostavnost, dok je nedostatak točnost detekcije u nekoliko situacija. Neka se uzme na primjer da je signal jako slab. Iako će ljudsko oko u vremensko-frekvencijskoj ravnini prepoznati da su se pojavili neki tonovi, koeficijenti neće proći razinu praga i sustav ih neće detektirati. Slična je situacija s preglasnim signalom. Čovjek će prepoznati da je signal snažniji nego obično te će za detekciju uzeti samo najjače tonove dok će one slabije zanemariti. S druge strane, sustav će, zbog predefiniranog praga, detektirati i one slabije tonove iako su puno slabiji od velike većine. Konačno može doći i do situacije da je signal bijeli šum. Sadrži sve frekvencije i tonove pa će sustav ovdje "pogrešno" detektirati pojavu svih tonova.

Sljedeći pristup temelji se na prvom uz nekoliko preinaka. Princip detekcije tonova pomoću amplitude koeficijenata transformacije i dalje ostaje isti, ali se pragovi definiraju tek nakon transformacije signala. Ideja je da se nađe maksimum svakog tona kroz cijeli signal. Nakon što se našao maksimum postavlja se prag koji će iznositi određeni postotak tog maksimuma. Dakle za svaki ton postoji zasebni prag. Ovakvim pristupom rješavamo se problema detekcije preslabih tonova radi preslabog signala. Također rješava se problem ukoliko je signal prejak tj. preglasan. Mana je ta da bez obzira koliko bio slab svaki se ton barem jednom u vremenu mora detektirati. Ovaj problem se može riješiti jednostavno tako da se maksimumi svakog tona uspoređuju s maksimumom svih tonova ili sa srednjom vrijednosti svih maksimuma. U tom slučaju mogu se izbaciti tonovi koji imaju premali maksimum u odnosu na ostale signale.

Sljedeća dva principa malo su složenija a kako bi ih se što bolje shvatilo potrebno je definirati pojam Wienerove entropije (eng. *Spectral flatness*).

Wienerova entropija je mjera koja se koristi u digitalnoj obradi signala kako bi se okarakterizirao audio spektar. Tipično se mjeri u decibelima a određuje način na koji se mjeri koliko je zvuk sličan tonu te koliko je sličan šumu. Značenje riječi ton u ovom kontekstu je u smislu broja vrhova ili rezonantne strukture u spektru snage, kao opreka ravnom spektru bijelog šuma. Visoka spektralna ravnoća (eng. *Spectral flatness*; za bijeli šum približava se jedinici) ukazuje kako spektar sadrži sličnu količinu snage u cijelom frekvencijskom pojasu, što je slično bijelom šumu. Graf će u tom slučaju (amplitudno-frekvencijska karakteristika) biti relativno ravan i gladak. Niska spektralna ravnoća (nula za čisti ton) ukazuje da je spektralna snaga koncentrirana u relativno malom frekvencijskom području. Graf u ovom slučaju izgleda poput šiljka.

Wienerova entropija računa se kao omjer geometrijske srednje vrijednosti te aritmetičke srednje vrijednosti snage spektra [6] tj.:

$$WE = \frac{\sqrt[N]{\prod_{n=0}^{N-1} x(n)}}{\frac{\sum_{n=0}^{N-1} x(n)}{N}} = \frac{e^{\left(\frac{1}{N}\sum_{n=0}^{N-1} \ln x(n)\right)}}{\frac{1}{N}\sum_{n=0}^{N-1} x(n)}, \tag{20}$$

gdje $x(n)$ označuje amplitudu uzorka broj $n$. Treba uočiti da se za bilo koji uzorak jednak nuli ravnoća približava nuli, tako da ova mjera nije korisna kada su uzorci jednaki nuli.

Ovaj omjer često se računa u decibelima pa je tada maksimum jednak 0 dB dok je minimum jednak $-\infty$ dB. Također, može se mjeriti i samo kroz jedan dio pojasa umjesto kroz cijeli pojas.

Ova mjera je jedna od mnogih audio deskriptora koji se koriste u MPEG-7 standardu, gdje je označena kao *"AudioSpectarFlatness"*.

U ovom radu će se koristiti malo izmijenjena verzija Wienerove entropije. Dovoljno će biti samo zamijeniti brojnik i nazivnik kako bi dobili sljedeći izraz:

$$WE^{-1} = \frac{\frac{\sum_{n=0}^{N-1} x(n)}{N}}{\sqrt[N]{\prod_{n=0}^{N-1} x(n)}}. \tag{21}$$

Razlog zbog kojeg će se invertirati entropija je čisto estetski jer autor ovog teksta voli kada nešto što je pozitivno teži višim brojevima a ne nuli.

Predstavit će se dvije opcije detekcije tona koje su povezane s idejom Wienerove entropije. Prva je ideja zasebno napraviti aritmetičku i geometrijsku srednju vrijednost svih tonova u određenom vremenskom trenutku (u ovom slučaju jedna osmina sekunde). Nakon toga jednostavno podijelimo aritmetičku srednju vrijednost (eng. *aritmetic mean*, AM) s geometrijskom srednjom vrijednosti (eng. *geometric mean*, GM). Ovim slučajem riješili smo sve probleme oko jačine signala. Bez obzira na jačinu signala, gdje god postoji relativni skok u amplitudi kod tona omjer će nam ukazati da je došlo do pojave tona. Jedini problem koji postoji ovdje je za slučaj da je ton prisutan kroz cijeli vremenski odsječak. Naravno, jedno rješenje je produžiti vremenski segment na cijeli signal no to rješenje je nespretno jer se korisnik stavlja u poziciju da ovisi o duljini signala i trajanju pojedinog tona.

Druga ideja je da se napravi geometrijska srednja vrijednost svih tonova u vremenskom segmentu koji promatramo. Ovim potezom dobiva se referenca koja je neovisna o trajanju jednog tona kao što je to bilo u prethodnom slučaju. Ta referenca usporedit će se s aritmetičkom srednjom vrijednosti svakog tona zasebno. Omjer AM-a i GM-a dat će koeficijente koji će određivati (ne)prisutnost tonova u određenom vremenu. Jedina situacija gdje postoji potencijalni problem je ta kada se zbog jednog jako izraženog tona cijela srednja vrijednost tonova poveća, te tako onemogući manjim tonovima da budu

detektirani. Zbog ove situacije, kao referentna vrijednost svih tonova u tom trenutku uzeta je geometrijska srednja vrijednost jer kod nje, za razliku od aritmetičke, srednja vrijednost neće toliko porasti zbog jednog izoliranog slučaja. Ovakva detekcija tonova korištena je i u ovom radu, te je implementirana u aplikaciju prepoznavanja tonova.

## 3.5. Detekcija frekvencije osnovnog tona

Svaki realni zvuk se sastoji od mnogo frekvencijskih komponenata. Da bi postojala samo jedna komponenta, zvuk bi se trebao sastojati od savršenog sinusnog vala beskonačnog trajanja. Ukoliko dolazi do bilokakvih modulacija zvuka (gašenje, paljenje), povećava se spektralna složenost signala (dodaju se frekvencijske komponente)[7].

Realni zvukovi nisu savršene sinusoide te svaka deformacija sinusoide rezultira stvaranjem harmonijske serije. Harmonijska serija je niz harmonika, gdje je frekvencija svakog harmonika cjelobrojni višekratnik osnovne frekvencije. Postavlja se pitanje kako se detektira osnovni ton tj. prvi harmonik.

Prema pristupu razlikujemo dvije vrste metoda estimacije glavne frekvencije. Prve su metode u vremenskoj domeni a druge su metode u frekvencijskoj domeni. Ove prve se koriste kod monofonih melodija dok je transkripcija polifonih melodija najčešća u frekvencijskoj domeni[8]. Pregled nekih od metoda estimacije visine osnovnog tona prikazan je u nastavku.

### 3.5.1. Metode u vremenskoj domeni

U vremenskoj domeni detekcija osnovne frekvencije temelji se na estimaciji perioda kvazi periodičnog signala, koja se kasnije prebacuje u frekvenciju.

Jedan jednostavan pristup je mjerenje udaljenosti između dijelova signala koji prolaze kroz nulu (eng. *zero crossing*). Te udaljenosti se koriste kako bi algoritam usporedio uzastopne vremenske intervale i tako odredio frekvenciju. Slična metoda je mjerenje udaljenosti između vrhova signala. Nedostatak ove metode (i njezinih varijacija) je ograničenost pri radu sa realnim glazbenim signalima. Problem se javlja zbog toga što kod stvarnih signala postoji više frekvencija zbog kojih će signal u vremenu također prolaziti kroz nulu i time unositi grešku u sustav. Upravo radi toga ova metoda je dobra samo kada je ulazni signal

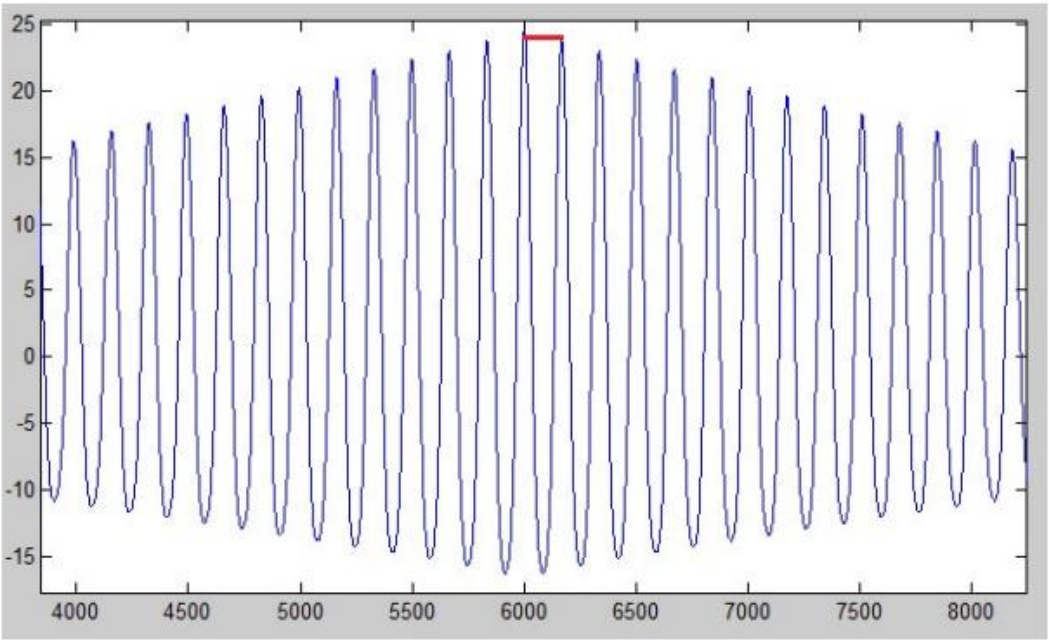
približan sinusoidi tj. kada je broj prolazaka kroz nulu približan kao i kod osnovne frekvencije. Osim ove, razvijena je i poboljšana verzija transkripcije monofonog signala tako da se signal filtrima rastavi u niz komponenata. Svaki filtar propušta pola oktave i time omogućava da izlaz sadrži samo jedan valni oblik bez harmonika. Nakon toga se na izlazu svakog filtra vrši *zero crossing* metoda. Algoritam dalje ustvrđuje koji od izlaza filtara sadrži osnovnu frekvenciju pomoću njihovih snaga. Ova metoda u obzir uzima i činjenicu da harmonik višeg reda u signalu može biti veći od osnovnog

Malo složenije metode su one gdje se dio signala uspoređuje sa svojom zakašnjelom verzijom. Metoda auto korelacije radi upravo to kako bi našla ponavljajući uzorak. Veličina segmenta koji se uspoređuje ovisi o implementaciji ali se obično uspoređuje točka po točka. Naravno tijekom usporedbi vrijeme kašnjenja stalno raste. Sređeni izraz za auto korelaciju nalazi se u nastavku.

$$autokorelacija[kašnjenje] = \sum_{n=0}^{N} signal[n] \times signal[n + kašnjenje], \tag{22}$$

gdje je $n$ uzorak, a kašnjenje se kreće od 0 do $N$.

Maksimum korelacijske funkcije dolazi kada se signali podudaraju a jasno se vidi na slici 47. Osnovna frekvencija signala nalazi se u vremenu kašnjenja između maksimuma korelacijske funkcije (na slici x označeno crvenom bojom).




Slika 47 - Maksimum korelacijske funkcije

Ova metoda najučinkovitija je na zvukovima niske do srednje frekvencije pa nije najprikladnija za glazbene signale koji pokrivaju puno šire frekvencijski pojas. Postoje i poboljšane verzije ove metode no unatoč poboljšanjima mogu se riješiti samo trodijelne polifonije, i to samo kad notni zapisi sadrže vrlo slične jačine zvuka. Auto korelacija generalno ima problema kod harmonijski kompleksnih signala.

Slična metoda se rabi i kod Yin estimatora osnovne frekvencije. U ovom slučaju umjesto da se maksimizira produkt valnog oblika i njegove zakašnjele verzije, pokušava se minimizirati pogrešku. Prednosti nad auto korelacijom su određivanje osnovnih frekvencija glasovnih i glazbenih signala visokih osnovnih frekvencija te stopa pogreške koja je triput niža. Mana ove metode je što se koristi samo za detekciju osnovne frekvencije monofonih signala.

### 3.5.2. Metode u frekvencijskoj domeni

Cilj algoritama u frekvencijskoj domeni je podijeliti zvučni signal na njegove frekvencijske komponente. Kada se signal prebaci iz vremenske domene u frekvencijsku dobiva se uvid u energiju signala na određenim frekvencijskim pojasevima. Tamo gdje su energije visoke treba otkriti predstavlja li ta frekvencija osnovnu frekvenciju, dio harmonika ili šum. Za pomoć oko ovakvog problema služe metode opisane u nastavku.

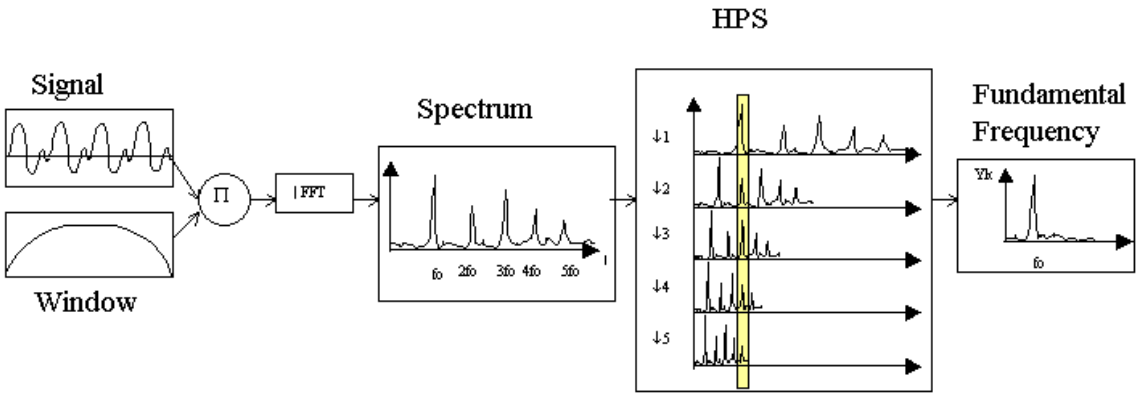
Metoda koja razdvaja dio zvuka koji sadrži osnovnu frekvenciju od ostatka spektra naziva se kepstralna analiza. Ime *kepstar* izvedeno je tako da se prvi slog riječi spektar napiše obrnutim poretkom (SPEK -> KEPS). Uglavnom se koristi kod govornih signala no može biti koristan i za određivanje notnog zapisa izvedenih na nekom instrumentu. Zbog modulacije signala koji je stvorio zvuk (bilo u tijelu instrumenta, bilo zbog medija kojim se prenosi signal) pojavit će se harmonici u frekvenciji te će kepstar ovdje odvojiti osnovnu frekvenciju od ostalog spektra. Definicija kepstra je spektar snage logaritma spektra snage, a izrazi su napisani u nastavku.

$$KEPSTAR = \left| F^{-1}\left\{ \log\left( \left| F\{f(t)\} \right|^2 \right) \right\} \right|^2 \tag{23}$$

Kao što se može vidjeti iz izraza analiza počinje Fourierovom transformacijom vremenskog signala. Nakon što se dobila Fourierova transformacija i snaga u frekvencijskoj domeni, računa se logaritam snage kako bi se signal doveo u korisno područje. Nakon toga izvodi se inverzna Fourierova transformacija te se tako dobiva valni

oblik koji sadrži jedan maksimum povezan s rezonancijom zvuka te drugi maksimum koji predstavlja segment zvuka s osnovnom frekvencijom. Ona se računa tako da odredi udaljenost između početka valnog oblika i mjesta drugog maksimuma na vremenskoj osi. Ova metoda najdjelotvornija je s govornim signalima no kada se primjeni na glazbene signale njena učinkovitost pada, te se stoga i ne koristi u sustavima za obradu polifonih signala.

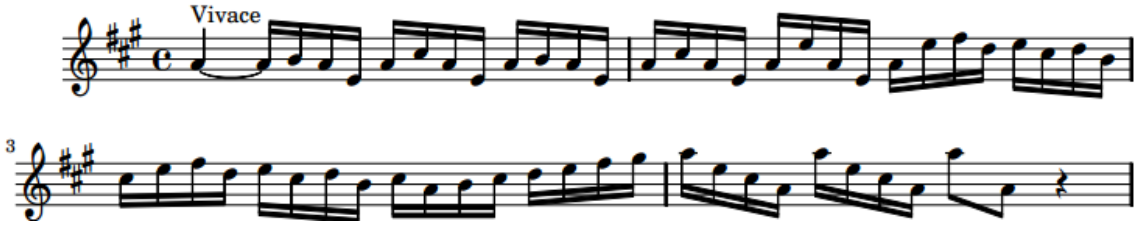
Vrijedi spomenuti i metodu spektra umnoška harmonika (eng. *harmonic product spectrum*; HPS), koja je bazirana na Fourierovoj transformaciji. Ovaj algoritam iskorištava tendenciju glazbenih signala da prikazuju jake harmonične strukture. Algoritam radi tako da se poduzorkuje spektar te se zatim rezultirajući spektri množe. Poduzorkovanje u frekvencijskoj domeni zapravo privlači harmonike prema nuli poravnavajući osnovnu frekvenciju i njezine harmonike (poduzorkovanje s 2 poravnava drugi harmonik s osnovnom frekvencijom, poduzorkovanje s 3 poravnava treći harmonik s osnovnom frekvencijom i tako dalje). Kada se svi spektri zajedno pomnože, stvorit će se veliki impuls na osnovnoj frekvenciji. Kada bi postojao savršeni sustav gdje bi svi harmonici na idealnim lokacijama (točnim višekratnicima osnovne frekvencije) imali amplitudu 1 a sve ostale frekvencije bi bile s amplitudom 0 ova situacija mogla bi se shvatiti kao logička "I" (eng. AND) operacija, koja rezultira samo jednim vrhom na osnovnoj frekvenciji. U realnim (stvarnim) primjerima harmonici i osnovna frekvencija nemaju jednake amplitude. Ono što je još gore je da ostale frekvencije nisu 0. No bez obzira na nesavršenosti ovaj princip detekcije osnovne frekvencije kod glazbenih signala radi zadovoljavajuće.




Slika 48 - Algoritam HPS-a

# 4. Rezultati

Kako bi se provjerile metode estimacije frekvencije, kroz sustav će se propustiti monofona melodija flaute. Flauta predstavlja odličan izbor za referencu jer je njezin zvuk najsličniji sinusu tj. njezini tonovi imaju najčišće vibracije. Pošto flauta nije toliko popularan instrument kao npr. gitara, teško je pronaći osobu koja će direktno odsvirati melodiju za potrebe ovog rada. Preostaje pronaći neku snimku melodije flaute na Internetu te notni zapis pripadajuće melodije. Prikladna skladba za ovu primjenu je *Fantasia 1* Georga Philippa Telemanna. Za usporedbu je dovoljno i par sekundi skladbe, a notni zapis koji slijedi prikazuje prvih 15 sekundi tj. prva četiri takta.

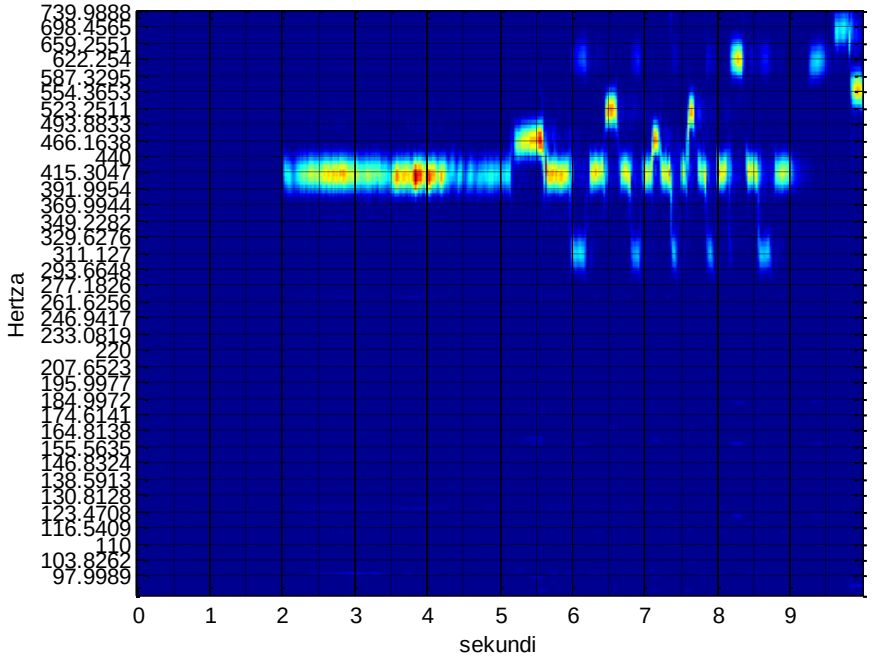



Slika 49 - Notni zapis prvih 15 sekundi *Fanatasie 1*

Skladbu je potrebno "prevesti" u tonove te će prva četiri takta izgledati ovako:

1. Takt: A3 (A3) B3 A3 E2 A3 C#3 A3 E2 A3 B3 A3 E2
2. Takt: A3 C#3 A3 E2 A3 E3 A3 E2 A3 E3 F#3 D3 E3 C#3 D3 B3
3. Takt: C#3 E3 F#3 D3 E3 C#3 D3 B3 C#3 A3 B3 C#3 D3 E3 F#3 G#3
4. Takt: A4 E3 C#3 A3 A4 E3 C#3 A3 A4 A3

Prije same analize melodije potrebno je napomenuti kako je skladba preuzeta sa *youtubea*, nakon toga je prebačena u *wav* format te naposljetku izrezana na svega 10 sekundi. CWT transformacija izrezanog signala izgleda ovako:




Slika 50 - *Fantasia 1* (10 sekundi) u vremensko-frekvencijskoj ravnini

Algoritam bi pomoću ovih koeficijenata trebao prepoznati tonove koji su pročitani iz notnog zapisa. Postupak detekcije je sljedeći:

1) Odrediti koeficijente za pojavljivanje svakog tona kao što je objašnjeno u poglavlju 3.4.

2) Izbrisati tonove koji se pojavljuju kao posljedica nesavršenosti CWT rezolucije ali i instrumenta kojim se svira. Drugim riječima, na slici 50 vidljivo je kako nikada nije točno jedna frekvencija detektirana nego su uvijek poput otoka u vremensko-frekvencijskoj ravnini. Kao što je već spomenuto, ove dodatne frekvencije koje dolaze uz odsvirani ton posljedica su nesavršenosti instrumenata koji se koriste (kako glazbenih tako i tehničkih). Željena frekvencija uvijek je najveće amplitude te se prema tom kriteriju, u okolini najveće frekvencije, brišu ostale po amplitudi manje frekvencije.

3) Potrebno je izbrisati sve harmonike koji se javljaju kao posljedica modulacije osnovnog tona. Ovdje će se koristiti metoda koja iskorištava vremensko-frekvencijsku ravninu. Poznato je kako su harmonici višekratnici osnovne frekvencije. Jednostavnim postupkom brišu se svi tonovi koji se pojavljuju u istom trenutku kao i tonovi koji su točno za oktavu niže. Dakle, ukoliko se istovremeno

pojavi ton A3 i ton A2, detektirat će se ton A2, dok će A3 biti prepoznat kao drugi harmonik te se neće detektirati.

Na sljedećim slikama prikazat će se rezultati dobiveni ovim algoritmom.

| | 2 | 2.125 | 2.25 | 2.375 | 2.5 | 2.625 | 2.75 | 2.875 | 3 | 3.125 | 3.25 | 3.375 | 3.5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D#2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G#2 | 0 | 95.6195 | 96.3353 | 102.8608 | 105.1309 | 104.0317 | 103.3571 | 98.7118 | 88.5762 | 85.1813 | 85.2840 | 82.4250 | 83.6220 |
| A#3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C#3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D#3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Slika 51 - Detektirani tonovi [2 3.5] s

Vidljivo je kako se općenito pojavljuje samo 7 tonova jer tablica sadrži samo 7 redaka. Od druge pa sve do pete sekunde detektiran je samo ton G#2. Prema notnom zapisu potrebno je detektirati ton A3 koji je na ljestvici za pola tona više.

| | 5.125 | 5.25 | 5.375 | 5.5 | 5.625 | 5.75 | 5.875 | 6 | 6.125 | 6.25 | 6.375 | 6.5 | 6.625 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D#2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 41.8945 | 19.5264 | 0 | 0 | 0 |
| G#2 | 101.6626 | 0 | 0 | 0 | 0 | 79.0386 | 96.3753 | 54.4733 | 0 | 21.4752 | 82.8291 | 41.1265 | 0 |
| A#3 | 0 | 63.1412 | 105.9336 | 98.8804 | 61.3969 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 35.9296 | 68.123 |
| C#3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D#3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Slika 52 - Detektirani tonovi [5.125 6.625] s

Prema notnom zapisu nakon tonova A3 slijede tonovi B3, A3, E2, A3 i C#2. Ovdje se također primjećuje da su svi tonovi niži za poluton nego što bi trebali biti.

| | 8.5 | 8.625 | 8.75 | 8.875 | 9 | 9.125 | 9.25 | 9.375 | 9.5 | 9.625 | 9.75 | 9.875 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D#2 | 0 | 21.9769 | 39.2035 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G#2 | 73.8735 | 38.9154 | 0 | 73.7285 | 97.8585 | 57.7075 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A#3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C#3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 29.1486 | 56.0281 |
| D#3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 85.9131 | 84.5405 | 0 | 0 | 0 | 0 |
| F3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 38.6549 | 82.9251 | 33.3288 | 0 |

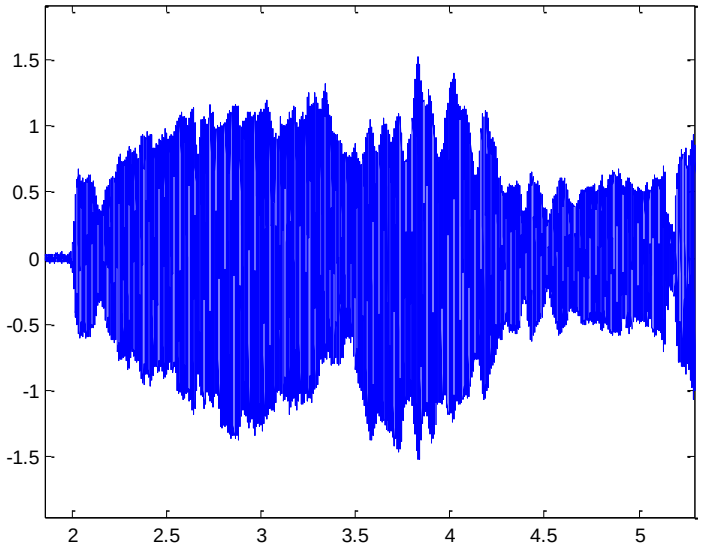
Slika 53 - Detektirano tonovi [8.5 10] s

Ista stvar se primjećuje kroz svih 10 sekundi skladbe (slika 53).

Iz priloženog možemo zaključiti kako su svi detektirani tonovi niži za jedan poluton od notnog zapisa skladbe. Je li ovaj rezultat zadovoljavajući? Koji je razlog ovoj grešci? Za početak potrebno je naglasiti kako je razlika od jednog polutona jako mala te ju je stoga teško detektirati ljudskom uhu (lakše se detektira ako se odsviraju dva tona jedan za drugim). Nekoliko je razloga zbog kojih je moglo doći do ove pogreške. Prvi je da algoritam nije dobro naštiman te će uvijek davati za jedan poluton nižu frekvenciju. Ova opcija je malo vjerojatna jer se razlike između tonova stalno mijenjaju, tako da ne postoji apsolutna pogreška (eventualni *offset* sustava). Drugi razlog je moguća pogreška prilikom izvođenja skladbe, gdje je izvođač slučajno otišao pola tona niže. Treća i najvjerojatnija mogućnost je ta da je flauta naštimana pola tona niže (namjerno ili slučajno). Treba napomenuti kako je do pogreške moglo doći i prilikom prebacivanja melodije u digitalni oblik odnosno, kasnije, kod prebacivanja iz jednog formata u drugi. Teško je sa sigurnošću reći gdje je pogreška ukoliko se ne zna po kojem notnom zapisu je izvođač odsvirao ovu skladbu, no i bez toga, pogreška od pola tona bit će zadovoljavajuća.

Osim detekcije frekvencije, koja je ovdje najvažniji element, proučit će se i vremenska rezolucija rezultata. Naime, potrebno je provjeriti detektira li algoritam tonove u vremenskim trenucima u kojima su stvarno bili odsvirani. Za primjere ćemo uzeti tonove A3, E3, i F3.

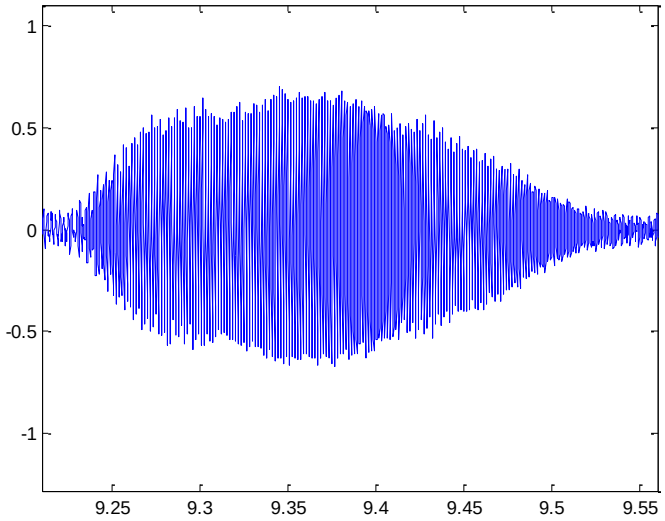
Ton A3 najduže se pojavljuje u signalu pa će se on prvi promatrati. Radi shvaćanja vremenske rezolucije, još jednom će se opisati pojedini dijelovi algoritma koji utječu na nju. Ulaz ovog sustava je audio zapis pohranjen u *wav* formatu frekvencije otipkavanja 44100 Hz. Ova količina informacije prevelika je za potrebe ovog algoritma te se radi njegovog ubrzavanja dodatno smanjuje na 5512,5 Hz što je i više nego dovoljno. Vremenska domena tona A3 može se vidjeti na slici 54. Zbog različitih amplituda istog tona, početak tona ne može se egzaktno odrediti pa će se uzeti okvirna vrijednost čija će točnost biti zadovoljavajuća. Isto vrijedi i za označavanje kraja tona. U ovom primjeru odredilo se protezanje tona A3 u intervalu od 2 s do 5.15 s. Prebacivanje iz vremenske u vremensko-frekvenciju domenu ostavlja isti broj uzoraka u vremenskoj domeni. Ovo znači da će vremenska rezolucija biti ista kao i u frekvencijskoj domeni, no zbog preglednosti rezultata rezolucija će se morati smanjiti. Algoritam pretpostavlja da je 1/8 sekunde dovoljno mala rezolucija da ne dolazi do velikih pogrešaka a puno je veća od početne rezolucije, pa će dati pregledne rezultate. Postavlja se pitanje kako dobiti uzorke za ovu smanjenu rezoluciju.



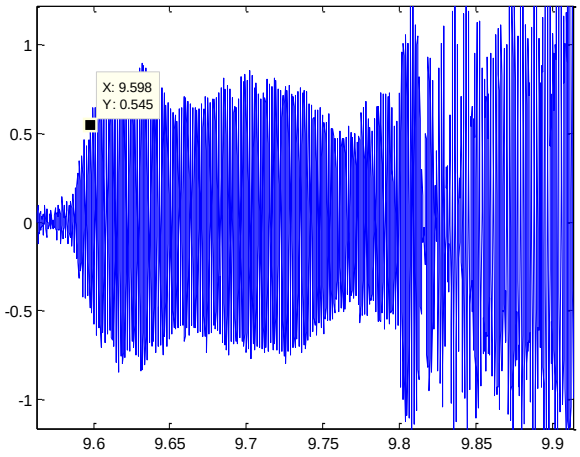

Slika 54 - Ton A3 u vremenskoj domeni

Ranije je spomenuto (poglavlje 3.4) kako će se za detekciju tonova koristiti aritmetičke i geometrijske srednje vrijednosti uzoraka pa će se pomoću njih dobiti koeficijenti koji detektiraju prisutnost tonova. Umjesto dodatnog poduzorkovanja signala na 8 Hz, napravit će se dvije tablice gdje će jedna sadržavati aritmetičku srednju vrijednost, a druga geometrijsku srednju vrijednost uzoraka kroz cijelu osminu sekunde (ukupno 689 uzoraka). Kasnije će se ove vrijednosti koristiti u izračunavanju konačnih koeficijenata. Ovim postupkom sačuvala se točnost, jer je umjesto jednog od 689 uzorka, iskorišteno svih 689 uzoraka a ujedno su se dobili novi uzorci koji će se koristiti za manju vremensku rezoluciju. Sa slika 51 i 52 može se primijetiti kako algoritam detektira ovaj ton u vremenskom intervalu [2 5.125] s. Za vrijeme od 2 s tablica pokazuje kako nije detektiran nijedan ton. Treba uzeti u obzir da je 2 s desna granica intervala koji se promatra tj. do 2 s nije detektiran nijedan ton. Zadnji interval na kojem se pojavljuje ton A3 je od 5 do 5.125 sekundi. Stvarni vremenski raspon tona je do 5.15 sekundi. Dobivena je pogreška od 0.025 sekundi. Jasno ne može se ni očekivati manja pogreška kada je vremensko razdoblje promatranja 0.125 sekundi. Provjerit će se rezultati i za ostale tonove.

Ton E3 pojavljuje se otprilike između 9.25 i 9.5 sekundi (slika 55). Detektirano vrijeme pojavljivanja tona E3, prema slici 53, je također između 9.25 i 9.5 sekundi. Ovo je primjer savršenog poklapanja stvarnog i detektiranog vremena sviranja tona, gdje zapravo i nema pogreške.




Slika 55 - Ton E3 u vremenskoj domeni




Slika 56 - Ton F3 u vremenskoj domeni

Suprotan primjer može se vidjeti kod tona F3 (slika 56), koji je odsviran od 9.6 pa otprilike do 9.8 sekundi (teško se procjenjuje jer se u isto vrijeme pojavljuje drugi ton). Pošto na vremenskoj osi kod rezultata ne postoji trenutak 9.6 detektirano je da se ton javlja od 9.5 sekundi. Ista stvar je i s desnom granicom gdje se umjesto 9.8 detektira 9.875 sekundi (slika 53).

Iz danih primjera može se vidjeti kako i kod vremenske osi rezultata dolazi do greške. Greška se uglavnom javlja zbog manje rezolucije. Za ovu primjenu greška od maksimalno 1/8 sekunde je prihvatljiva. Ukoliko ona ne zadovoljava kriterije, algoritam se može

podesiti tako da se uzme manji vremenski odsječak za promatranje signala. Zadnja granica do koje se s rezolucijom može ići je $44100^{-1}$ sekunde koliko dozvoljava ulazni signal. Sve ostalo iznad toga je stvar kontinuirane valićne transformacije tj. valića s kojim se radi CWT.

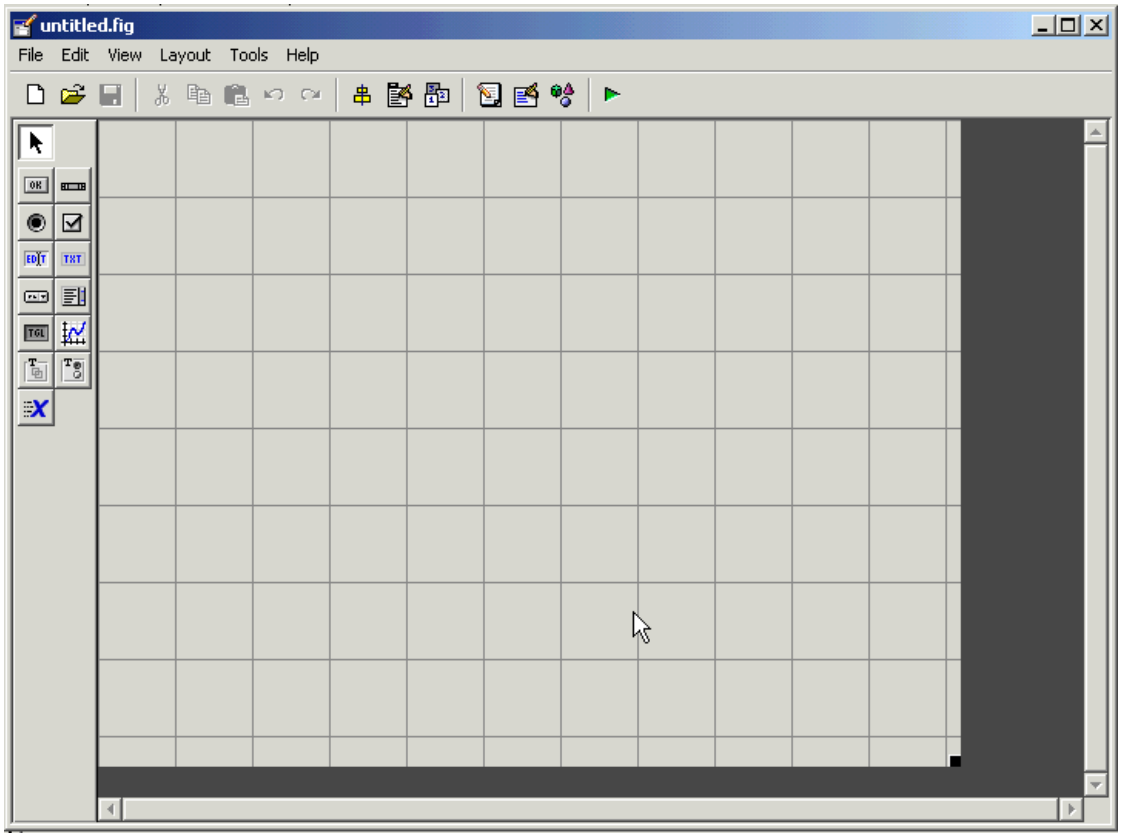
Može se zaključiti kako je maksimalna greška kod detektiranja tonova jedan poluton za frekvencijsku domenu, odnosno 1/8 sekunde za vremensku domenu. Ove greške prihvatljive su s obzirom na način (i okolnosti) detektiranja grešaka te za primjene ovog rada.

# 5. MATLAB GUIDE

Korisničko grafičko sučelje (eng. *graphical user interface*; GUI) dozvoljava korisniku da interaktivno provodi različite zadatke preko kontrola kao što su to gumbi i klizni izbornici. GUI se može koristiti tako da se pokreće iz MATLAB-a ili kao samostalna aplikacija.

GUI razvojno okruženje (eng. GUI *development enviroment*; GUIDE) osigurava alate za dizajniranje i programiranje GUI-a. Koristeći GUIDE editor za izgled (eng. GUIDE *Layout editor*), GUI se može i grafički uređivati. GUIDE tada automatski generira MATLAB kod koji definira komponente i uspostavlja okvir za GUI pozivne funkcije (funkcije koje se izvršavaju kada korisnik koristi GUI komponentu).

MATLAB GUIDE pokreće se klikom na ikonu u alatnoj traci ili upisivanjem `guide` u komandnom prozoru. Otvaranjem izbornika može se birati između nekoliko predefiniranih predloška a za početak je najbolje uzeti prazan GUI.

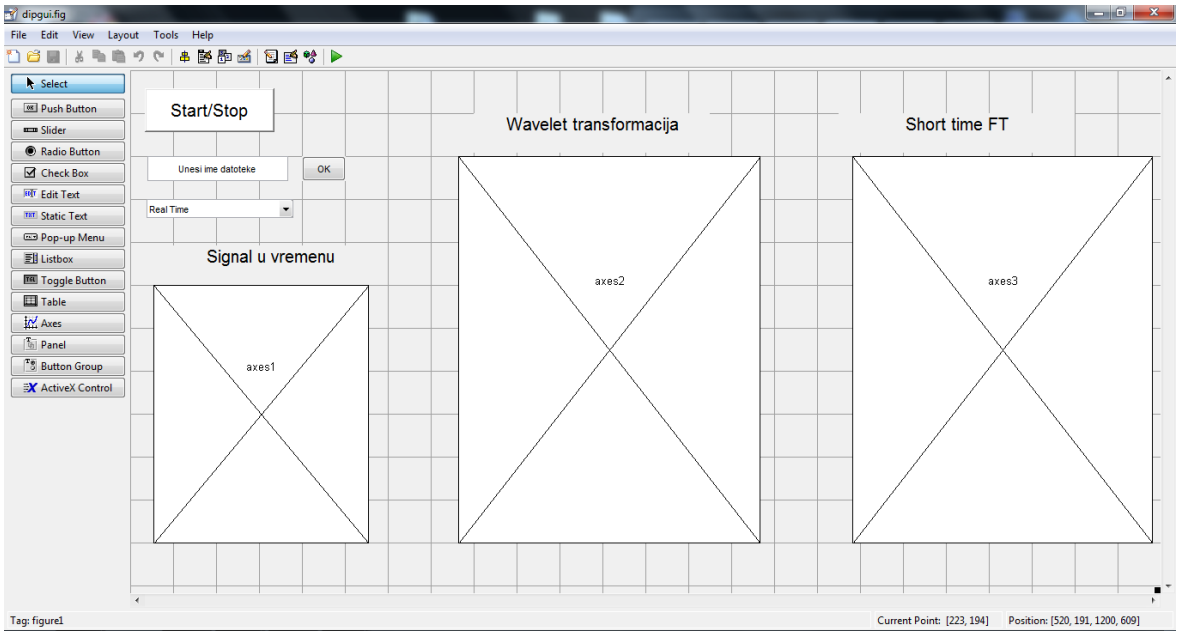



Slika 57 - Prazni GUI

Ovaj editor omogućuje jednostavno, grafičko dizajniranje izgleda GUI-a. Korisnik može graditi svoj GUI od komponenti koje su dostupne u paleti na lijevoj strani editora. Sljedeća slika pokazat će paletu sa svim komponentama i njihovim imenima.


Slika 58 - Komponente u GUI-i

GUI se dizajnira tako da se selektiraju kontrole iz palete lijevo, te se vuku i postavljaju na željenu poziciju. Svojstva svake komponente u GUI-u mogu se mijenjati tako da se otvori preglednik svojstava (eng. *property inspector*), koji se poziva ili desnim klikom na objekt ili iz izbornika "prikaz" (eng. *view*). Različite komponente imaju različita svojstva, no postoje neke generalne kao što je ime, pozicija u prostoru, veličina i tako dalje. Aplikacija koja se napravila tijekom ovog rada u GUIDE-u izgleda ovako:




Slika 59 - Izgled aplikacije u GUIDE-u

Pokretanje GUI-a provodi se tako što se klikne na zeleni trokut u alatnoj traci. Aktiviranjem GUI-a generira se M-datoteka aplikacije i fig datoteka te se prikaže "stvaran" izgled GUI-a. M-datoteka je stvorena od strane GUIDE-a kako bi implementirala grafičko sučelje. Ona sadrži automatski generirani kod uz prazan prostor rezerviran za nadopunjavanje koda koji će se pokretati kada su određene kontrole birane. Kada je GUI završen i pokrenut, a korisnik aktivira jednu od kontrola korisničkog sučelja kao što je gumb za stiskanje, MATLAB izvršava pozivnu funkciju aktivirane kontrole (eng. *callaback function*). Pomoću objektnog preglednika (eng. *object browser*), koji se nalazi u alatnoj traci pokraj ikone za pokretanje GUI-a, GUI komponente se mogu jednostavno upravljati te se mogu odrediti pozivne funkcije. Nakon što se prođu ovi koraci, GUI bi trebao biti potpuno funkcionalan i spreman za uporabu.

# Zaključak

U ovom radu analizirane su metode za estimaciju frekvencije osnovnog tona. Na primjerima se pokazalo kako se signal prebacuje iz vremenske u vremensko-frekvencijsku domenu. Usporedbom kratkotrajne Fourierove te kontinuirane valićne transformacije, došlo se do zaključka kako je za primjene kod zvučnih signala svakako prikladnija valićna transformacija. Prednost nad STFT-om dobila je zbog promjenjivih rezolucija s obzirom na porast frekvencije. Zbog ovog svojstva CWT je najprihvatljivija transformacija ljudskom uhu, te se stoga i koristila u nastavku rada.

Sljedeći problem je bio interpretacija te detekcija tonova kod transformiranih signala. Prve dvije metode bazirale su se na jednostavnoj detekciji pomoću amplituda CWT koeficijenata. Problemi kod prve metode bile su situacije ukoliko je signal preslab ili prejak te situacija kada se pojavljuje bijeli šum. Druga metoda dala je rješenje za prvi problem ali je ostala nemoćna kod bijelog šuma. Druge dvije metode temeljene su na postupku pronalaska Wienerove entropije. Ona prva radila je inverznu Wienerovu entropiju tona kroz cijeli vremenski odsječak. Uspješno je riješen problem bijelog šuma no ovisno o dužini trajanja tona postoji mogućnost da se neki ton predstavi i kao bijeli šum. Ovu manu riješio je posljednji algoritam koji je uspoređivao geometrijsku srednju vrijednost svih tonova zajedno s aritmetičkom srednjom vrijednosti pojedinog tona.

Konačno trebalo je odrediti postupak koji će odrediti koji tonovi su osnovni a koji su harmonici osnovnog tona. Predstavljene su metode koje traže osnovnu frekvenciju u vremenu te metode koje to rade u frekvenciji. Uspoređujući sve metode dolazi se do zaključka kako spektar harmonijskog produkta najviše zadovoljava zadane kriterije.

# Literatura


[1] TOMAC, I. *Detekcija modalnih parametara primjenom valne transformacije*. http://www.fesb.hr/Portals/0/docs/nastava/kvalifikacijski/KI-Tomac_ver2_2-FINAL.pdf, 4. 2012.

[2] THE MATHWORKS INC. *Continuous Wavelet Transform*. http://www.mathworks.com/help/toolbox/wavelet/gs/f3-1000759.html, 4.2012.

[3] CUTHBERT A. N. *Short Time Fourier Transform STFT*. http://cnyack.homestead.com/files/artran/stft2t1.htm, 5.2012.

[4] FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA. *L3. Fourierova transformacija na vremenskom otvoru – STFT*. http://nmdos.zesoi.fer.hr/laboratorij/pdf/03STFTPriprema.pdf, 6. 2012.

[5] ELEKTROTEHNIČKI FAKULTET BANJALUKA. *Multirezuluciona analiza*. http://dsp.etfbl.net/dip/predavanja/7multirezoluciona.pdf, 5.2012.

[6] WIKIPEDIA. *Spectral flatness*. http://en.wikipedia.org/wiki/Spectral_flatness, 6.2012.

[7] CORBEN C. *Harmonics.* http://users.lmi.net/corben/hrmncs.htm, 5. 2012.

[8] CENTER FOR COMPUTER RESEARCH IN MUSIC AND ACOUSTICS. *Pitch detection methods review*. https://ccrma.stanford.edu/~pdelac/154/m154paper.htm, 6.2012.

# Estimacija frekvencije osnovnog tona pomoću valićne i Fourierove transformacije

## Sažetak

U ovom diplomskom radu opisane su tehnike estimacije frekvencije osnovnog tona. Cijeli rad se podijelio na tri glavna dijela: vremensko-frekvencijsku transformaciju, detekciju tonova te estimaciju frekvencije osnovnog tona. Opisane su kontinuirana valićna te kratkotrajna Fourierova transformacija kao dvije glavne tehnike prebacivanja signala iz vremenske u vremensko-frekvencijsku ravninu. Nastavilo se s uspoređivanjem četiri metoda detekcije frekvencije. Opisane su glavne ideje tih metoda te njihove mane i prednosti. Zaključna faza rada je prolazak kroz tehnike estimacije frekvencije osnovnog tona u vremenskoj i frekvencijskoj domeni.

Osim dijela koji je vezan za obradu signala prikazan je i objašnjen rad sustava za implementaciju korisničkog grafičkog sučelja u MATLAB-u nazvan GUIDE.

## Ključne riječi

Valić, kontinuirana valićna transformacija, kratkotrajna Fourierova transformacija, CWT, STFT, skala, Wienerova entropija, harmonik, osnovna frekvencija, spektar

# Estimation of the fundamental frequency using Wavelet and Fourier transform

## Summary

This thesis described techniques for estimating frequency of the fundamental tone. Work has been divided into tree main segments: time-frequency transform, tone detection and estimating frequency of the fundamental tone. Continuous wavelet transform and short term Fourier transform were described as two main techniques of trensfering signals from time to time-frequency plane. It continued with comparison of four detection frequency methods. Main ideas, along with the advantages and disadvantages of these methods were described. Finishing part of this thesis were techniques of estimating frequency of the fundamental tone in time and frequency domain.

Besides the part associated with the signal processing, working principle of the user graphical interface implementation in MATLAB called GUIDE, was presented and explained.

## Keywords

Wavelet, continuos wavelet transform, short term Fourier transform, CWT, STFT, scale, Wiener entropy, harmonic, fundamental frquency, spectrum

# Skraćenice

| | | |
|---|---|---|
| CWT | *Continuous wavelet transform* | kontinuirana valićna transformacija |
| STFT | *Short term Fourier transform* | kratkotrajna Fourierova transformacija |
| AM | *Arithmetic mean* | aritmetička sredina |
| GM | *Geometric mean* | geometrijska sredina |
| MPEG | *Moving Pictures Experts Group* | skupina stručnjaka za pokretne slike |
| HPS | *Harmonic product spectrum* | spektar harmonijskih umnožaka |
| GUI | *Graphical user interface* | korisničko grafičko sučelje |
| GUIDE | *GUI development environment* | GUI razvojna okolina |
| MATLAB | *Matrix laboratory* | matrični laboratorij |

## Privitak

## MATLAB kod

```
function varargout = dipgui(varargin)
% DIPGUI MATLAB code for dipgui.fig
%      DIPGUI, by itself, creates a new DIPGUI or raises the existing
%      singleton*.
%
%      H = DIPGUI returns the handle to a new DIPGUI or the handle to
%      the existing singleton*.
%
%      DIPGUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in DIPGUI.M with the given input
arguments.
%
%      DIPGUI('Property','Value',...) creates a new DIPGUI or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before dipgui_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to dipgui_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help dipgui

% Last Modified by GUIDE v2.5 30-May-2012 12:13:51

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @dipgui_OpeningFcn, ...
                   'gui_OutputFcn',  @dipgui_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before dipgui is made visible.
function dipgui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
```

```
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to dipgui (see VARARGIN)


% fs predstavlja željenu frekvenciju otipkavanja
% ai će biti analogni ulaz s mikrofona
% analognom ulazu dodaje se jedan kanal
handles.fs=44100;
handles.ai=analoginput('winsound');
addchannel(handles.ai, 1);


% signal se uzorkuje željenom frekvencijom otipkavanja
% uzima se beskonačno podataka s kanala
% alocira se 512 bloka od 1024 uzoraka
% definira se varijbla buff koja će biti svojevrsni buffer
set(handles.ai, 'SampleRate',handles.fs);
set(handles.ai,'samplespertrigger',Inf);
set(handles.ai,'bufferingconfig',[1024 512]);
handles.buff=zeros(1,2048*10);

% gumb start/stop postavljamo aktivnim
% gumb ok postavljamo zasivljenim (neaktivnim)
% prostora z upis teksta se postavlja neaktivnim
set(handles.start_stop,'Enable','on');
set(handles.ok,'Enable','off');
set(handles.adresa,'Enable','off');

% Choose default command line output for dipgui
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes dipgui wait for user response (see UIRESUME)
% uiwait(handles.figure1)


% --- Outputs from this function are returned to the command line.
function varargout = dipgui_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in start_stop.
function start_stop_Callback(hObject, eventdata, handles)
% hObject    handle to start_stop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of start_stop
```

```
% pojavit će se dva tekst boxa a jedan će ostat nevidljiv
% crtat će se na objektu axes2
% proširit će se graf2 a smanjiti graf3
set(handles.text3,'Position', [119.8 41.230769 55 3.58461538]);
set(handles.text3, 'Visible', 'on');
set(handles.text4, 'Visible', 'off');
set(handles.text1, 'Visible', 'on');
axes(handles.axes2);
set(handles.axes2,'Position',[76.2 4.385 150 34.692]);
set(handles.axes3,'Position',[100 25 50 5]);

% uzima se vrijednost gumba start/stop
% dok god je u jedinici obavlja stvari u petlji
toggle=get(hObject,'Value');
while (toggle)

    % ukoliko analogni ulaz nije pokrenut pokreće se
    if ~isrunning(handles.ai)
        start(handles.ai)
    end



    % gleda se je li popupmenu aktiviran ili ne
    % ako je upaljen tj. 'on'
    % ugasi popupmenu1, text box i gumb ok
    status=get(handles.popupmenu1,'Enable');
    if strcmp(status,'on')
        set(handles.popupmenu1,'Enable','off');
        set(handles.adresa, 'Enable','off');
        set(handles.ok, 'Enable','off');
    end



    % gleda se jeli dvoljno uzoraka stiglo analognom ulazu
    % ukoliko ih je manje od 2048
    % vrti se u petlji dok ne dođe do 2048
    N = get(handles.ai,'samplesavailable');
    while N < 2048
        N = get(handles.ai,'samplesavailable');
    end

    % dohvaća se 2048 uzoraka
    % stavljaju se na prvo mjesto buff-a
    % a ostali uzorci se pomiču nazad s tim da se
    % najstarijih 2048 uzoraka odbacuje
    D = getdata(handles.ai,2048);
    handles.buff(1:2048*9)=handles.buff(2049:end);
    handles.buff(9*2048+1:end)=D;

    % odbacivanje trenutnih uzoraka koji su došli na ulaz
    flushdata(handles.ai);

    % iz buffera se direktno ispisuje na graf1
    axes(handles.axes1)
    plot(handles.buff);ylim([-2 2]);xlim([0 2048*10])

    % potrebno je poduzorkovat signal radi bržeg izvođenja
    % računa se period poduzorkovanog signala
```

```
    % računa se i vremenska os
    % računa se početna "frekvencija" a3 s kojom će se dobiti
    % kasnije prava frekvencija A3
    buf = handles.buff (1:32:end);
    Ts = 1/handles.fs*32;
    t = [0:length(buf)-Ts]' * Ts;
    a3 = scal2frq(1, 'cmor32-1', Ts) / 440;

    % izračun cijele oktave
    % pomoću oktava se dobivaju skale
    % pomoću kojih dobivamo tražene frekvencije
    step = 0.25;
    octave = a3 * 2.^([-9:step:2+(1/step-1)*step]/12);
    scales = [octave 2*octave 4*octave];
    frequencies = scal2frq(scales, 'cmor32-1', Ts);


    % računa se CWT transformacija vremenskog signala pomoću skala koje
smo
    % dobili
    % crtanje se izvodi na grafu2
    BUF = cwt(buf, scales, 'cmor32-1');
    axes(handles.axes2);
    h=imagesc(t, frequencies, abs(BUF));
    ylabel('Hertza'), xlabel('sekundi');

    % potrebno je okrenuti y os i postaviti je u logaritamskoj skali
    % rade se nove skale koje imaju manje uzoraka
    % te varijabla fre se dodatno obrće
    % te se postavlja za y os
    set(get(h, 'Parent'), 'YDir', 'normal')
    set(get(h, 'Parent'), 'YScale', 'log')
    oct = a3 * 2.^([-9:2]/12);
    fre =fliplr(scal2frq([oct 2*oct 4*oct], 'cmor32-1', Ts));
    set(get(h, 'Parent'), 'YTick', fre)


    grid
    grid minor

    % potrebno je pauzirati izvođenje kako bi se sve vidjelo
    % uzima se status aktivnosti gumba START/STOP da se zna jel treba
izaći
    % iz petlje
    pause (1/1000);
    toggle=get(hObject,'Value');
end

% kada se izađe iz petlje tj kaka je pauzirano izvođenje omogućava se
% interakcija s popupmenuom
set(handles.popupmenu1,'Enable', 'on');



% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu1

% uzimaju se 'string' i 'value' radi usporedbe
str=get(hObject, 'String');
val=get(hObject, 'Value');

% ukoliko je trenutno odabran 'realtime'
% omogući gumb start/stop
% onemogući gumb ok i polje za upload
% ukoliko je odabrana opcija 'Uploadaj snimljeno'
% onemogući gumb start/stop a omogući gumb ok i polje za upload
switch str{val}
    case 'Real Time'
        set(handles.start_stop,'Enable','on');
        set(handles.ok,'Enable','off');
        set(handles.adresa,'Enable','off');


    case 'Uploadaj snimljeno'
        set(handles.start_stop,'Enable','off');
        set(handles.ok,'Enable','on');
        set(handles.adresa,'Enable','on');
    end


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function adresa_Callback(hObject, eventdata, handles)
% hObject    handle to adresa (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% Hints: get(hObject,'String') returns contents of adresa as text
%        str2double(get(hObject,'String')) returns contents of adresa as
a double


% --- Executes during object creation, after setting all properties.
```

```
function adresa_CreateFcn(hObject, eventdata, handles)
% hObject    handle to adresa (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in ok.
function ok_Callback(hObject, eventdata, handles)
% hObject    handle to ok (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% postavljaju se pozicije grafova 2 i 3
set(handles.axes2,'Position',[76.2 4.385 70 34.692]);
set(handles.axes3,'Position',[168 4.385 70 34.692]);

% postavljaju se pozicije tekst boxova
set(handles.text3,'Position', [79.8 39.384615 55 3.58461538]);
set(handles.text3, 'Visible', 'on');
set(handles.text4, 'Visible', 'on');
set(handles.text1, 'Visible', 'on');

% uzima se polje znakova sa polja za unos teksta
% učitava se datoteka s tim imenom u trenutnoj mapi
str=get(handles.adresa,'String');
[x, fs] = wavread(str);
% za svaki slučaj povećat će se amplitude signala
x=x*10;


% uzima se lijevi stupac signala koji se poduzorkuje radi brzine
% računaju se period
% i vremenska os
% a sve to iscrtava se na prvom grafu
x = x(1:8:length(x),1);
T = 1/fs*8;
t = [0:length(x)-T]' * T;
axes(handles.axes1);
plot(t,x);

% slika valića kojeg koristimo: Morletov wavelet
[psi, xval] = wavefun('cmor32-1');

% računa se početna "frekvencija" a3 s kojom će se dobiti
% kasnije prava frekvencija A3
a3 = scal2frq(1, 'cmor32-1', T) / 440;

%              F#,  F,  E, D#,  D, C#,  C,  B, A#,  A, G#, G
%        2.^([ -9, -8, -7, -6, -5, -4, -3, -2, -1, 0 , 1 , 2 ]/12) *
frek_a
```

```
% izračun cijele oktave
% pomoću oktava se dobivaju skale
% pomoću kojih dobivamo tražene frekvencije
step = 0.25;
octave = a3 * 2.^([-9:step:2+(1/step-1)*step]/12);   % Octave=3
scales = [octave 2*octave 4*octave]; % scales for 3 octaves
frequencies = scal2frq(scales, 'cmor32-1', T); % frequencies for the
above scales

% računa se CWT transformacija vremenskog signala pomoću skala koje smo
% dobili
% crtanje se izvodi na grafu3
X = cwt(x, scales, 'cmor32-1');
axes(handles.axes2);
h=imagesc(t, frequencies, abs(X));
ylabel('Hertza'), xlabel('sekundi')

% potrebno je okrenuti y os i postaviti je u logaritamskoj skali
% rade se nove skale koje imaju manje uzoraka
% te varijabla fre se dodatno obrće
% te se postavlja za y os
set(get(h, 'Parent'), 'YDir', 'normal')
set(get(h, 'Parent'), 'YScale', 'log')
oct = a3 * 2.^([-9:2]/12);
fre =fliplr(scal2frq([oct 2*oct 4*oct], 'cmor32-1', T));
set(get(h, 'Parent'), 'YTick', fre)



grid
grid minor

% tražimo AM_ i GM_. to su vrijednosti
% svih tonova zasebno za svaku osminu sekunde.
% dakle računa se aritmeticka sredina i geometrijska sredina za svaku
% osminu sekunde
osmina=round(0.125/T);
brojosmina=floor(length(X)/osmina);
AM_=zeros(144,brojosmina);
GM_=ones(144,brojosmina);
k=1;
for i=1:brojosmina*osmina
    if mod(i,osmina)~=0
        AM_(:,k)=abs(X(:,i))+AM_(:,k);
        GM_(:,k)=nthroot(abs(X(:,i)),689).*GM_(:,k);
    else
        AM_(:,k)=(abs(X(:,i))+AM_(:,k))/osmina;
        GM_(:,k)=nthroot(abs(X(:,i)),689).*GM_(:,k);
        k=k+1;
    end
end

% do sada postoji sritmetička i geometrijska srednja vrijednost svakog
tona
% u osmini sekunde. dakle osima sekunde je jedan 'trenutak'. imamo 144
% frekvencije a trebaju nam samo 36. dakle uzimamo samo one frekvencije
% koje nam pašu

AM=zeros(36,brojosmina);
GM=ones(1,brojosmina);
```

```
redak=35;
AM(36,:)=(AM_(1,:));
for i=5:4:141
    AM(redak,:)=AM_(i,:);
    redak=redak-1;
end

% za koeficijente će se uzeti omjer aritmetičke sredine tona u tom
trenutku
% i geometrijske sredine svih tonova u tom trenutku

GM=nthroot(prod(GM_(:,:)),144);

koeficijent=ones(36,brojosmina);
for i=1:36
    koeficijent(i,:)=AM(i,:)./GM;
end

% slijedi izbacivanje koeficijenata koji nam ne trebaju
k2=zeros(36,brojosmina);
for i=1:36
    for k=1:brojosmina
        % prvo će se riješiti prvi i zadnji redak koji ne smiju biti Inf
i
        % Nan te prvi redak ne smije imat veći član ispod njega a zadnji
ne
        % smije imat veći član iznad njega
        if (i==1)
            if ((koeficijent(i,k)>koeficijent(i+1,k))&...
                    (koeficijent(i,k)<Inf))
                k2(i,k)=koeficijent(i,k);
            end
        elseif (i==36)
            if ((koeficijent(i,k)>koeficijent(i-1,k))&...
                    (koeficijent(i,k)<Inf))
                k2(i,k)=koeficijent(i,k);
            end
        else

        % ovaj if riješava Inf i NaN,
        % ako iznad i ispod tona nema većeg tona onda dolazi u obzir
            if ((koeficijent(i,k)<Inf)&...
                    (koeficijent(i,k)>koeficijent(i+1,k))&...
                    (koeficijent(i,k)>koeficijent(i-1,k)))
                k2(i,k)=koeficijent(i,k);
            end
        end
    end
end

% slijedi brisanje harmonika i onih tonova koji su manji od nekog praga
l=0;
maksimum=max(max(k2));
for i=1:36
    for k=1:brojosmina
        % ukoliko je broj u prvoj oktavi gleda se je li opće veći od 15%
        % najveće vrijednosti, ako je znači da mu odma brišemo harmonike
a
        % ako nije onda njega brišemo
```

```
        if (i<13)
            if (k2(i,k)<0.15*maksimum)
                k2(i,k)=0;
            else
                k2(i+12,k)=0;
                k2(i+24,k)=0;
            end
        elseif (i<25)
            if (k2(i,k)<0.15*maksimum)
                k2(i,k)=0;
            else
                k2(i+12,k)=0;
            end
        else
            if (k2(i,k)<0.15*maksimum)
                k2(i,k)=0;
            end
        end
    end
    if max(k2(i,:)~=0)
           l=l+1;
    end
end

% ispis
% gleda se koji tonovi su uoće prisutni pa ih se ispisuje
ton=cell(1,l);
podaci=zeros(l,brojosmina);
l=1;
for i=1:36
    if max(k2(i,:)~=0)
        podaci(l,:)=k2(i,:);
        switch i
            case 1
                ton{1,l}='G0';
            case 2
                ton{1,l}='G#0';
            case 3
                ton{1,l}='A1';
            case 4
                ton{1,l}='A#1';
            case 5
                ton{1,l}='B1';
            case 6
                ton{1,l}='C1';
            case 7
                ton{1,l}='C#1';
            case 8
                ton{1,l}='D1';
            case 9
                ton{1,l}='D#1';
            case 10
                ton{1,l}='E1';
            case 11
                ton{1,l}='F1';
            case 12
                ton{1,l}='F#1';
            case 13
                ton{1,l}='G1';
            case 14
                ton{1,l}='G#1';
```

```
            case 15
                ton{1,l}='A2';
            case 16
                ton{1,l}='A#2';
            case 17
                ton{1,l}='B2';
            case 18
                ton{1,l}='C2';
            case 19
                ton{1,l}='C#2';
            case 20
                ton{1,l}='D2';
            case 21
                ton{1,l}='D#2';
            case 22
                ton{1,l}='E2';
            case 23
                ton{1,l}='F2';
            case 24
                ton{1,l}='F#2';
            case 25
                ton{1,l}='G2';
            case 26
                ton{1,l}='G#2';
            case 27
                ton{1,l}='A3';
            case 28
                ton{1,l}='A#3';
            case 29
                ton{1,l}='B3';
            case 30
                ton{1,l}='C3';
            case 31
                ton{1,l}='C#3';
            case 32
                ton{1,l}='D3';
            case 33
                ton{1,l}='D#3';
            case 34
                ton{1,l}='E3';
            case 35
                ton{1,l}='F3';
            case 36
                ton{1,l}='F#3';
        end
        l=l+1;
    end
end

f = figure('Position', [200 200 1000 500]);
t = uitable('Parent', f, 'Position', [1 1 1000 500]);
set(t,'Data', podaci);
set(t, 'ColumnName',{[1:brojosmina]/8 'sekunde'});
set(t, 'RowName',ton);

%%
% STFT

x = x(1:4:length(x),1);
T = 1/fs*32;
t = [0:length(x)-T]' * T;
```

```
% gaussov vremenski prozor
% vremenske granice su takve da dobijemo 256 uzoraka vremenskog otvora
% radi brzeg izvodjenja FFTa
tg=[-0.0929:T:0.0929-T];
wg=exp(-tg.^2/2*128);

% signal se pretvara u redak
% len označa duljinu signala
x = x(:)';
len = length(x);

% vremenski otvor također se pretvara u redak;
% lenw će označavati duljinu vremenskog otvora
wg = wg(:)';
lenw = length(wg);

% signal će se nadopuniti nulama s jede i druge strane
% kako bi se što preciznije radio FFT
% njegova duljina je duljina signala + duljina otvora -1
% inicijalizira se varijabla X koja će biti FFT(x)
x = [zeros(1, lenw-1), x, zeros(1, lenw-1)];
ls = len+lenw-1;
X = complex(zeros(lenw, ls), zeros(lenw, ls));


% potrebno je konjugirati vremenski otvor pa ukoliko ima irealnih članova
% oni će se konjugirati
if (~isreal(wg))
    wg = conj(wg);
end

% korektivni faktor uslijed proširenja signala nulama
neg_shift = floor((lenw-1)/2);

% množenje signala s otvorom pa se taj dio transformira FFT-om
% sve se sprema u varijablu X te se za svaki pomak prozora računa FFT
for i=1:ls
    x1 = x(i:i+lenw-1) .* wg;
    X(:, i) = (fft(ifftshift(x1)).* exp(-j*2*pi/lenw *([0:lenw-1]) * ((i-
1)-neg_shift))).';
end
k = ([1:ls]-1)-neg_shift;


% 2D vizualizacija STFT
% računaju se frekvencije
% nalaze se samo one koje su između 100 i 700
% iscrtava se sve na grafu3
lenWG = length(wg);
f = [-ceil(lenWG/2):ceil(lenWG/2)-1]/lenWG * 1/T;
ind = find(f<700 & f> 100);
f1 = f(ind);
X1 = abs(fftshift(T*X,1));
X1 = X1(ind,:);
tau = k*T;
axes(handles.axes3); %colorbar;
h = imagesc(tau, f1, X1); ylabel('Hertza'), xlabel('sekundi'), set(get(h,
'Parent'), 'YDir', 'normal')
```

```
grid
grid minor
```