Primjena algoritama za detekciju komponenata rubova spektrograma

Turković, Erik

Sveučilište Jurja Dobrile u Puli

Tehnički Fakultet Pula

Erik Turković

Primjena algoritama za detekciju rubova komponenti spektrograma

Završni rad

Pula, 2022.

Sveučilište Jurja Dobrile u Puli

Tehnički Fakultet Pula

Erik Turković

Primjena algoritama za detekciju rubova komponenti spektrograma

Završni rad

JMBAG: 0303090434

Studijski smjer: Računarstvo

Predmet: Signali i sustavi

Znanstveno područje: Tehničke znanosti

Znanstveno polje: Računarstvo

Znanstvena grana: Obrada informacija

Mentorica: Doc. dr. sc. Nicoletta Saulig, dipl. ing.

Pula, 2022.

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani_____, kandidat za prvostupnika _____ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujemda niti jedan dio Završnog rada nije napisan na nedozvoljeni način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student _____

U Puli, _____

IZJAVA O KORIŠTENJU AUTORSKOG DJELA

Ja,_____dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj Završni rad pod nazivom

_____

_____

koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganjejavnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, _____

Potpis

Sadržaj

1.Uvod 1
1.1 Spektrogram 2
1.2 MatLab 4
2 Kod generiranja signala 4
2.2 Odabrani signali za primjenu algoritama 6
2.3 Algoritmi za detekciju rubova 8
2.3.1 Primjena Sobel algoritma na signal sinusoidnom modulacijom 9
2.3.2 Primjena Log algoritma na signal sinusoidnom modulacijom 10
2.3.3 Primjena Zerocross algoritma na signal sinusoidnom modulacijom 11
2.3.4 Primjena Prewitt algoritma na signal sinusoidnom modulacijom 12
2.3.5 Primjena Roberts algoritma na signal sinusoidnom modulacijom 13
2.3.6 Primjena Canny algoritma na signal sinusoidnom modulacijom 14
2.3.7 Primjena sobel algoritma na signal paraboličnom modulacijom 15
2.3.8 Primjena Log algoritma na signal paraboličnom modulacijom 16
2.3.9 Primjena Zerocross algoritma na signal paraboličnom modulacijom 17
2.3.10 Primjena Prewitt algoritma na signal paraboličnom modulacijom 18
2.3.11 Primjena Roberts algoritma na signal paraboličnom modulacijom 19
2.3.12 Primjena Canny algoritma na signal paraboličnom modulacijom 20
3. Završni dio 21
4 Popis slika 22
5. Izvori 24

Sažetak

Zadatak ovog završnog rada je generirati i obraditi različite signale preko MatLab-a. U uvodu je opis različitih pojmova koji su potrebni za razumjeti rad. Glavni i završni dio se sastoje on tehničkih detalja. Opisan postupak generiranja različitih signala, njihov kod u MatLabu, vizualizacija signala kod obrađivanja i vizualizacija signala nakon primjene algoritama za detekciju rubova. Testirano je ukupno šest različitih algoritama za detekciju rubova i generirano je četiri različita signala. Algoritmi za detekciju rubova su primijenjeni na signale bez šuma i na signale sa šumom koji imaju tri razine šuma

Ključne riječi:

Spektrogram, signali, detekcija rubova, MatLab, SNR, šum

Abstract

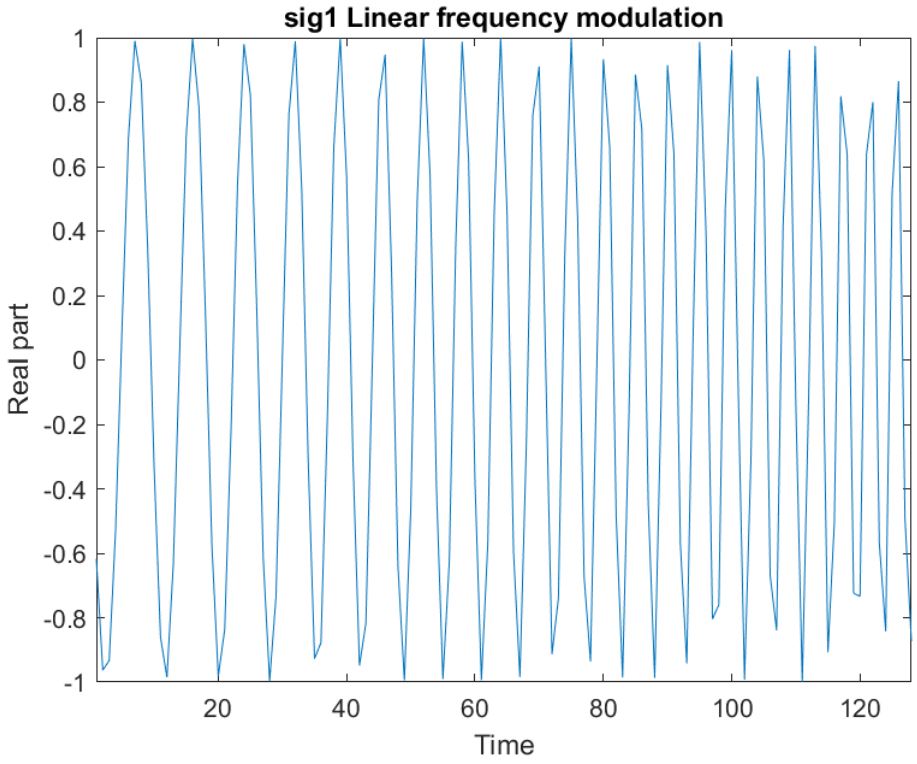
The task of this final paper was to generate and process different signals with MatLab. In the introduction we se the description of different terms that are needed to understand this paper. The main and final part are consist of technical details. It describes the proces of generating different signals, their code in MatLab, visualization before and after the application of edge detecting algorithms. Six different algorithms were tested on four different signals. The algorithms where applied on signals without noise and with noise ranging beetween 3 noise levels.

Key words:

Spectrogram, signals, edge detection, MatLab, SNR, noise

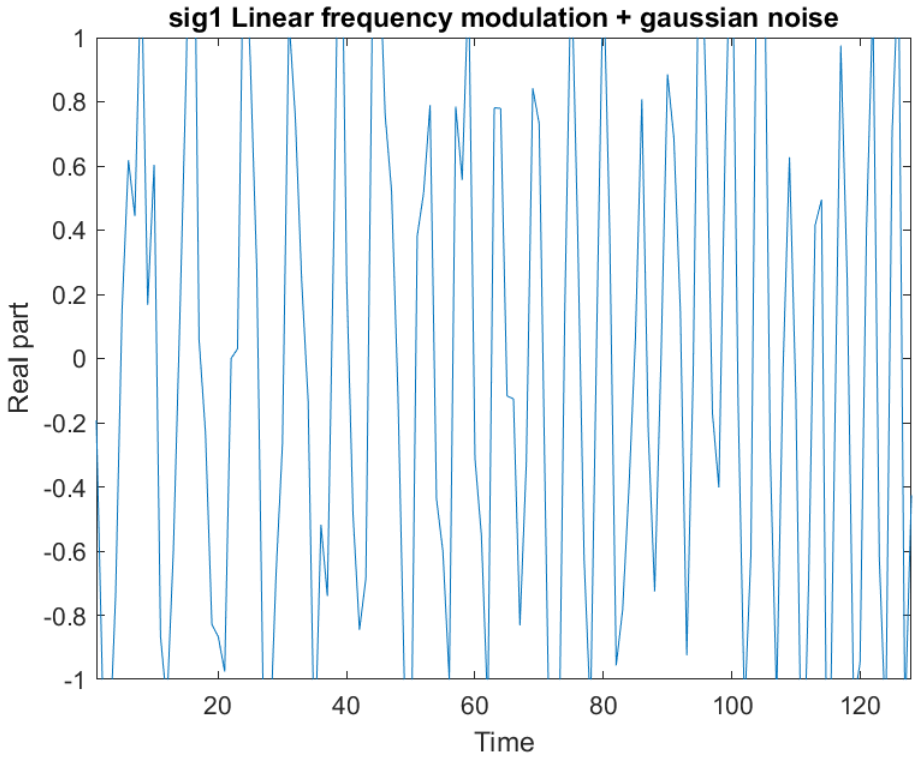
1.Uvod

Prvo moramo razumjeti što su signali. Iz naše okoline konstantno primamo različite signale, koje možemo mjeriti. Ako je u pitanju zvuk, snimamo ga mikrofonom. Mikrofon je senzor, odnosno mjerni pretvornik. Mjerni pretvornici su uređaji koji pretvaraju različite signale iz okoline u električni signal. Električni signal je naponska ili strujna funkcija vremena pogodna za daljnju obradu. Ovisno o vizualizaciji signala, možemo dobiti različite rezultate i izvući različite podatke. U sljedećoj slici vidimo jedan jednostavan signal prikazan u vremenskoj domeni.



Slika 1 *linearni signal, generiran u MatLab-u*

Osim prikaza signala u vremenu, postoje i prikazi signala u frekvencijskoj domeni (spektar signala) te prikazi signala u vremenu i frekvenciji (Wigner-Ville-ova distribucija, Spektrogrami itd). Snimanje signala nije uvijek savršeno, postoje šumovi koji nam degradiraju kvalitetu promatranog signala. Na sljedećoj slici je linearni signal (Slika jedan) s aditivnim šumom.



Slika 2. *linearni signal sa šumom, generiran u MatLab-u gaussovim šumom*

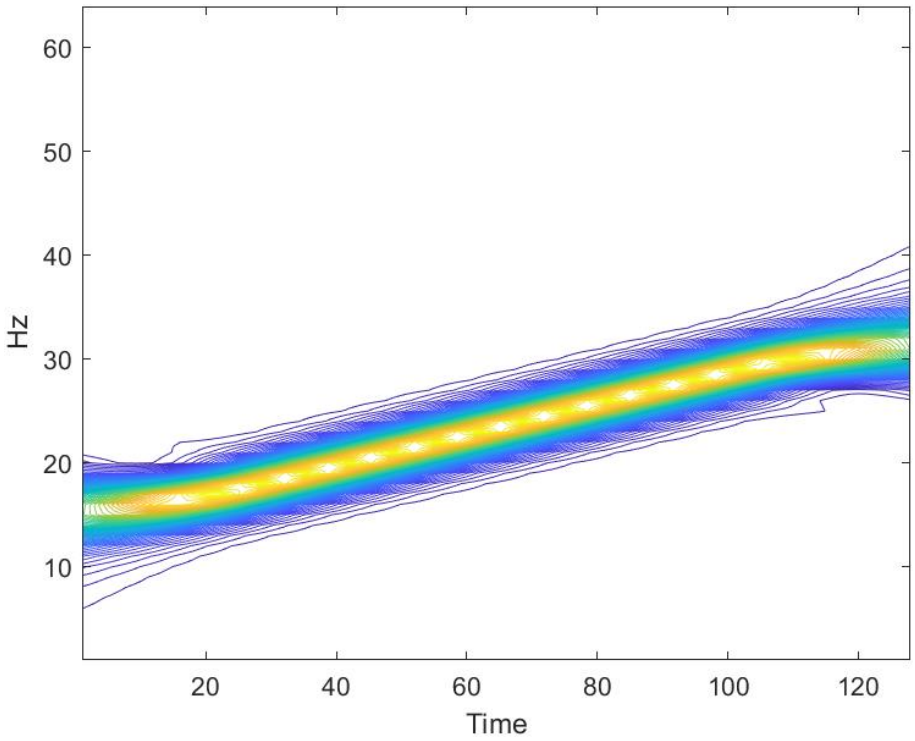
Vidimo kako je signal izobličen šumom. Neki prikazi bolje a neki lošije prikazuju signale sa šumom, a to utječe na očuvanje rubova signala kod primjene algoritama za detekciju rubova komponenata na vremensko-frekvencijskoj distribuciji signala.

1.1 Spektrogram

$$S_x(t,\nu) = \left| \int_{-\infty}^{+\infty} x(u)\, h^*(u-t)\, e^{-j2\pi\nu u}\, du \right|^2$$

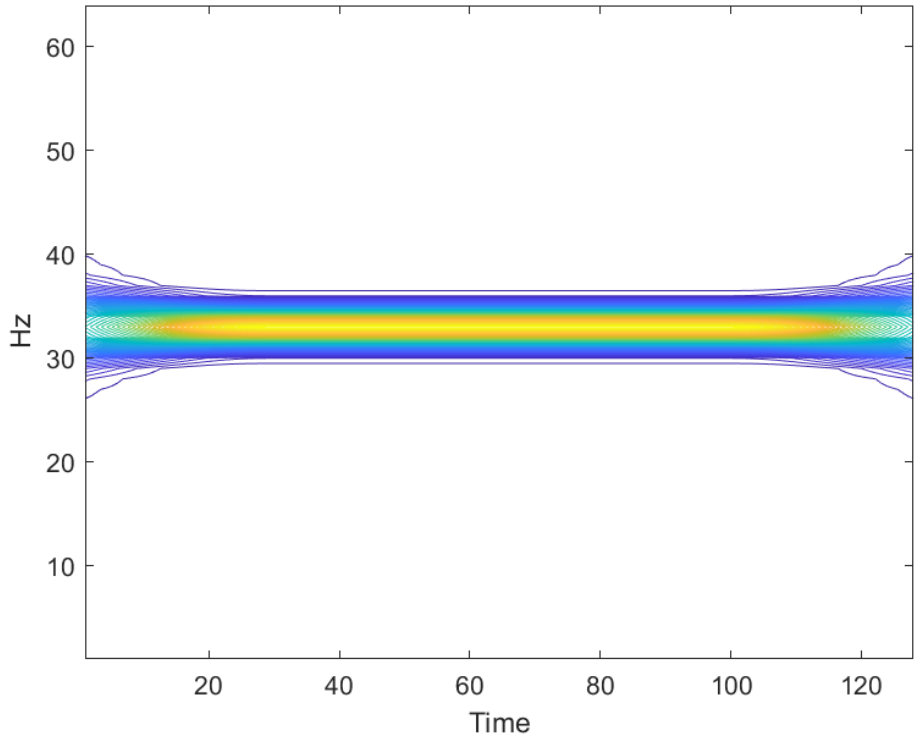
Slika 3*. formula funkcije spektrograma korištene u radu, iz Time-Frequency Toolbox, [7]*

Spektrogram je jedna od vremensko-frekvencijskih distribucija signala. Jednak je kvadratu STFT-a (Short-Time Fourier Transform). U formuli imamo: $x(u)$ je signal, $h^*(u-t)$ je prozor signala centrirani oko vremena t a h je veličina prozora. Time dobimo $x(u)h^*(u-t)$ nad čime napravimo furierovu transformaciju da dobimo STFT i onda je rezultat kvadratne veličine STFT-a spektrogram(Slika 3).Pokazuje nam istovremeno tri veličine. Jedna os označava vrijeme, isto kao u Slikama jedan, dva i tri, ali spektrogram nam istovremeno prikazuje frekvencijski sadržaj signala kroz vrijeme.



***Slika 4**. Spektrogram linearnog signala, generiran u MatLab -u*

Iz spektrograma prvog signala(Slika 4) vidimo da je frekvencijska modulacija linearna, tj. da linearno raste, početna frekvencija mu je između deset i dvadeset a na kraju je otprilike trideset Hz.



***Slika 5**. Spektrogram konstantnog signala, generiran u MatLab -u*

Iz ove slike možemo zaključiti da je frekvencijska modulacija signala konstantna, frekvencija se ne mijenja od dvadeset Hz.

1.2 MatLab

Matlab je alat koje je korišten u ovom završnom radu za generiranje i obrađivanje signala te primjenu algoritama za detekciju rubova. Alat ima svoj programski jezik i služi nam da bi analizirali različite podatke. Za različite namjene su nam potrebni različiti paketi zvani ToolBoxes. Za ovaj rad su uglavnom korištena 2 dodatna paketa za MatLab. Jedan je Signal Proccessing Toolbox koji je dio MatLaba i Time-Frequency Toolbox. Licenca za Time-Frequency Toolbox je na stranici nakon naslova kao što je navedeno u uputama za korištenje toolboxa i dodavanja licence.

2 Kod generiranja signala

Sada prelazimo na izradu koda za generiranje signala. Nazivi signala na slikama i u kodu dodijeljeni su redosljedom, sig1 je linearni, sig2 je sinusoidni, sig3 je parabolični a sig4 konstantni. Proći ćemo kroz kod za generiranje konstantnog i linearnog signala, spektrograma i dodavanja šuma. Kod za generiranje signala linearnom modulacijom je :

```
N=128;
sig1=fmlin(N,0.1,0.250);
plot(real(sig1))
title('sig1 Linear frequency modulation');
axis([1 N -1 1]);
xlabel('Time'); ylabel('Real part');
```

N nam predstavlja broj točaka ili vrijeme. 0.1 u fmlin je inicijalna normirana frekvencija a 0.25 nam je finalna normirana frekvencija. Slijedi ispis signala s dodatnim postavkama kao što je naslov, pa osi i nazivi za x i y os. Sljedeći kod generira i ispisuje signal s konstantnom frekvencijskom modulacijom:

```
sig4=fmconst(N,0.20,N);
plot(real(sig4))
```

N je isti kao i u prvom signalu, a 0.20 predstavlja normiranu frekvenciju, ista je za cijelo trajanje signala. Drugi N u funkciji je središte vremena kojem je zadana vrijednost N/2. Na prvi signal smo dodali i šum.
Šum je generiran ovim kodom:

```
gnsig1=sigmerge(sig1,noisecg(N),SNR);
plot(real(gnsig1))
title('sig1 Linear frequency modulation + gaussian noise');
axis([1 N -1 1]);
xlabel('Time'); ylabel('Real part');
```

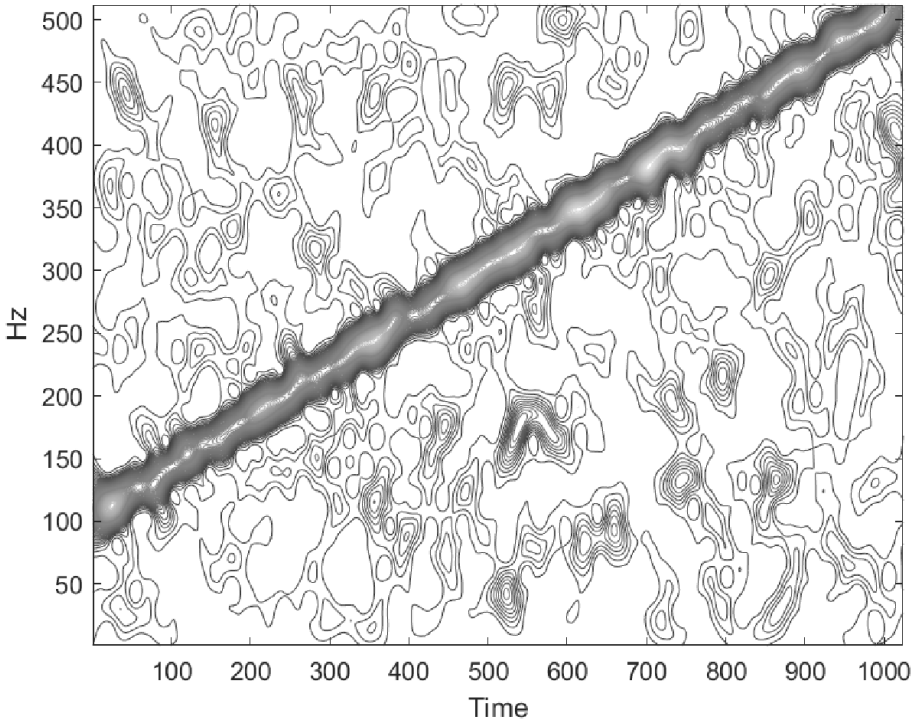
Vrlo je jednostavno primijeniti šum na signale. Napravimo novi signal, ovdje je definiran sa gnsig1 što stoji za gaussian noise signal jedan. Pozivamo sigmerge() u kojem imamo sig1 što je signal koji smo prvo generirali te dodajemo zvuk sa parametrom N koji mora biti isti kao i za signal. Na kraju odaberemo vrijednost za SNR. SNR je Signal to Noise Ratio i određuje nam jačinu šuma. Što je manji broj to je jači šum. U primjenama detekcije rubova ćemo vidjeti različiti jačine šumova. Sljedeći korak je realizirati spektrogram signala. Kod za spektrogram je :

```
h=tftb_window(61, 'hanning');
[ssig1]=tfrsp((sig1),1:N,N,h);
figure(1)
t=1:1:N;
f=1:1:N;
contour(t,f,ssig1,52 )
title('sig1 Spectrogram');
xlabel('Time'); ylabel('Hz');
```

U kodu tftb_window definira veličinu prozora spektrograma. Spektrogram signala sig1 označen je sa ssig1(spektrogram signal jedan), 1:N je vremenski korak, a N frekvencijski. Parametar h je veličina veličina prozora koju smo definirali sa tftb_window. Contour nam stvori matricu. Boje matrice spektrograma je potrebno pretvoriti u crno bijele jer su algoritmi namijenjeni da rade sa takvim bojama. Kod za to je :

```
I =rgb2gray(imread("C:\Users\Korisnik\Desktop\Testiranje\sgnsig1/10sgnsig1.png"));
J =rgb2gray(imread("C:\Users\Korisnik\Desktop\Testiranje\ssig1/ssig1.png"));
imshow(I);
title("Gray Scale Image");
```

Pohranimo matrice spektrograma i onda ih pomoću imread() pretvorimo u rgb2gray da bi postale crno bijele. Koristimo imshow() da vizualiziramo matricu. Jedino što preostaje je primijeniti algoritme za detekciju rubova na neke signale. Primjer GrayScale spektrograma u sljedećoj slici:



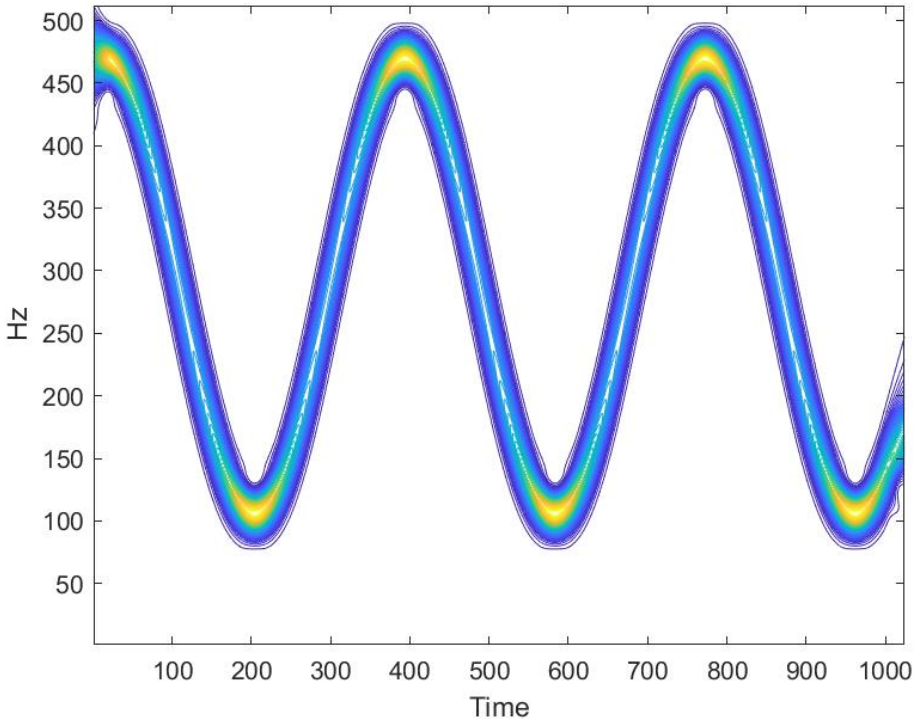
Slika 6. *GrayScale Spektrogram sa šumom linearnog signala, generiran u MatLab-u*

2.2 Odabrani signali za primjenu algoritama

Algoritmi će biti primijenjeni na dva signala različite modulacije. Prva dva pokazana signala, linearni i konstantni, služili su za primjer signala, dodavanja šuma i stvaranja spektrograma. Algoritme ćemo primijeniti na dvije druge vrste signala. Jedan je sinusoidni a drugi je paraboličan, to su kompleksniji signali pa će primjena algoritama izgledati zanimljivije. Jedino što se mijenja je generiranje signala jer svaka modulacija ima drugačiji kod. Procesiranje signala, primjenjivanje šuma, generiranje spektrograma, rezanje i pretvaranje u crno bijelu je isto kao i na primjerima za linearni i konstantni signal. Prvo ćemo pokazati generaciju signala sa sinusoidnom frekvencijskom modulacijom.

```
sig2=fmsin(N, 0.1,0.46,0.37*N,35,0.45,-1.0);
plot(real(sig2))
title('sinusoidal frequency modulation sig2');
```

Napravimo novi signal i odaberemo parametre u fmsin(), N je isti kao i u svakom, predstavlja nam broj točaka ili vrijeme, onda 0.1 je najmanja normalizirana frekvencija a 0.46 najveća. Nakon toga je period, uvijek je N, ali ovdje je pomnožen sa 0.37. 35 je time reference, 0.45 normalizirana frekvencija a -1 je smjer(može biti ili 1 ili -1). Slika signala bez šuma:

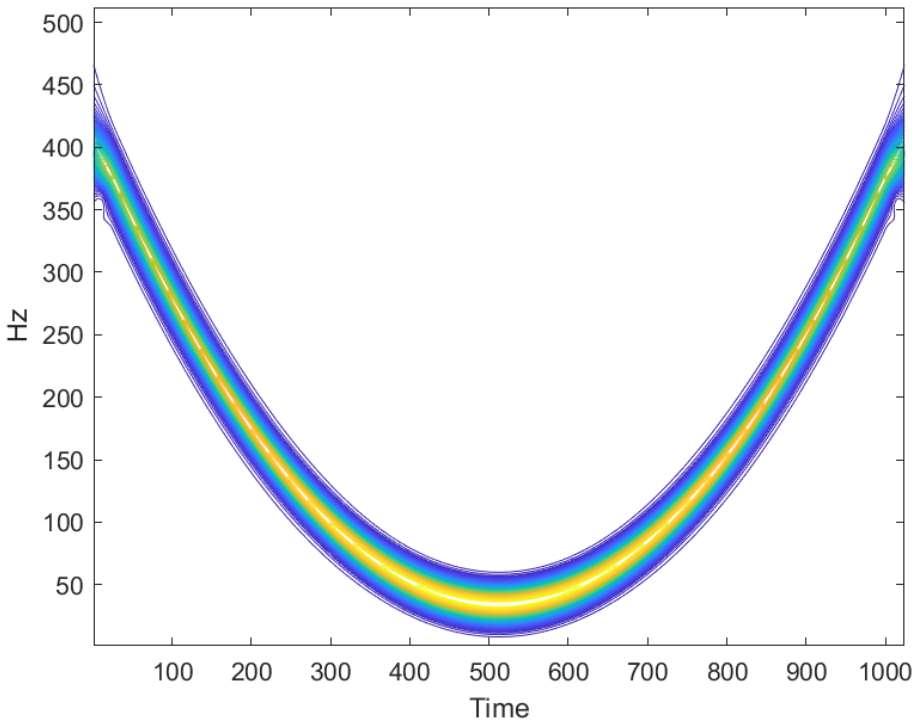


Slika 7. *Spektrogram bez šuma sinusoidnog signala, generiran u MatLab-u*

Drugi signal nad kojim ćemo primijeniti algoritme je u obliku parabole. Kod je :

```
[sig3]=fmpar(N,[1 0.4],[400 0.05],[N 0.4]);
plot(real(sig3))
```

Isto napravimo novi signal i stavimo odgovarajuću funkciju u ovom slučaju fmpar(). U fmpar() isto imamo N za vrijeme ili broj točaka, nakon N u uglati zagradama imamo vektore i time je definirani signal paraboličnom modulacijom. Slika spektrograma signala bez šuma:



Slika 8*. Spektrogram bez šuma paraboličnog signala, generiran u MatLab-u*

2.3 Algoritmi za detekciju rubova

Imamo 6 algoritama, poziv za algoritme je jednostavan. Stvorimo novu sliku i primijenimo algoritam funkcijom edge(). U edge učitamo matricu, vrstu algoritma i dodatne parametre. Koristit ćemo 6 algoritama, Sobel, Prewitt, Roberts, log, zerocross i canny. U sljedećoj tablici su algoritmi sa kratkim opisom izvađeni iz stranice MathWorksa.

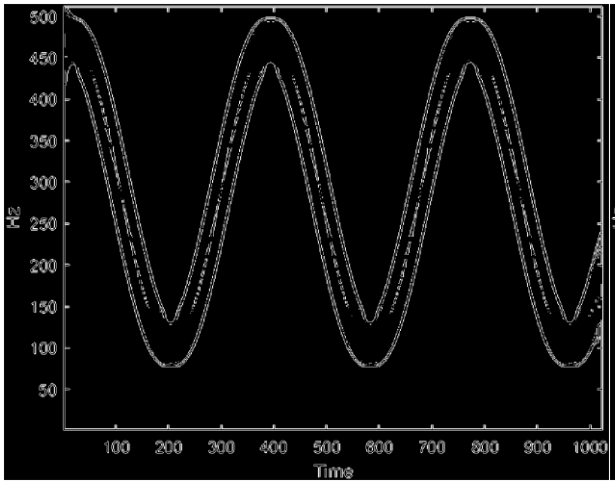
1	Method	Description
2	"Sobel"	Finds edges at those points where the gradient of the image I is maximum, using the Sobel approximation to the derivative.
3	"Prewitt"	Finds edges at those points where the gradient of I is maximum, using the Prewitt approximation to the derivative.
4	"Roberts"	Finds edges at those points where the gradient of I is maximum, using the Roberts approximation to the derivative.
5	"log"	Finds edges by looking for zero-crossings after filtering I with a Laplacian of Gaussian (LoG) filter.
6	"zerocross"	Finds edges by looking for zero-crossings after filtering I with a filter that you specify, h.
7	"Canny"	Finds edges by looking for local maxima of the gradient of I. The edge function calculates the gradient using the derivative of a Gaussian filter. This method uses two thresholds to detect strong and weak edges, including weak edges in the output if they are connected to strong edges. By using two thresholds, the Canny method is less likely than the other methods to be fooled by noise, and more likely to detect true weak edges.

Tablica 1. Metode detekcije rubova, MathWorks, Edge
(https://www.mathworks.com/help/images/ref/edge.html)

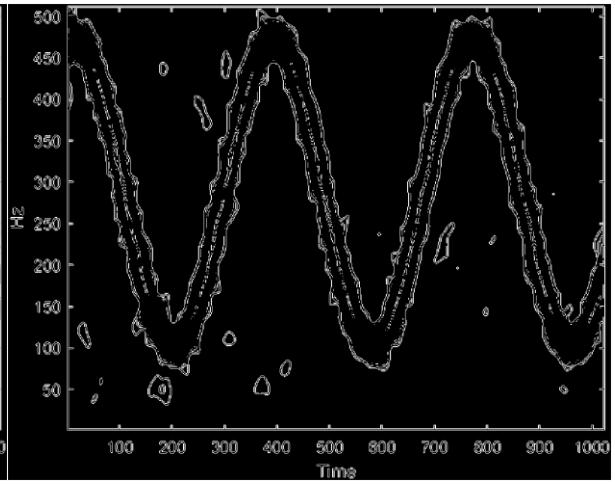
2.3.1 Primjena Sobel algoritma na signal sinusoidnom modulacijom

Kod za primjenu algoritma sobel:

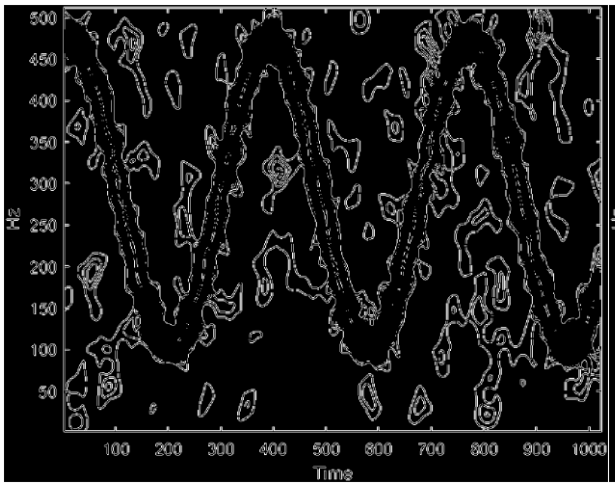
```
ssig2Sobel = edge(slika, 'sobel','both');
imshow(ssig2Sobel);
```



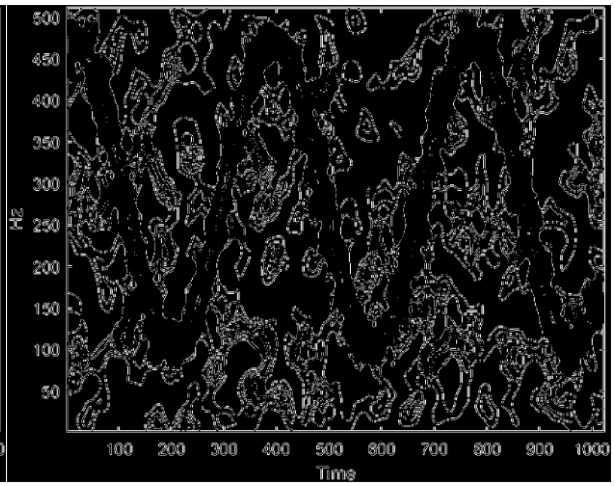
Slika 9. *Primjena sobel algoritma na sig2 bez šuma*



Slika 10. *Primjena sobel algoritma na sig2 sa SNR=10*



Slika 11. *Primjena sobel algoritma na sig2 sa SNR=5*



Slika 12. *Primjena sobel algoritma na sig2 sa SNR=1*

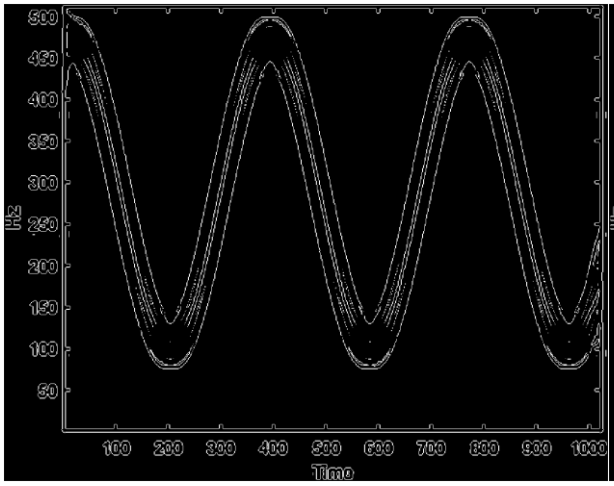
Poseban parametar za ovaj algoritam je $'both'$, to znači da uzima u obzir smjer rubova, vertikalne i horizontalne. Sa samo horizontalnim ili samo vertikalnim usmjerenjem nestaju neki rubovi, zbog toga je odabrani parametar $'both'$. Slika devet je idealni primjer jer nema šum. Usporedit ćemo ga sa različitim jačinama šuma, Slika 10 ima SNR od deset, Slika 11 od pet a Slika 12 od jedan. $Sobel$ algoritam dobro prikazuje rub vanjske komponente na

Slici 9, ali to ne uspijeva sa unutarnjom komponentom. Možemo primijetiti da povećavanjem SNR-a izobličavaju se i sve manje vide rubovi unutarnje komponente(Slika 12, Slika 11, Slika10). Algoritam je vrlo osjetljiv na šum pa zbog toga sinusoida na nekim mjestima postaje prekrivena šumom. Sljedeći algoritam je log.

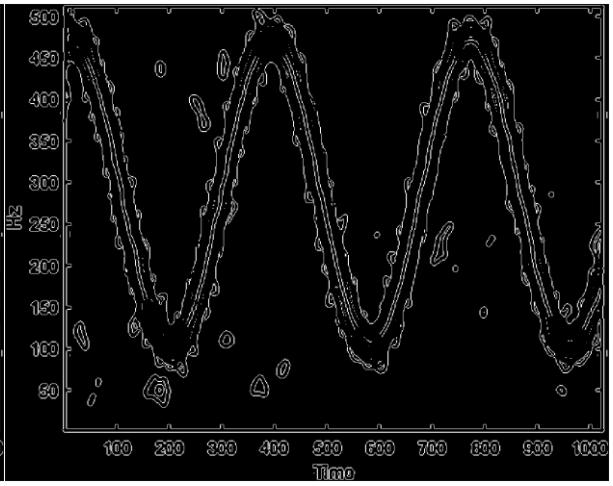
2.3.2 Primjena Log algoritma na signal sinusoidnom modulacijom

Kod za primjenu log algoritma:

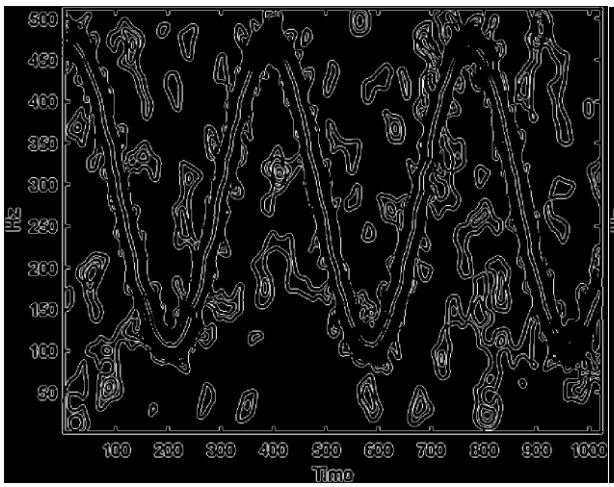
```
ssig2Log = edge(slika, 'log');
imshow(ssig2Log);
```



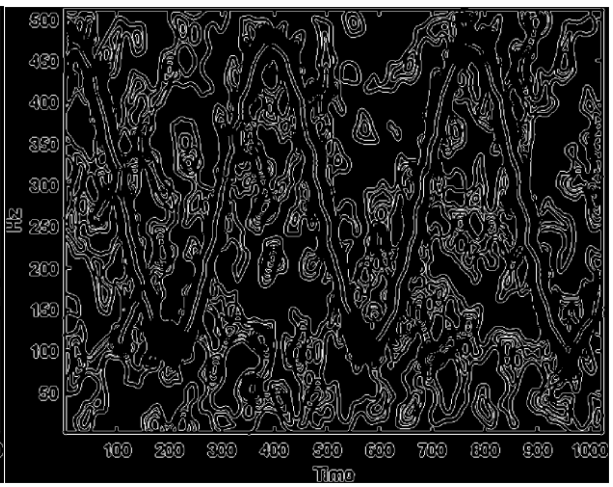
Slika 13. *Primjena log algoritma na sig2 bez šuma*



Slika 14. *Primjena sobel algoritma na sig2 sa SNR=10*



Slika 15. *Primjena log algoritma na sig2 sa SNR=5*



Slika 16. *Primjena log algoritma na sig2 sa SNR=1*

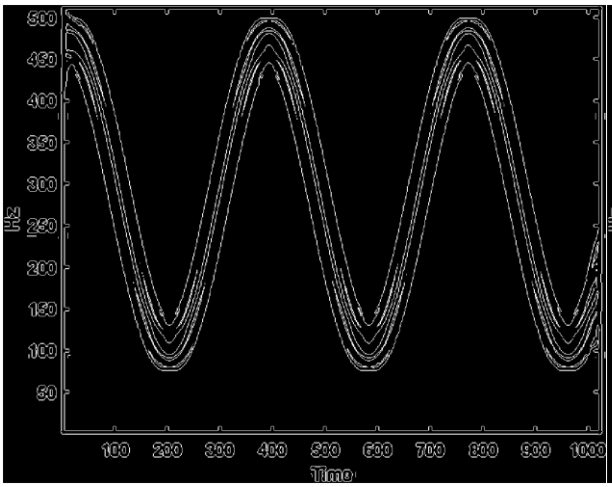
Za log algoritam nije primijenjeni posebni parametar. Već iz usporedbe sa $Sobel$ algoritmom da je $sobel$ precizniji u rubovima jer log u Slici 13 pokazuje točkaste,tj. nedovršene rubove. Log pokušava prikazati više rubova drugih komponenata ali ne uspijeva. Log algoritam izgleda više osjetljivi na šum nego $Sobel$ jer je primio više rubova šuma i postao više iskrivljeni(slika 14, slika 15, slika16). Sa jačim šumom sve više gubi na rubovima

komponenata bliže šumu, ali unutarnja komponenta postaje čak bolje vidljiva. Na Slici 16 vidimo da je vanjski rub komponenti gore izobličen i spojen sa šumom nego u primjerima *sobel* algoritma. Sljedeći algoritam je *Zerocross*.

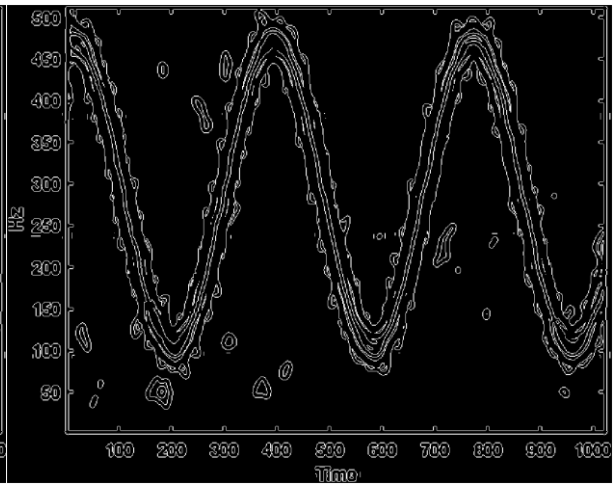
2.3.3 Primjena Zerocross algoritma na signal sinusoidnom modulacijom

Kod za primjenu zerocross algoritma:

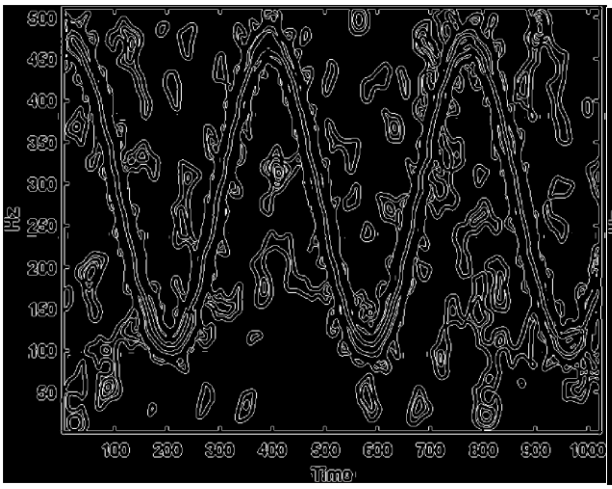
```
ssig2Zerocross = edge(slika, 'zerocross',0.0005);
imshow(ssig2Zerocross);
```



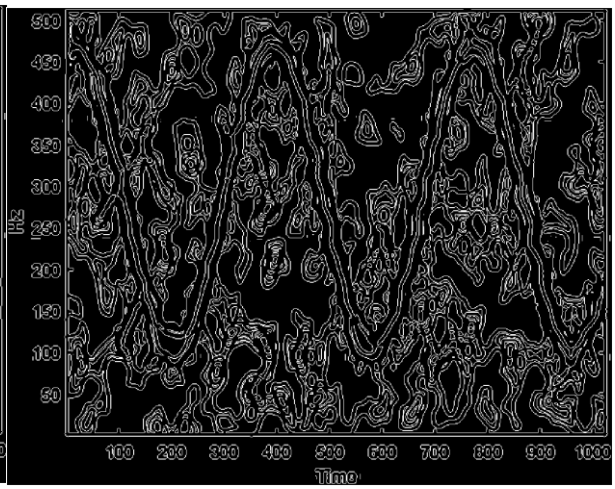
Slika 17. Primjena zerocross algoritma na sig2 bez šuma



Slika 18. Primjena zerocross algoritma na sig2 sa SNR=10



Slika 19. Primjena zerocross algoritma na sig2 sa SNR=5



Slika 20. Primjena zerocross algoritma na sig2 sa SNR=1

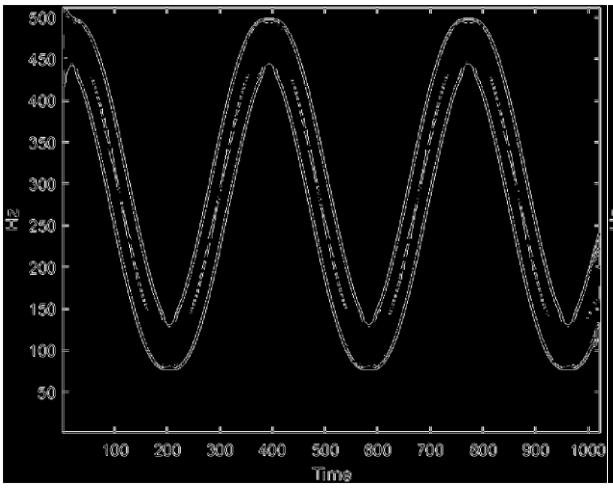
Zerocross ima poseban filter '*h*'. Broj filtera 0.0005 je dobiven nakon testiranja manjim i većim brojevima do željenog rezultata, da se vide što više rubova što više komponenata. Izgleda da je signal otprilike isto osjetljiv kao *log* algoritam samo što prikazuje malo više

rubova(Slika 17). $Zerocross$ pokazuje više detalja, više rubova nego log i $sobel$. Jačim šumom signal postaje sve manje prepoznatljivi, ali istovremeno održava više rubova komponenata nego $sobel$ i log (Slika 18, Slika 19,Slika 20). Sa SNR od jedan vanjska komponenta je gore izobličena nego u log i $sobel$ primjerima, ali je zato unutarnja komponenta bolje vidljiva (slika dvadeset). Sinusoida je još uvijek prepoznatljiva. Sljedeći algoritam je $prewitt$.

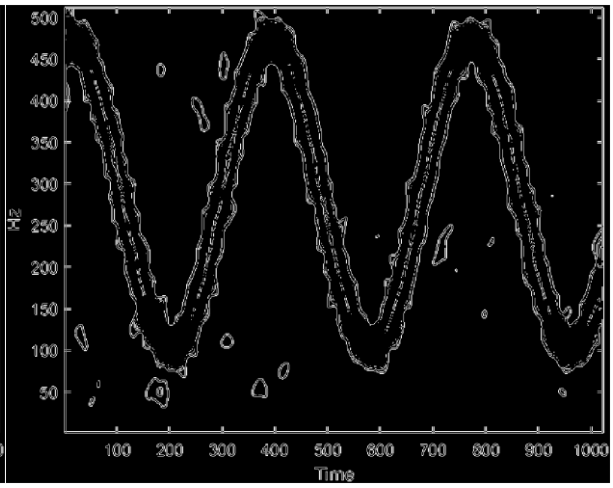
2.3.4 Primjena Prewitt algoritma na signal sinusoidnom modulacijom

Kod za primjenu $zerocross$ algoritma:

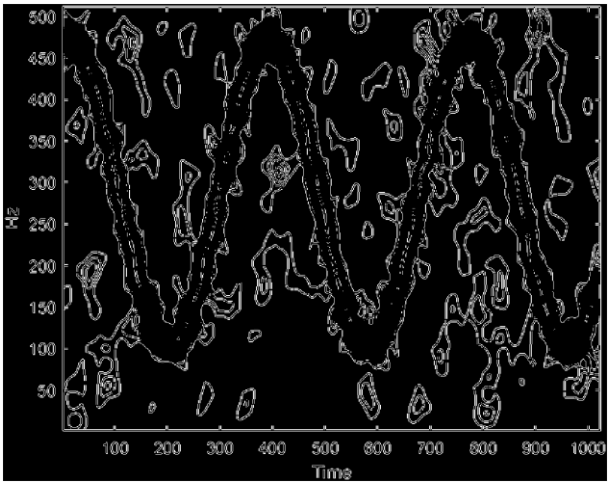
```
sgnsig2Prewitt = edge(slika, 'prewitt','both');
imshow(sgnsig2Prewitt);
```



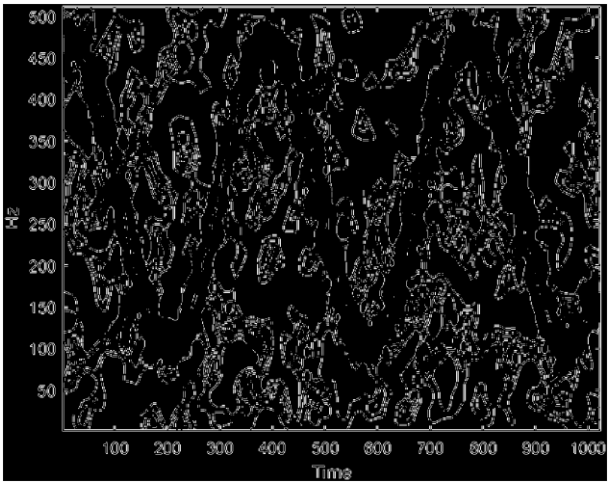
Slika 21. Primjena prewitt algoritma na sig2 bez šuma



Slika 22. Primjena prewitt algoritma na sig2 sa SNR=10



Slika 23. Primjena prewitt algoritma na sig2 sa SNR=10



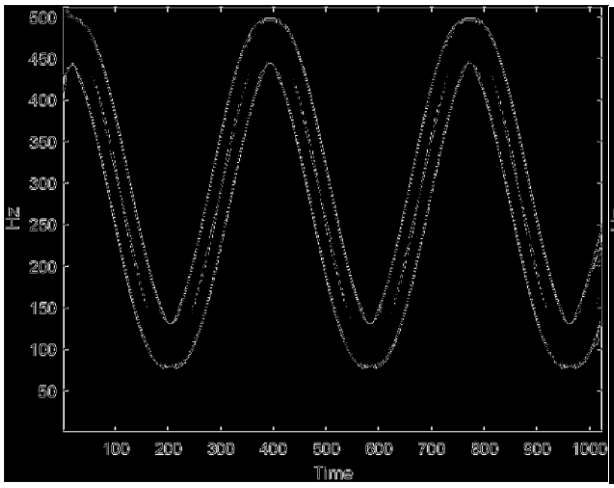
Slika 24. Primjena prewitt algoritma na sig2 sa SNR=

Prewitt ima posebni parametar za smjer rubova kao i *sobel*. Algoritam dobro prokazuje vanjski rub ali ne unutarnji(Slika 20, Slika 21). Primjeri *prewitt* algoritma izgledaju kao kopija primjera *sobel* algoritma. Na primjeru sinusoide *prewitt* nam daje rezultate kao i *sobel*(Slika 22, Slika 23, Slika 24). Sljedeći algoritam je *Roberts*.

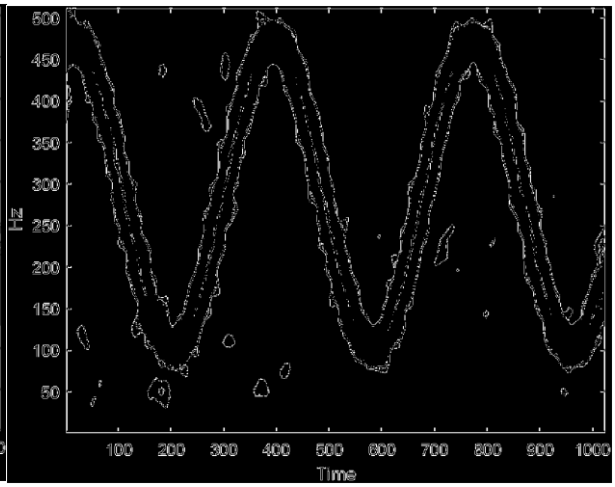
2.3.5 Primjena Roberts algoritma na signal sinusoidnom modulacijom

Kod za primjenu *roberts* algoritma:

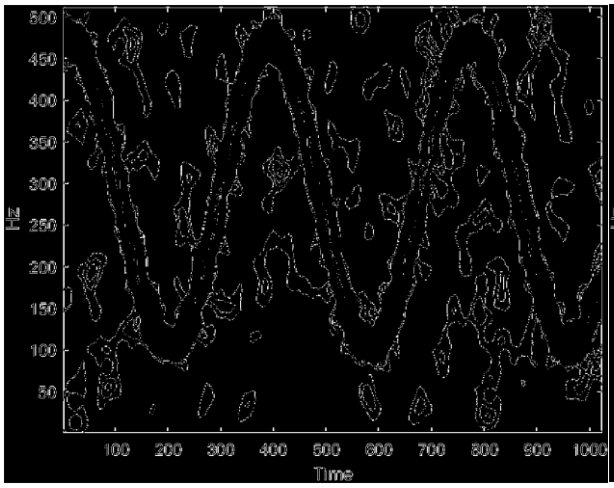
```
sgnsig2Roberts = edge(slika, 'roberts');
imshow(sgnsig2Roberts);
```



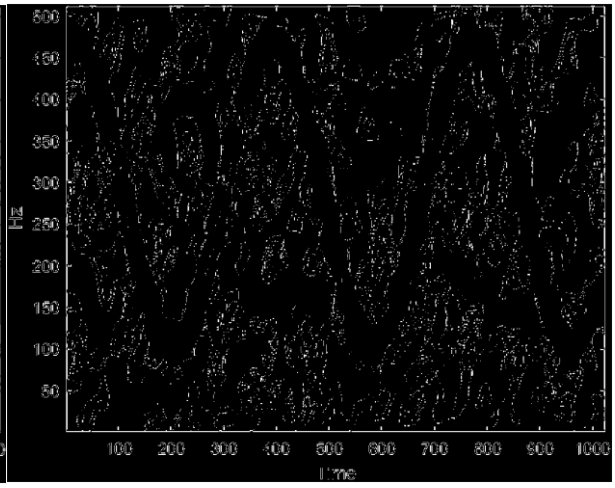
Slika 25. Primjena roberts algoritma na sig2 bez šuma



Slika 26. Primjena roberts algoritma na sig2 sa SNR=10



Slika 27. Primjena roberts algoritma na sig2 sa SNR=5



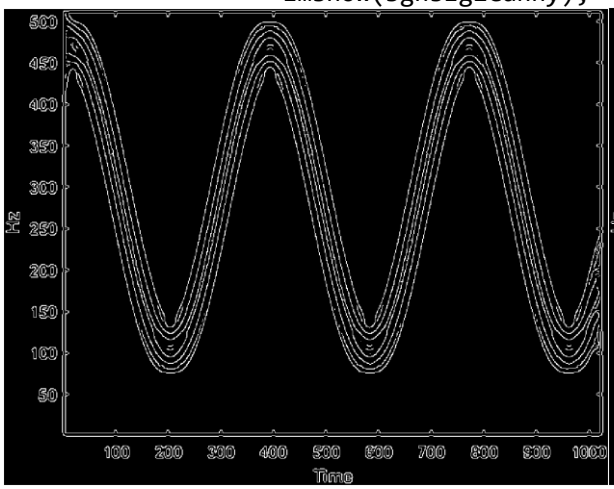
Slika 28. Primjena roberts algoritma na sig2 sa SNR=1

Rezultati *roberts* algoritma izgledaju jako slično rezultatima *sobel* i *prewitt* algoritama, ali imaju manje jasne rubove. Više su točkasti u usporedbi sa ostalim algoritmima(Slika 25, Slika 26, Slika 27, Slika 28). Ima prednost sa manjim šumom ali ne prikazuje sve komponente. Sa više šuma sve komponente postaju manje vidljive. Zbog male osjetljivosti na šum algoritam je također slabo osjetljiv na rubove komponenata. Sljedeći algoritam je *Canny*.

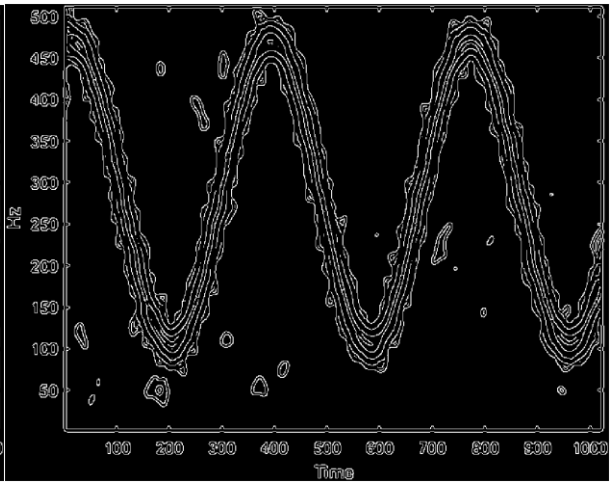
2.3.6 Primjena Canny algoritma na signal sinusoidnom modulacijom

Kod za primjenu *canny* algoritma:

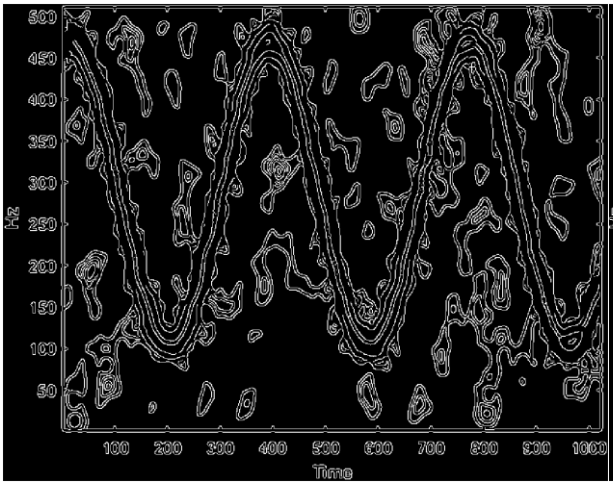
```
sgnsig2Canny = edge(slika, 'canny');
imshow(sgnsig2Canny);
```



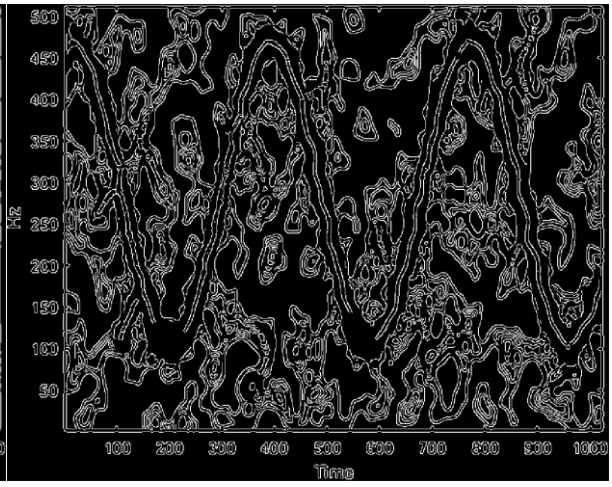
Slika 29. Primjena canny algoritma na sig2 bez šuma



Slika 30. Primjena canny algoritma na sig2 sa SNR=10



Slika 31. Primjena canny algoritma na sig2 sa SNR=5

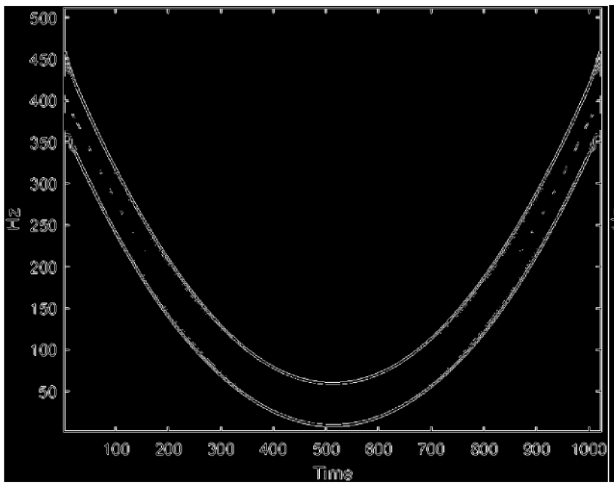


Slika 32. Primjena canny algoritma na sig2 sa SNR=1

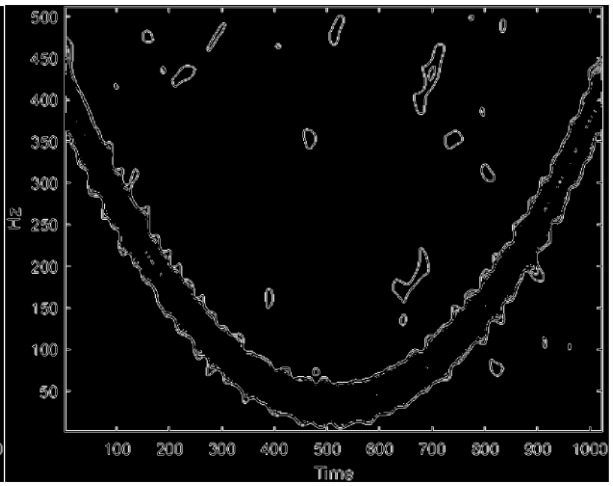
Canny algoritam najbolje prikazuje komponente spektrograma kad nema šuma. Pojačavanjem šuma vidimo da je algoritam jako osjetljiv na šum(Slika 30, Slika 31, Slika 32). Također gubi rubove na komponentama. *Zerocross* bolje prikazuje komponente signala

na jačim šumovima. Možemo zaključiti da je *canny* najviše osjetljiv algoritam na rubove komponenata i na šum. Istih 6 algoritama sada primjenjujemo na sljedeći signal.

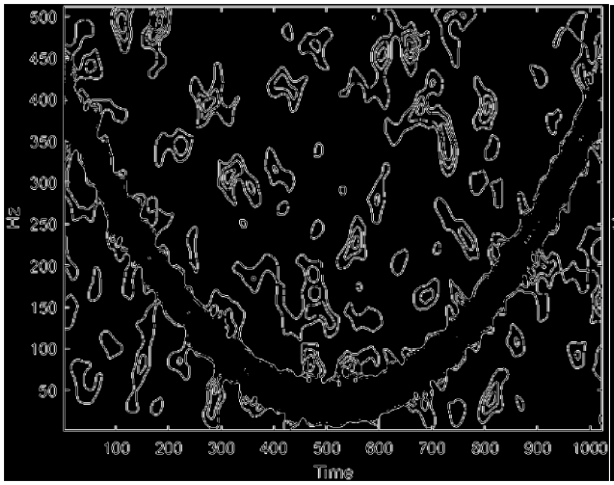
2.3.7 Primjena sobel algoritma na signal paraboličnom modulacijom



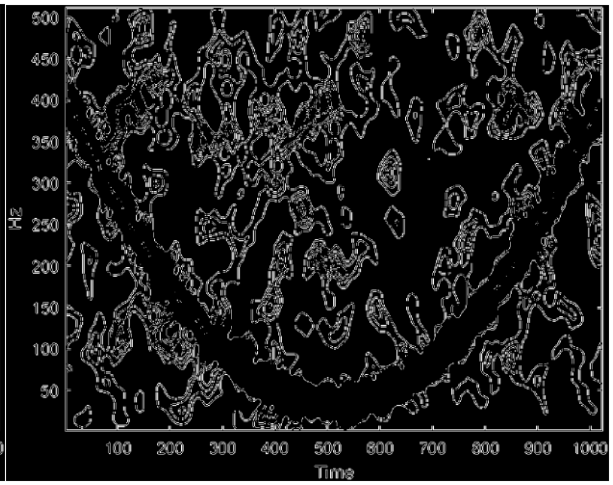
Slika 33. Primjena sobel algoritma na sig3 bez šuma



Slika 34. Primjena sobel algoritma na sig3 sa SNR=10



Slika 35. Primjena sobel algoritma na sig3 sa SNR=5

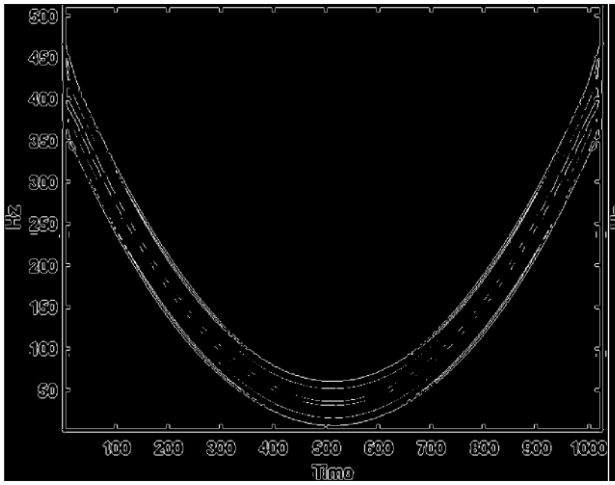


Slika 36. Primjena sobel algoritma na sig3 sa SNR=1

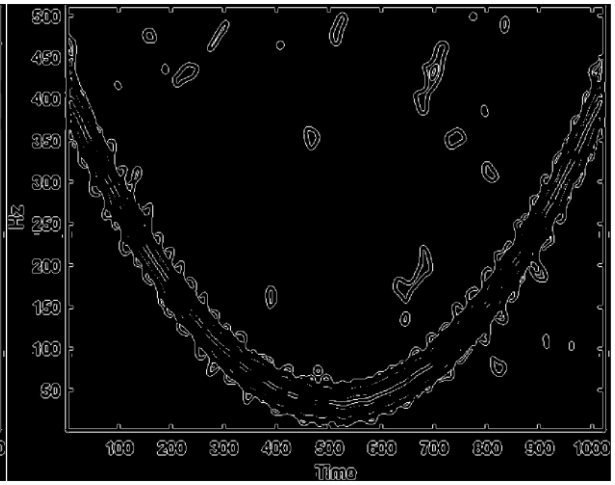
Sobel algoritam je na prošlom signalu bolje pokazivao rubove komponenata(Slika 9, Slika 10, Slika 11, Slika 12). Na ovom signalu unutarnja komponenta nema rubove nego ima točke(Slika33). Povećavanjem SNR-a vidimo neke nove točke koje bi trebale biti dio unutarnje komponente ali neke točke iz primjera sa slabijim SNR-om nestaju(Slika 34, Slika 35, Slika 36). Rub komponente se sve više iskrivljava sa povećavanjem SNR-a(Slika 34, Slika 35, Slika 36).

2.3.8 Primjena Log algoritma na signal paraboličnom modulacijom

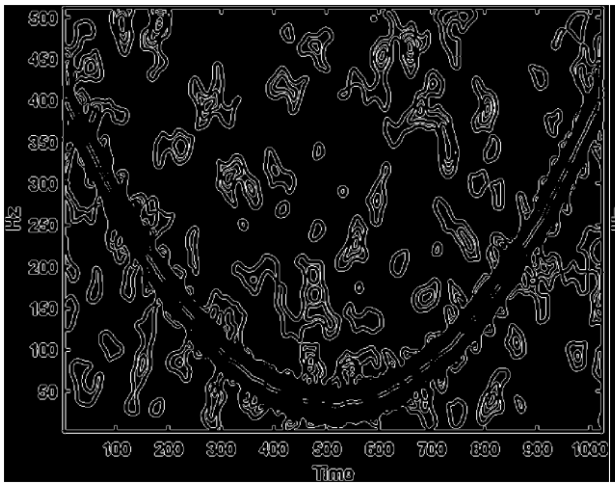
Kod za primjenu algoritama na signalu paraboličnom modulacijom je isti kao i u prošlim primjerima.



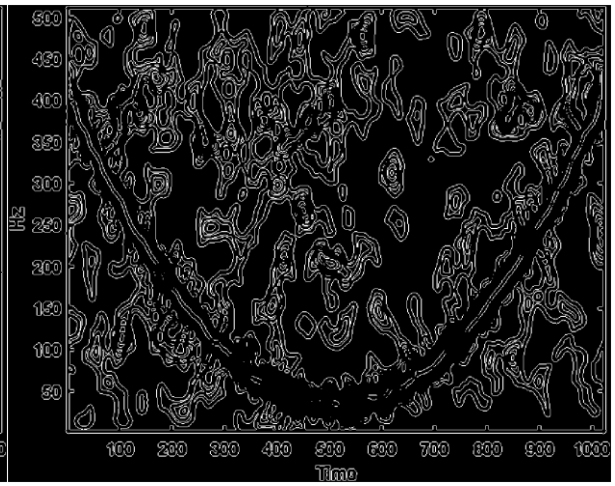
Slika 37. Primjena log algoritma na signalu3 bez šuma



Slika 38. Primjena log algoritma na signalu3 sa SNR=10



Slika 39. Primjena log algoritma na signalu3 sa SNR = 5



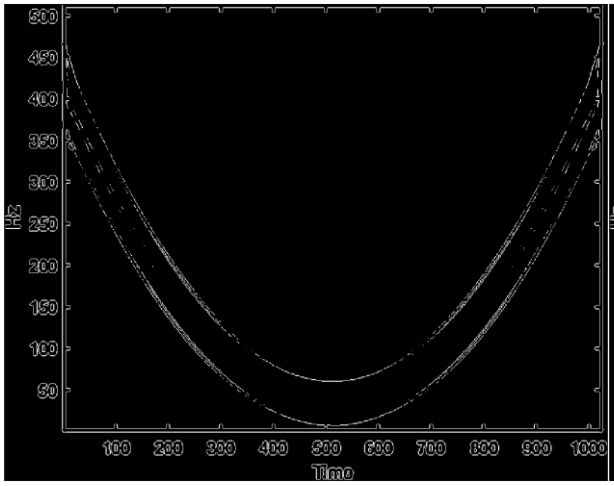
Slika 40. Primjena log algoritma na signalu3 sa SNR=1

Unutarnje komponente su više vidljive nego na primjeru $sobel$ algoritma(Slika 9). Rub unutarnje komponente je bolje vidljivi sa jačim šumom ali je isprekidani(Slika 39, Slika 39, Slika 40). $Sobel$ i Log algoritmi lošije prikazuju rubove komponenata na paraboličnom signalu nego na sinusoidi. Sljedeći primjeri su rezultati $zerocross$ algoritma.

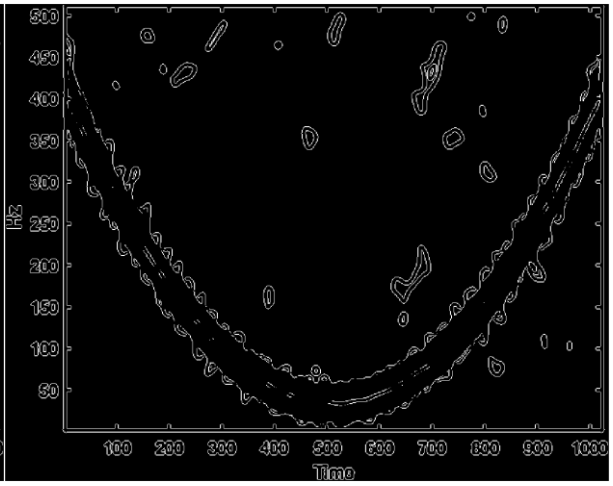
2.3.9 Primjena Zerocross algoritma na signal paraboličnom modulacijom

Kod za primjenu $zerocross$ algoritma:

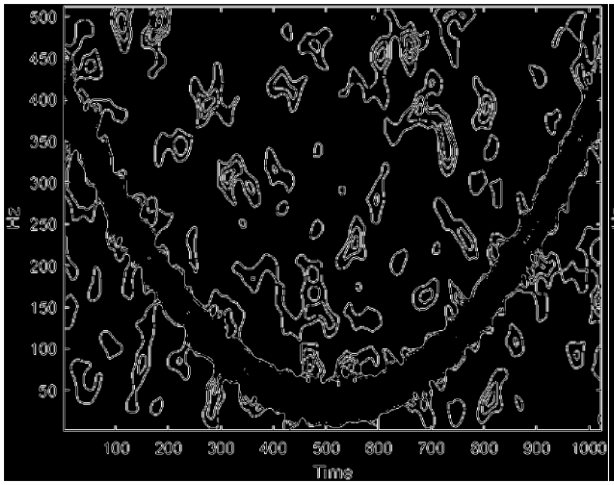
```
ssig3Zerocross = edge(J3, 'zerocross',0.0059);
imshow(ssig3Zerocross);
```



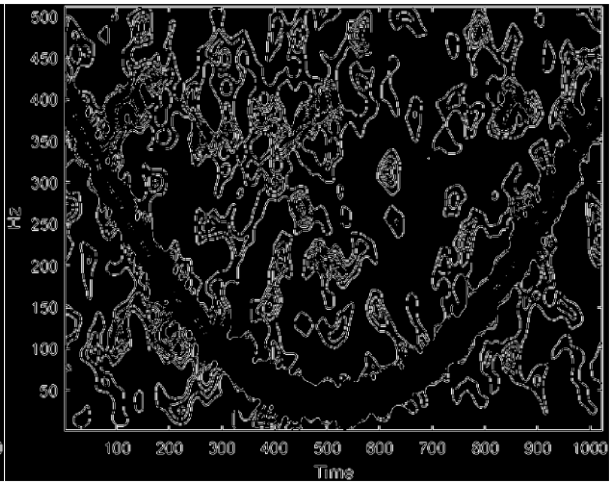
Slika 41. Primjena zerocross algoritma na signalu3 sa bez šuma



Slika 42. Primjena zerocross algoritma na signalu3 sa SNR=10



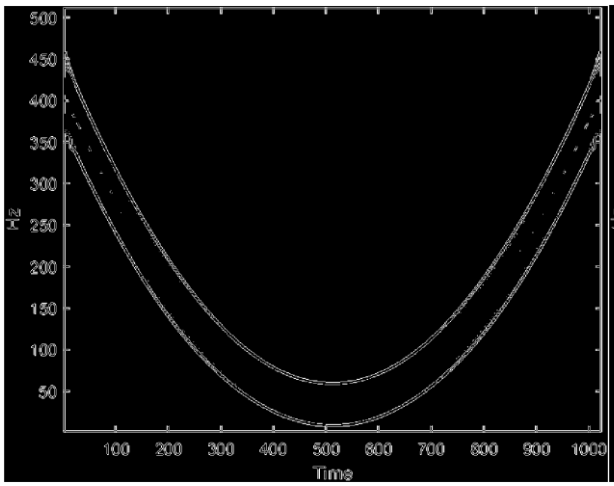
Slika 43. Primjena zerocross algoritma na signalu3 sa SNR = 5



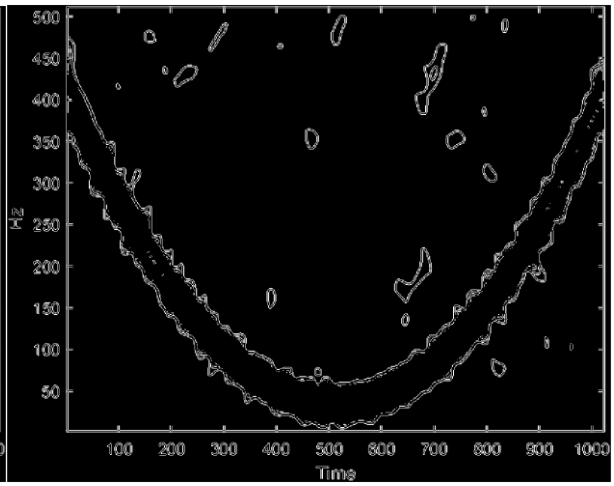
Slika 44. Primjena zerocross algoritma na signalu3 sa SNR=1

Na prošlom signalu je $zerocross$ algoritam dobio bolje rezultate nego log ali na ovom signalu je suprotno, vidi se manje komponenta i više su točkasti. Osjetljivost na šum izgleda ista kao i na prošlim primjenama algoritma samo što gore vidimo komponente(Slika 17, Slika 18, Slika 19, Slika 20). Sljedeći signal je $prewitt$.

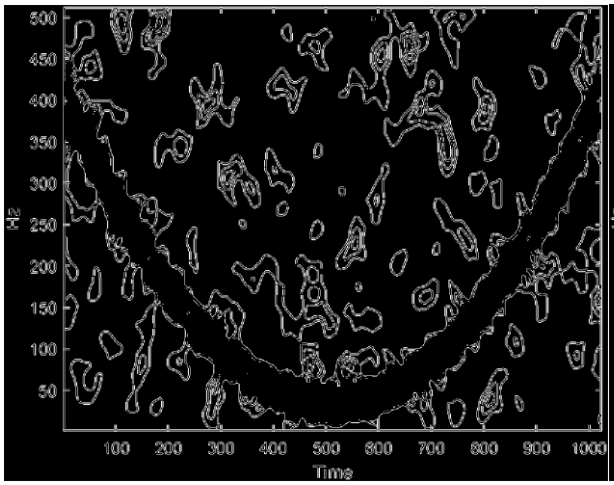
2.3.10 Primjena Prewitt algoritma na signal paraboličnom modulacijom



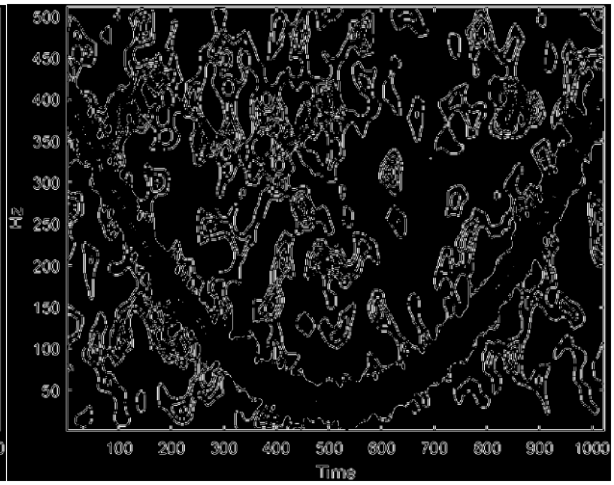
Slika 45. Primjena prewitt algoritma na signalu3 bez šuma



Slika 46. Primjena prewitt algoritma na signalu3 sa SNR=10



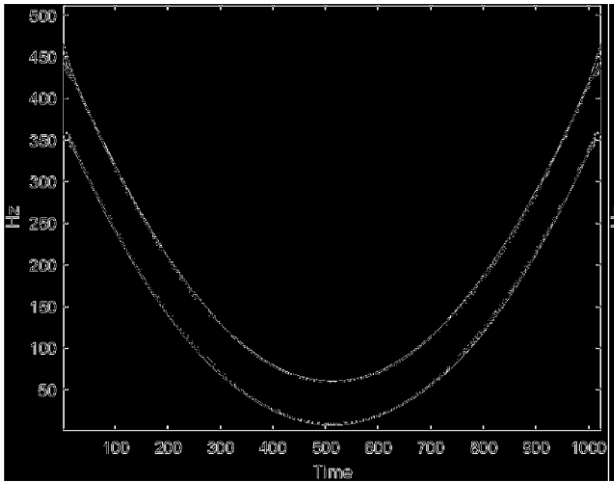
Slika 47. Primjena prewitt algoritma na signalu3 sa SNR = 5



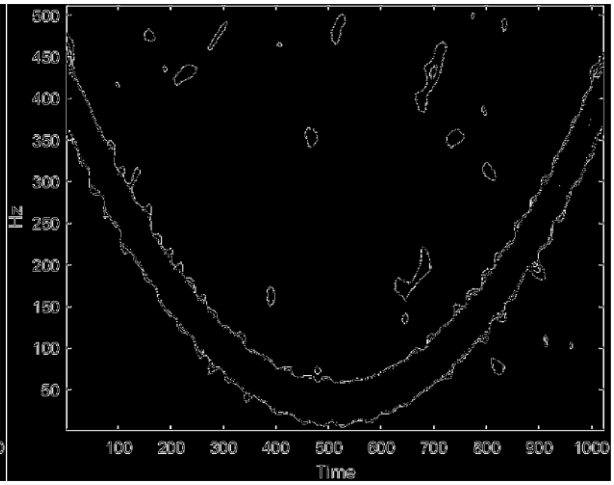
Slika 48. Primjena prewitt algoritma na signalu3 sa SNR=1

Prewitt algoritam je opet isti kao i *Sobel*(Slika 33, Slika 34, Slika 35, Slika36). Također ima dodatni parametar ta smjer rubova, '*both*'. Nema razlike korištenja *sobel* i *prewitt* algoritma u ovom kontekstu.

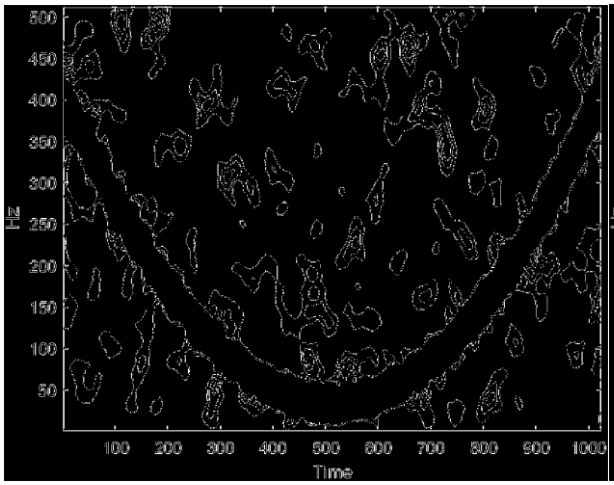
2.3.11 Primjena Roberts algoritma na signal paraboličnom modulacijom



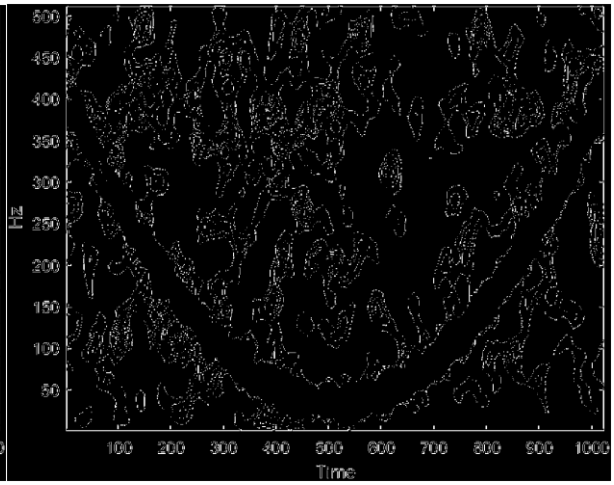
Slika 49. Primjena roberts algoritma na signalu3 bez šuma



Slika 50. Primjena roberts algoritma na signalu3 sa SNR=10



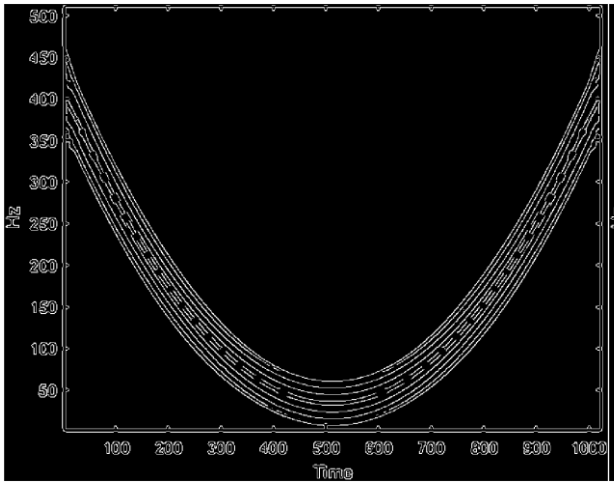
Slika 51. Primjena roberts algoritma na signalu3 sa SNR = 5



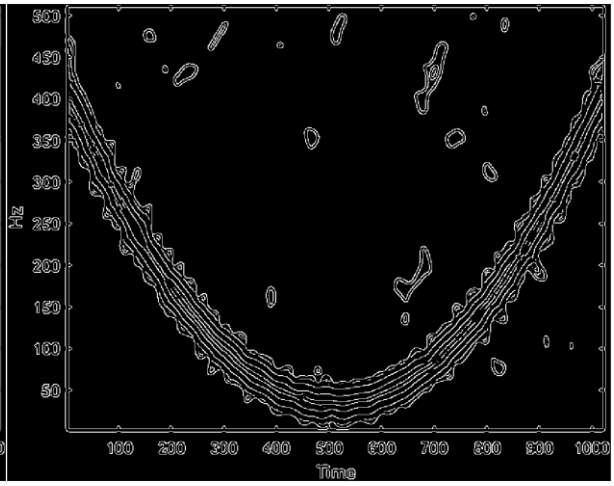
Slika 52. Primjena roberts algoritma na signalu3 sa SNR=1

Roberts algoritam ne djeluje drugačije na paraboličnom signalu. Također je slabo osijetljivi na šum ali zbog toga gubi i na točnosti prikazivanja rubova komponenta. Uopće se ne vidi unutarnja komponenta(slika 49, slika 50, slika 51, slika 52). Još nam preostaje *canny* algoritam za ovaj signal.

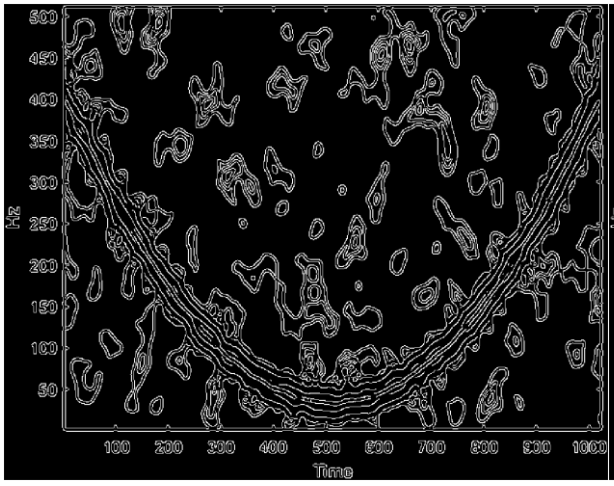
2.3.12 Primjena Canny algoritma na signal paraboličnom modulacijom



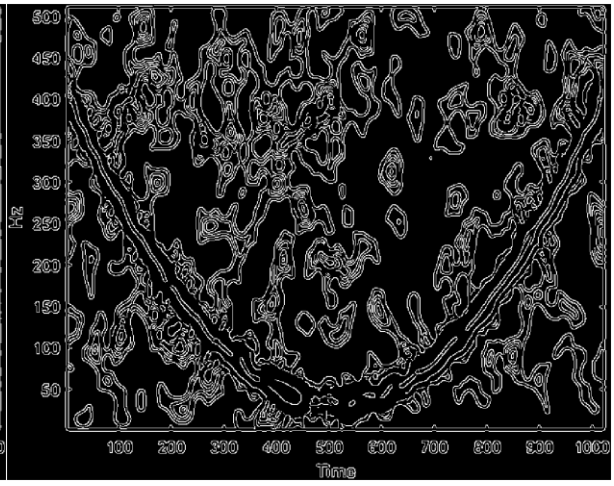
Slika 53. Primjena canny algoritma na signalu3 bez šuma



Slika 54. Primjena canny algoritma na signalu3 sa SNR=10



Slika 55. Primjena canny algoritma na signalu3 sa SNR = 5



Slika 56. Primjena canny algoritma na signalu3 sa SNR=1

Možemo zaključiti isto kao i na prošlom signalu(Slika 29, Slika 30, Slika 31, Slika 32), $Canny$ opet najbolje prikazuje komponente ali je i zbog toga jako osjetljiv na šum. Sa više šuma vanjski rub nestaje(Slika 56), tj. stapa se sa šumom ali svejedno imamo vidljivu unutarnju komponentu.

3. Završni dio

Generirano je četiri signala različitim modulacijama u Matlabu. Prvi signal je linearnom modulacijom, drugi sinusoidnom, treći paraboličnom a četvrti konstantnom. Linearni i konstantni signali su bili korišteni za demonstraciju generiranja i procesiranja signala. Na sinusoidnom i paraboličnom signalu smo primijenili algoritme za detekciju rubova. Primijenjeno je šest algoritama. Za svaki signal je generirano dvadeset i četiri matrica, četiri matrica po algoritmu. Rezultati su uspoređeni i opisani.

$Sobel$ i $prewitt$ u kontekstu spektrograma nemaju drugačije rezultate, isti su. Vanjski rub komponente dobro pokazuju ali unutarnji rubovi su jedva vidljivi te s većim razinama šuma rubovi se gube, iako ostaje dobro vidljiva silueta.

Log i $zerocross$, algoritmi daju jako slične rezultate. Log algoritam manje ističe rubove jer nije toliko osjetljiv na komponente kao i $zerocross$. Na jačim šumovima $zerocross$ mnogo bolje pokazuje rubove komponenata ali isto tako i jako prikazuje rubove šuma.

$Roberts$ je definitivno najmanje osijetljivi algoritam, ne samo na rubove komponenata nego i na šum. Što je veći šum gube se i komponente i siluete komponenta. Dobar je na manjim šumovima iako gubi unutarnje komponente spektrograma.

$Canny$ najbolje prikazuje komponente, gotovo kao da je originalna slika spektrograma u crno bijeloj boji. Upravo zbog toga na $Canny$ algoritam najgore utječe šum. Vanjska komponenta signala se najviše iskrivala na primjenama $canny$ algoritma, ali prednost u tome je da najbolje pokazivao unutarnje rubove komponenata.

4. Popis slika

Slika 1 linearni signal, generiran u MatLab-u……1

Slika 2. linearni signal sa šumom, generiran u MatLab-u gaussovim šumom……2

Slika 3. formula funkcije spektrograma korištene u radu, iz Time-Frequency Toolbox, [7] ……2

Slika 4. Spektrogram linearnog signala, generiran u MatLab -u……3

Slika 5. Spektrogram konstantnog signala, generiran u MatLab -u……3

Slika 6. GrayScale Spektrogram sa šumom linearnog signala, generiran u MatLab-u……6

Slika 7. Spektrogram bez šuma sinusoidnog signala, generiran u MatLab-u……7

Slika 8. Spektrogram bez šuma paraboličnog signala, generiran u MatLab-u……7

Slika 9. Primjena sobel algoritma na sig2 bez šuma……9

Slika 10. Primjena sobel algoritma na sig2 sa SNR=10……9

Slika 11. Primjena sobel algoritma na sig2 sa SNR=5……9

Slika 12. Primjena sobel algoritma na sig2 sa SNR=1……9

Slika 13. Primjena log algoritma na sig2 bez šuma…… 10

Slika 14. Primjena sobel algoritma na sig2 sa SNR=10……10

Slika 15. Primjena log algoritma na sig2 sa SNR=5……10

Slika 16. Primjena log algoritma na sig2 sa SNR=1……10

Slika 17. Primjena zerocross algoritma na sig2 bez šuma……11

Slika 18. Primjena zerocross algoritma na sig2 sa SNR=10……11

Slika 19. Primjena zerocross algoritma na sig2 sa SNR=5……11

Slika 20. Primjena zerocross algoritma na sig2 sa SNR=1……11

Slika 21. Primjena prewitt algoritma na sig2 bez šuma……12

Slika 22. Primjena prewitt algoritma na sig2 sa SNR=10……12

Slika 23. Primjena prewitt algoritma na sig2 sa SNR=10……12

Slika 24. Primjena prewitt algoritma na sig2 sa SNR=1……12

Slika 25. Primjena roberts algoritma na sig2 bez šuma……13

Slika 26. Primjena roberts algoritma na sig2 sa SNR=10……13
Slika 27. Primjena roberts algoritma na sig2 sa SNR=5……13
Slika 28. Primjena roberts algoritma na sig2 sa SNR=1……13
Slika 29. Primjena canny algoritma na sig2 bez šuma……14
Slika 30. Primjena canny algoritma na sig2 sa SNR=10……14
Slika 31. Primjena canny algoritma na sig2 sa SNR=5……14
Slika 32. Primjena canny algoritma na sig2 sa SNR=1……14
Slika 33. Primjena sobel algoritma na sig3 bez šuma……15
Slika 34. Primjena sobel algoritma na sig3 sa SNR=10……15
Slika 35. Primjena sobel algoritma na sig3 sa SNR=5 ……15
Slika 36. Primjena sobel algoritma na sig3 sa SNR=1……15
Slika 37. Primjena log algoritma na signalu3 bez šuma ……16
Slika 38. Primjena log algoritma na signalu3 sa SNR=10……16
Slika 39. Primjena log algoritma na signalu3 sa SNR = 5 ……16
Slika 40. Primjena log algoritma na signalu3 sa SNR=1……16
Slika 41. Primjena zerocross algoritma na signalu3 sa bez šuma……17
Slika 42. Primjena zerocross algoritma na signalu3 sa SNR=10……17
Slika 43. Primjena zerocross algoritma na signalu3 sa SNR = 5……17
Slika 44. Primjena zerocross algoritma na signalu3 sa SNR=1……17
Slika 45. Primjena prewitt algoritma na signalu3 bez šuma……18
Slika 46. Primjena prewitt algoritma na signalu3 sa SNR=10……18
Slika 47. Primjena prewitt algoritma na signalu3 sa SNR = 5 ……18
Slika 48. Primjena prewitt algoritma na signalu3 sa SNR=1……18
Slika 49. Primjena roberts algoritma na signalu3 bez šuma ……19
Slika 50. Primjena roberts algoritma na signalu3 sa SNR=10……19
Slika 51. Primjena roberts algoritma na signalu3 sa SNR = 5 ……19
Slika 52. Primjena roberts algoritma na signalu3 sa SNR=1……19
Slika 53. Primjena canny algoritma na signalu3 bez šuma……20
Slika 54. Primjena canny algoritma na signalu3 sa SNR=10……20
Slika 55. Primjena canny algoritma na signalu3 sa SNR = 5……20
Slika 56. Primjena canny algoritma na signalu3 sa SNR=1……20

5. Izvori

Internet stranice

1) TechTarget, Search networking, definicija signala. [ONLINE] Dostupno na: https://www.techtarget.com/searchnetworking/definition/signal [Pristupljeno : 15.9.2022]
2) Wikipedia, Signal. [ONLINE] Dostupno na: https://hr.wikipedia.org/wiki/Signal [Pristupljeno : 15.9.2022]
3) Izotope, LEARN MUSIC AND AUDIO PRODUCTION | IZOTOPE TIPS AND TUTORIALS, Understanding Spectrograms. [ONLINE] Dostupno na: https://www.izotope.com/en/learn/understanding-spectrograms.html [Pristupljeno: 15.9.2022]
4) PNSN, What is a Spectrogram. [ONLINE] Dostupno na: https://pnsn.org/spectrograms/what-is-a-spectrogram [Pristupljeno: 16.9.2022]
5) MathWorks, Edge Detection, Edge detection methods for finding object boundaries in images. [ONLINE] Dostupno na: https://www.mathworks.com/discovery/edge-detection.html [Pristupljeno: 16.9.2022]
6) MathWorks, Signal Processing Toolbox, Perform signal processing and analysis. [ONLINE] Dostupno na: https://www.mathworks.com/products/signal.html [Pristupljeno: 16.9.2022]
7) François Auger, Patrick Flandrin, Paulo Gonçalvès, Olivier Lemoine(1995-1996) Time-Frequency Toolbox For Use with MATLAB. [ONLINE] Dostupno na : https://tftb.nongnu.org/ [Pristupljeno : 11.4.2022]

Knjige

1) Boualem Boashash (2003), Time Frequency Signal Analysis and Processing [Pristupljeno: 19.9.2022]