TU Berlin Fakultät IV
Institut für Energie und Automatisiertungstechnik
Fachgebiet Elektronische Mess- und Diagnosetechnik
Praktikum Messdatenverarbeitung

Praktikum Messdatenverarbeitung
Termin 7

Özgü Dogan (326 048)
Timo Lausen (325 411)
Boris Henckell (325 779)

9. Juli 2012

Gruppe: G1 Fr 08-10

Betreuer: Jürgen Funk

Inhaltsverzeichnis

1 Vorbereitungsaufgaben

1.1 Vorbereitungsaufgaben zu Termin 7

1.1.1 Chirp-Signal erzeugen

Als erstes sollte anhand Matlab ein chirp-Signal erzeugt und untersucht werden. Dieser wurde mit dem Aufruf chirp() erzeugt, dem man einen Zeitvektor, die Startfrequenz und weitere Angaben über den Verlauf geben konnte. Bei unserem Signal sollte ein linearer Frequenzanstieg erfolgen.

Als Beispiel wurde folgendes Signal erstellt:
t = $0 : 0.001 : 2$
chirp(t,0,1,100)

Das Chirp-Signal und das Spektrogram dazu sehen so aus:

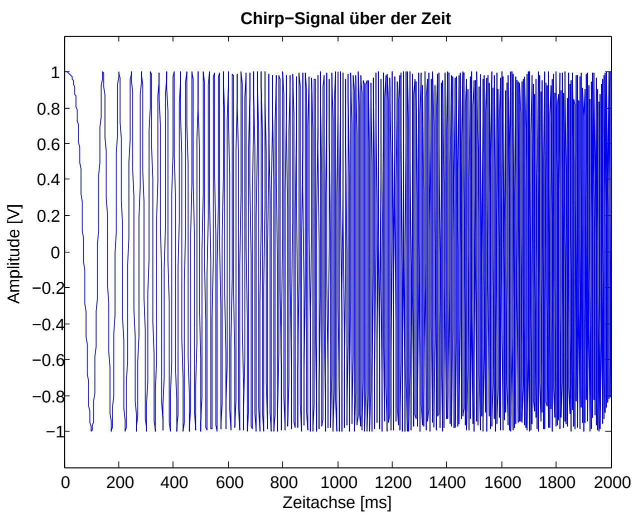

Abb. 1: erzeugtes Chirp-Signal über der Zeit

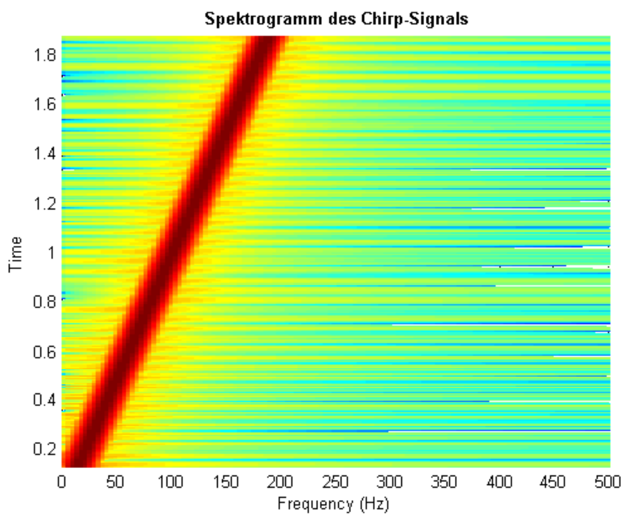

Abb. 2: Spektrogramm des erzeugten Chirp-Signals

Man sieht einen Sinusverlauf, dessen Frequenz mit der Zeit immer größer wird. Wir vermuten einen linearen Abstieg der Frequenz. Im Spektrogramm kann man deutlich sehen, dass nach einer Sekunde die Frequenz den erwünschten Wert von $100Hz$ annimmt.

Weiterhin kann man die Auswirkung der Eingabevariablen des Spektrogrammaufrufs auf das entstehende Spektrogramm untersuchen. Hier sind zwei Beispiele, in denen einmal die Überlappungsfläche zwischen zwei Segmenten verkleinert wird und einmal die verwendete Fenstergröße vergrößert wird.

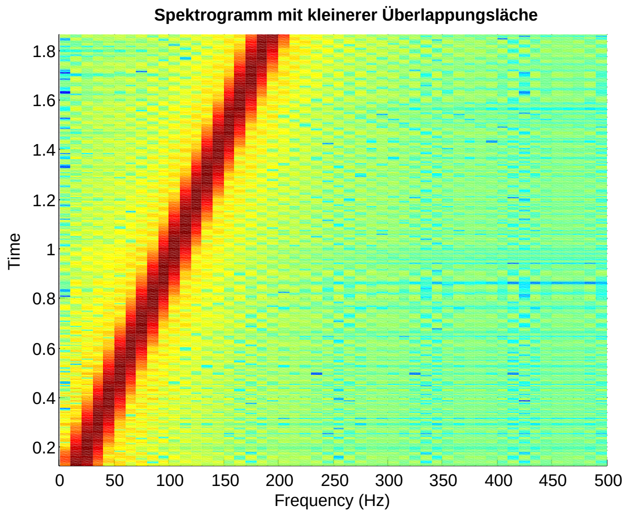

Abb. 3: Überlappung zwischen den Segmenten im Spektrogram wird kleiner gewählt

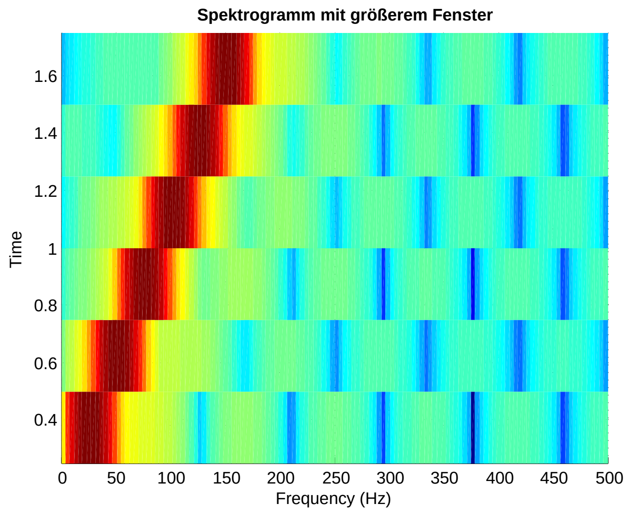

Abb. 4: verwendete Fensterfolge wird größer gewählt

Innerhalb des gewählten Fensters, kann das verwendete Signal als stationär angenommen werden. Wird das Beobachtungsfenster kürzer, nimmt auch die Frequenzauflösung ab. Wird das Fenster zu groß, kann wiederum das Signal innerhalb des Fensters nicht mehr als stationär angenommen werden. Die Verschlechterung und Ungenauigkeiten im Spektrogramm kann man in den Plots oben gut erkennen.

1.1.2 Matlab-Funktion: Frequenzverlau über der Zeit

Als nächstes sollte eine Matlab-Funktion erstellt werden, die im Zeitbereichdie momentane Frequenz ermittelt. Dieser berechnete Frequenzverlauf über der Zeit sollte geplottet und mit dem erwarteten Verlauf verglichen werden. In die erstellte Funktion kann in den Codes am Ende des Protokolls eingesehen werden. Der entstandene Plot ist hier zu sehen:

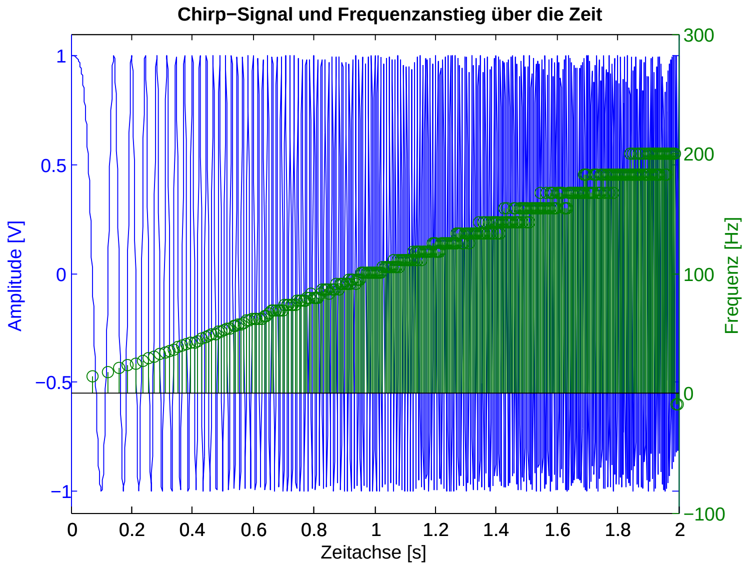

Abb. 5: Chirp-Signal und der dazugehörige Frequenzanstieg, berechnet aus dem Zeitsignal

Da beim Erstellen des Chirpsignals ein linearer Anstieg der Frequenz vorausgesetzt war, wurde auch ein linearer Frequenzverlauf erwartet. Das Ergebnis bestätigt die Erwartung.

1.1.3 Matlab-Funktion: Frequenzverlauf anhand Spektrogram

Nun sollte der gleiche Frequenzverlauf anhand des Spektrogramms ermittelt werden. Dafür implementierten wir eine weitere Matlab-Funktion, welche in den Codes zu sehen ist. Das erwartete Ergebnis war wieder ein positiver linearer Anstieg.

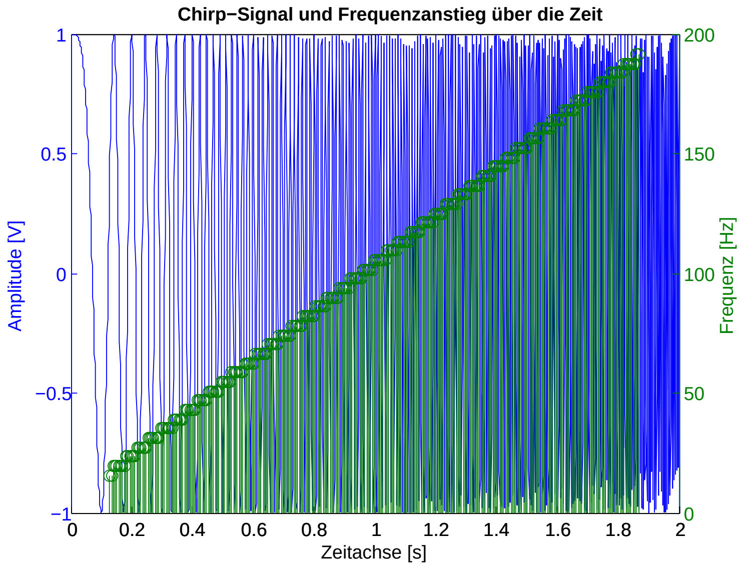

Abb. 6: Chirp-Signal und der dazugehörige Frequenzanstieg, berechnet aus dem Spektrogramm

1.1.4 Drehzahl-Berechnung anhand Amplitudenspektrum des Motorstroms

Zuletzt wurde die Drehzahl des verwendeten Motors berechnet. Dafür wurden uns drei Messreihen mit jeweils Strom- und Tachomesswerten vorgegeben. Zunächst sollten die Amplitudenspektren der Motorströme erstellt werden, welche uns durch die DFT der Stromwerte gelang. Die erhöhten Amplitudenwerte entsprachen dabei den einfachen Vielfachen der Drehfrequenz des Motors mit der jeweiligen Versorgungsspannung. Die deutlich heraussstehenden Peaks dagegen traten nur bei dem 18- oder 36-fachen Vielfachen der Drehfrequenz auf. Dieses Wissen nutzen wir, indem wir anhand eines Matlabalgorithmuses den höhsten Peak mit Index ausgaben lassen (die höchsten Peaks waren stets jene, die am nächsten zu der null standen), um daraus eine Frequenz zu bestimmen und durch 18 zu teilen. So berechneten wir die drei geforderten Drehfrequenzen. Der Algorithmus steht in den Codes, die drei Amplitudenspektren sehen folgendermaßen aus:

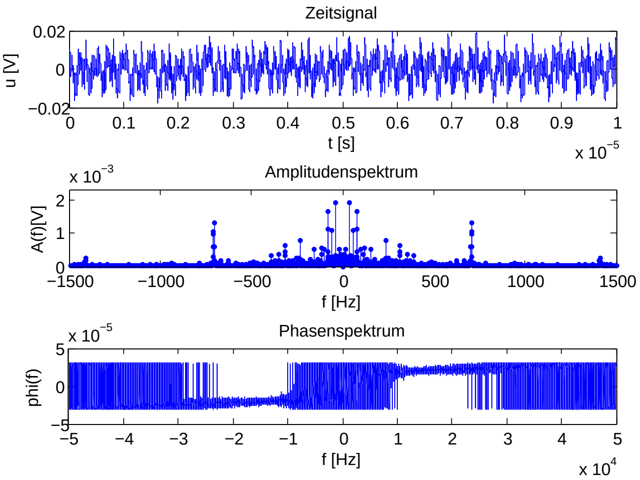

Abb. 7: Spektrum des Motorstroms bei 10V

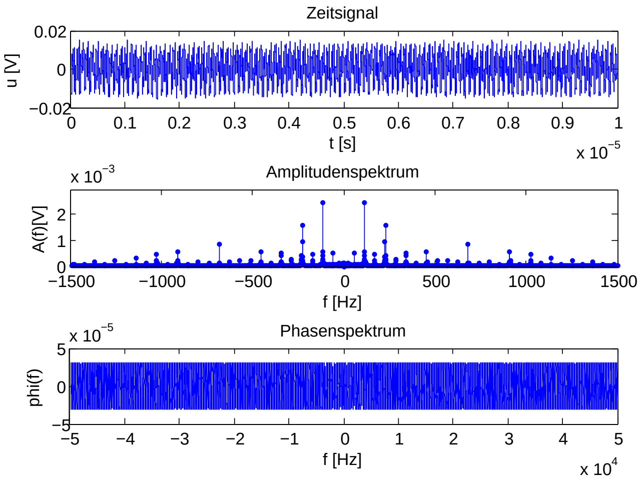

Abb. 8: Spektrum des Motorstroms bei 20V

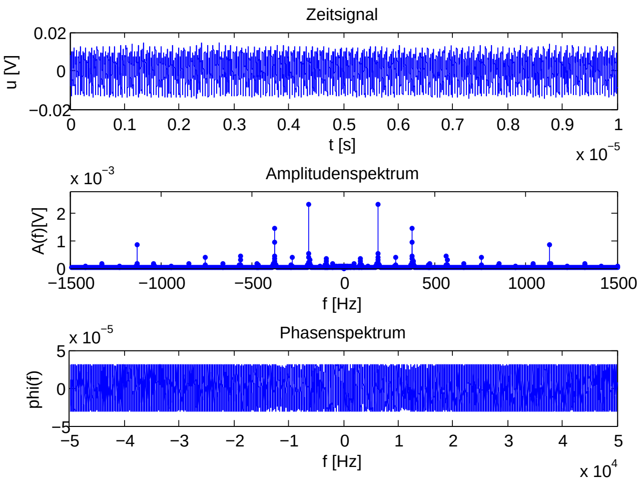

Abb. 9: Spektrum des Motorstroms bei 30V

Die berechneten Drehfrequenzen betragen für die erste Messung $(10V) = 2.1389Hz$, für die zweite Messung $(20V) = 6.3056Hz$ und für die dritte Messung $(30V) = 10.4723Hz$.

TODO:

Hier sind die Werte falsch, weil wir die falschen Paeks rausgesucht haben, Fehler muss noch korrigiert werden

Außerdem wurden die Drehfrequenzen auch aus den jeweiligen Tachosignalen bestimmt. Dafür wurde genauso vogegangen wie bei den Motorströmen. Die Spektren sind hier:

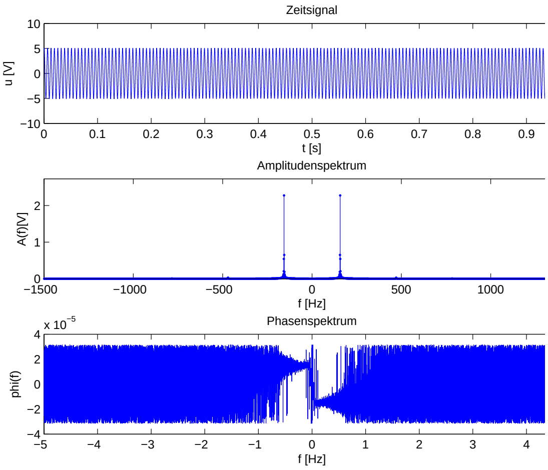

Abb. 10: Spektrum des Tachosignals bei 10V

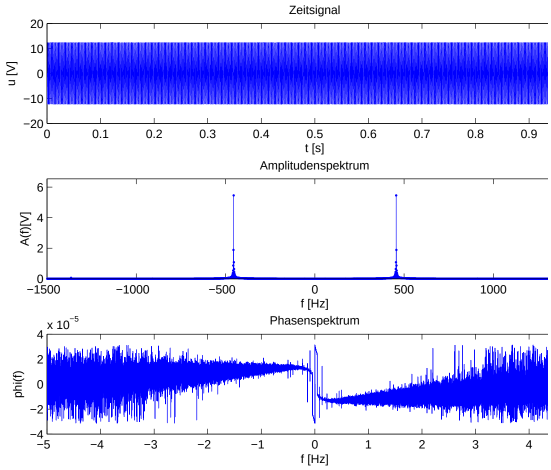

Abb. 11: Spektrum des Tachosignals bei 20V

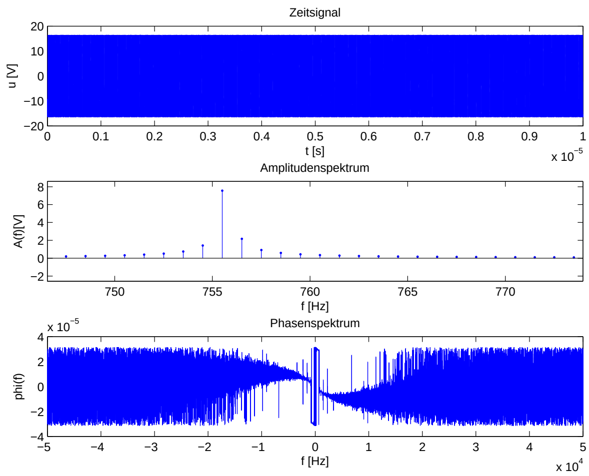

Abb. 12: Spektrum des Tachosignals bei 30V

Die anhand der Tachosignale berechneten Drehfrequenzen betragen für die erste Messung $(10V)$ = $19.5627Hz$, für die zweite Messung $(20V)$ = $56.8131Hz$ und für die dritte Messung $(30V)$ = $94.3134Hz$.
Es fällt auf, dass die Drehzahlen, welche über die Ströme berechnet werden, ca. 4mal kleiner sind als die Drehzahlen über die Tachowerte. Ein konstanter Faktor von 4 kann angenommen werden.

1.2 Vorbereitungsaufgaben zu Termin 8

1.2.1 Zerlegung des Signals mittels Haar-Tranformation

In der ersten Vorbereitungsaufgabe des 8.Praktikumstermins wird eine vorgegebene strom.m Datei verwendet, welche einen angechnittenen Sinus im Bereich $[0, 8\pi]$ darstellt. Dieses Signal wird mit der Schnellen Haar-Transformation zerlegt. Außerdem wird die Approximation und die Details dür die Skalierungen $m = 1 \ldots 5$ berechnet werden, wofür die drei Funktionen haardec.m, haardeclevel.m und getAppDet.m implementiert werden. Diese stehen unter den Codes.

1.2.2 Darstellung der Approximationen

Als nächstes werden die Approximationen und die Details des angeschnittenen Sinus-Signals dargestellt und das stationäre Spektrum sowie das Spektrogramm berechnet. Diese spektralen Darstellungen werden mit den Darstellungen mittels Wavelets verglichen. Am Ende soll ermittelt werden, welche Darstellung bestimmte Informationen über das verwendete Signal besser veranschaulicht.

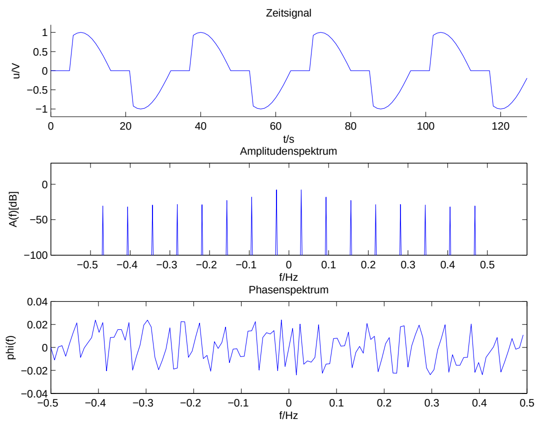

Abb. 13: Spektrum des angeschnittenen Sinus-Signal

Im stationären Spektrum des gegebenen Signals kann man die Frequenzanteile mit ihren entsprechenden Amplituden erkennen.

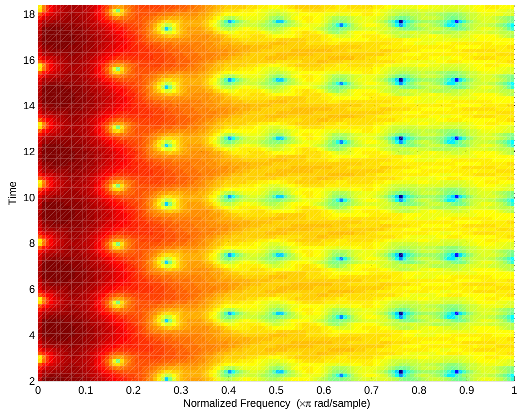

Abb. 14: Spektrogram des angeschnittenen Sinus-Signal

Außerdem kann man im Spektrogram des Signals die durch die Anschneidung entstandenen Flanken erkennen. Dies macht sich in dem zeitlich periodischen Farbverlauf des Spektrograms deutlich sichtbar.

Die Darstellungen der Approximationen und der Details für fünf Skalierungslevel sind dabei in der vierten Aufgabe der Vorbereitung dargestellt, wo sie auch zeitgleich mit den Zerlegungen des Daubechies-Wavelets (db3) verglichen werden. Da eine Zerlegung mittels Wavelets eine Tief- und Hochpassfilterung darstellt, kann man es so interpretieren, dass die gefilterten tiefen Frequenzen als Approximationen und die gefilterten hohen Frequenzen als Details abgespeichert werden. Je öfter eine Zerlegung stattfindet, desto öfter wird die Approximation hoch- und tiefpassgefiltert, wodurch am Ende in dem Approximationsvektor nur noch die aller tiefsten Frequenzen enthalten sind.

1.2.3 Daubechies-Wavelets

Nun wird das angeschnitte Sinus-Signal mit Hilfe von Daubechies-Wavelets zerlegt. Dazu verwenden wir die Matlab-Fuktionen wavedec, appcoef und detcoef und variieren die Anzahl der verschwindenen Momente der Wavelets, indem wir bei der wavedec.m Ausführung die Wavelets db1, db5, db10 und db15 wählen. Damit werden die verschwindenden Momente in jedem Wavelet 1, 5, 10 und 15 betragen. Es werden nur die Zerlegungen in Level 2 angezeigt.

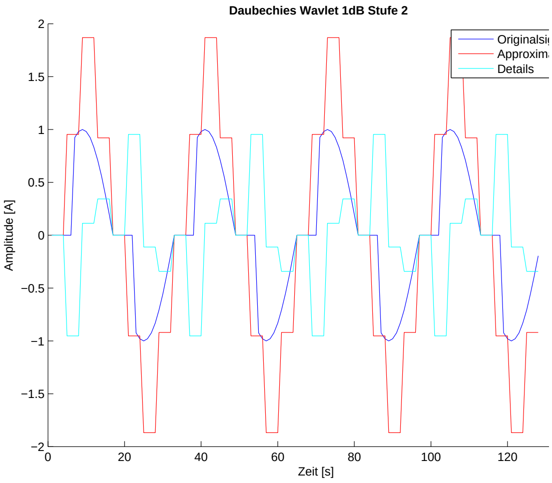

Abb. 15: Daubechies-Wavelet mit 1 vanishing moment

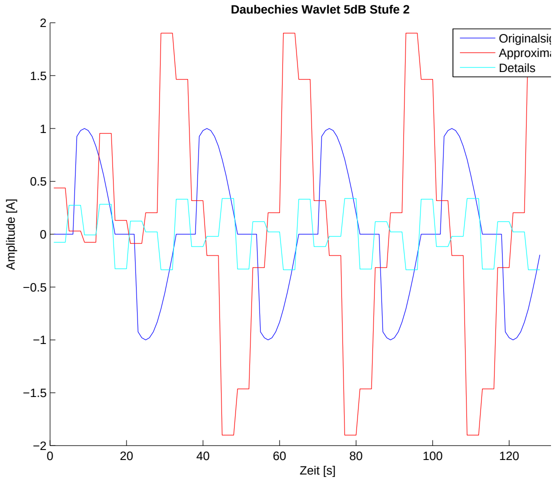

Abb. 16: Daubechies-Wavelet mit 5 vanishing moment

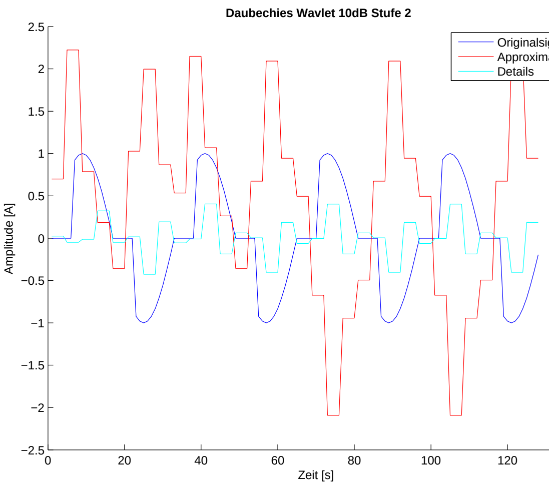

Abb. 17: Daubechies-Wavelet mit 10 vanishing moment

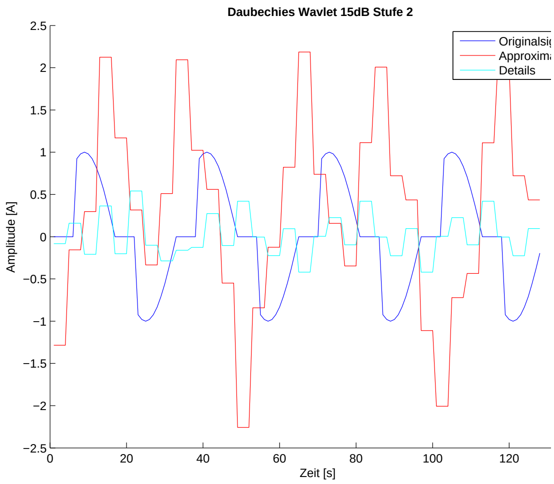

Abb. 18: Daubechies-Wavelet mit 15 vanishing moment

Man kann sehen, dass sich die Zerlegungen deutlich voneinander unterscheiden. Gehen wir zunächst auf die Approximationen ein. Es ist deutlich, dass es bei der ersten Darstellung mit nur einem verschwindenden Moment kaum einen Unterschied zu einer Haar-Wavelet Zerlegung gibt, wohingegen es bei der Zerlegung mit fünf verschwindenen Momenten der Verlauf der Approximationen sich ein wenig verändert. Wird die Anzahl der verschwindenen Momente auf zehn verdoppelt, so wird der Verlauf noch unkontinuierlicher und einzelne Amplituden fangen an zu schwanken. Bei fünfzehn bestätigt sich diese Beobachtung sogar verstärkter.
Sieht man sich die Details der Zerlegungen an, entdeckt man ebenfalls eine Veränderung. Ähnlich wie bei den Approximationen wird der kontinuierliche Verlauf mit steigender Anzahl an verschwindenen Momenten gestört und die Amplituden variieren und verkleinern sich.

1.2.4 Vergleich der Zerlegungen mittels Daubechies-Wavelets und Haar-Wavelets

Zuletzt wird in der Vorbereitung des Praktikums die Zerlegungen mittels Daubechies-Wavelets mit den Zerlegungen mittels Haar-Wavelets verglichen. Die Unterschiede sind folgende:

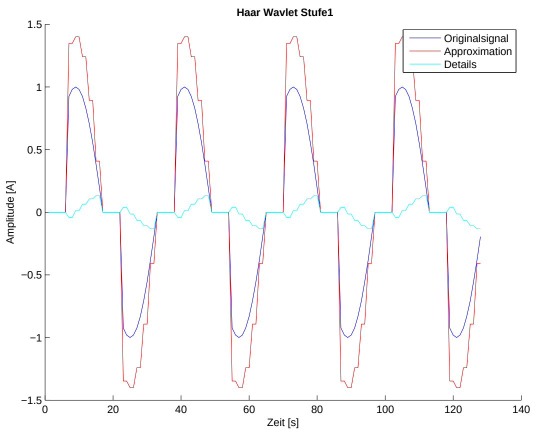

Abb. 19: Haar-Wavelet Zerlegung, Level 1

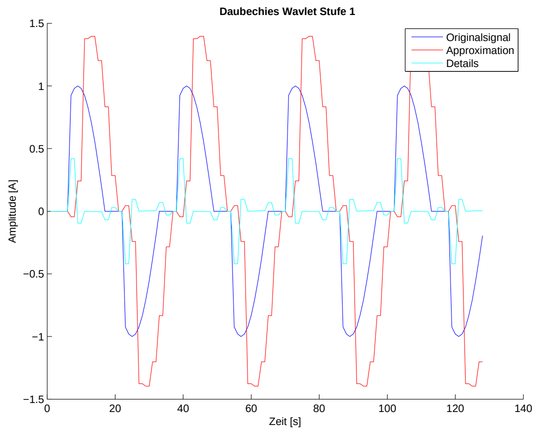

Abb. 20: Daubechies-Wavelet Zerlegung, Level 1

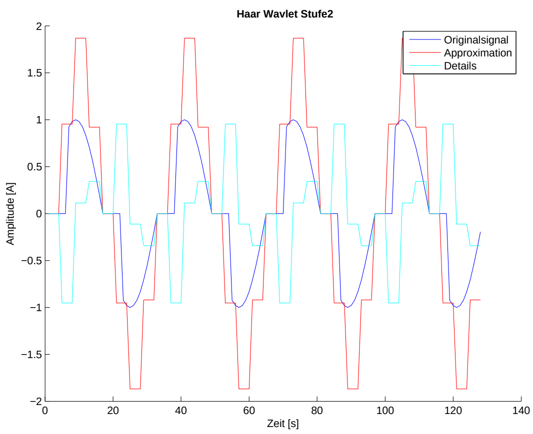

Abb. 21: Haar-Wavelet Zerlegung, Level 2

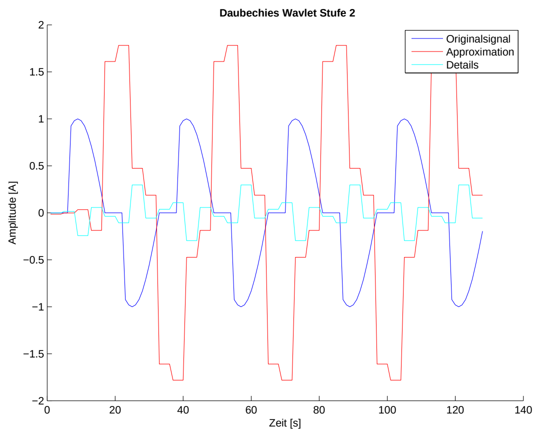

Abb. 22: Daubechies-Wavelet Zerlegung, Level 2

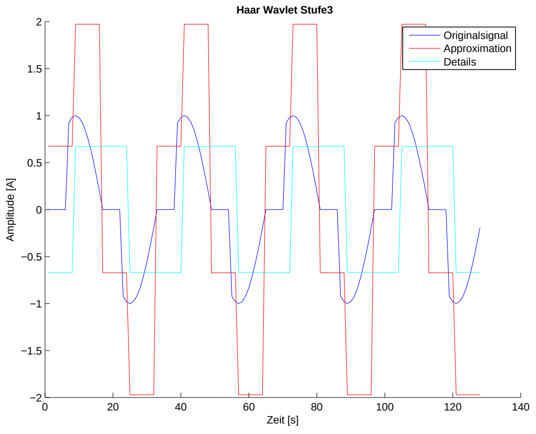

Abb. 23: Haar-Wavelet Zerlegung, Level 3

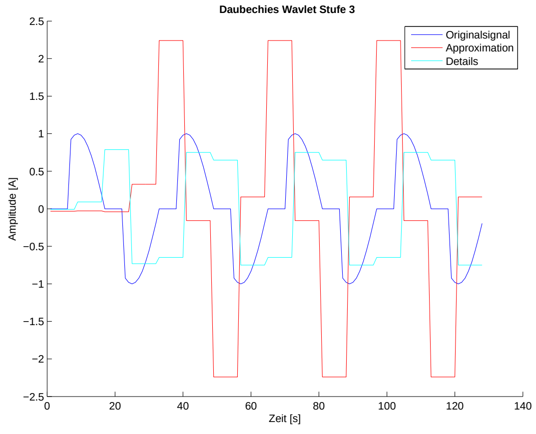

Abb. 24: Daubechies-Wavelet Zerlegung, Level 3

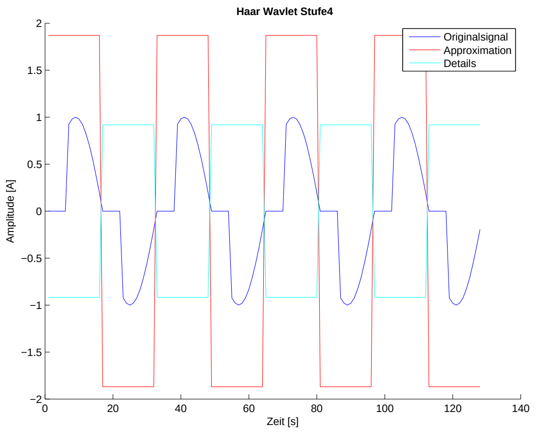

Abb. 25: Haar-Wavelet Zerlegung, Level 4

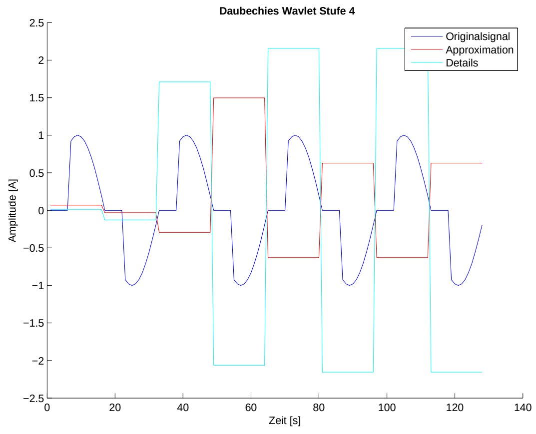

Abb. 26: Daubechies-Wavelet Zerlegung, Level 4

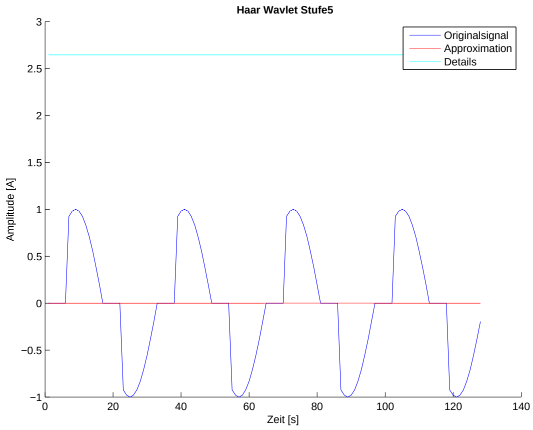

Abb. 27: Haar-Wavelet Zerlegung, Level 5

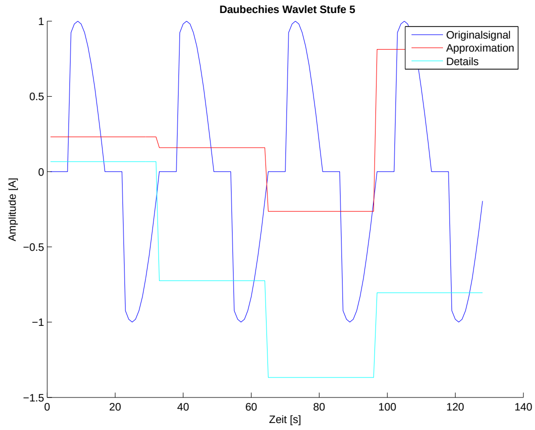

Abb. 28: Daubechies-Wavelet Zerlegung, Level 5

Zuerst fällt auf, dass die Zerlegung mit unterschiedlichen Wavelets unterschiedliche Approximation und daraus resultierend unterschiedliche Details zur Folge hat. Des weiteren fällt auf, dass bei der Haar-Wavelet Transformation in der Stufe 5 alle 4 Werte konstant 0 sind, während die Werte der Approximation bei der Daubechies-Wavelets Transformation unterschiedlich sind.

2 Durchführungen

2.1 Durchführung zu Termin 7

2.2 Durchführung zu Termin 8

Der Versuchsaufbau im 8.Termin des Praktikums ähnelte zwar der gleich Aufbau wie im 7.Termin. Anstatt aber den Motorstrom und das Tachosignal aufzunehmen, maßen wir nur den Strom (AC- und DC-Anteil) des fehlerfreien Universalmotors. Dieses Signal wurde dann mithilfe von Haar- und Daubechies-Wavelets zerlegt und untersucht. Dabei sollten geeignete Approximations- und Detailsdarstellungen gewählt werden, um den Einschaltzeitpunkt und den Drehzahlanstiegt über der Zeit darzustellen.

Außerdem wurde der Strom (nur AC-Anteil) des fehlerfreien Motors und des Motors mit Lamellenfehler im stationären Zustand aufgenommen. Diese Stromsignale der zwei verschiedenen Motoren wurden dann anhand der Wavelets auf Unterschiede untersucht. Die Ergebnisse werden in der Auswertung diskutiert.

3 Auswertung

3.1 Auswertung Termin 7

3.2 Auswertung Termin 8

4 Quellcodes

4.1 Codes aus Termin 7

4.1.1 Frequenzverlauf über der Zeit

Listing 1: frequenz imZeitbereich ausSignal

```
%Funktion zum Errechnen der Frequenz aus dem Zeitsignal

function frequenz_imZeitbereich_ausSignal (u, t)

p = length(t);
q = max(t)/p;

i = 0;
for k = 1:1:length(u)-1     %i = Anzahl der  Nulldurchge
    if (u(k) < 0 && u(k+1) >=0) || (u(k) > 0 && u(k+1) <=0)
        i = i+1;
    end
end

nulldurchgang = ones(1,i)*-9;     %Vektor der  Le   i

j = 1;
for indx = 1:1:length(u)-1
    if (u(indx) < 0 && u(indx+1) >=0) || (u(indx) > 0 && u(indx+1) <=0)
        if abs(u(indx)) > abs(u(indx+1))
            nulldurchgang(j) = indx+1;
        else
            nulldurchgang(j) = indx;        %Index, welcher am  nsten  am Nulldurchgang ist, wird
                bernommen
        end
        j = j+1;
    end
end

%Vektor fr   Frequenzeintr
v = length(nulldurchgang);
frequenzen = ones(1,v)*-9;

for n = 1:1:v-4                  %4 Nulldurchge , also 2 Perioden werden betrachtet
    diff = nulldurchgang(n+4)-nulldurchgang(n);
    frequenzen(n) = 1/(diff*0.5*q);
end

figure(1);
[AX H1 H2] = plotyy(t,u,nulldurchgang*(max(t)/p),frequenzen,'plot','stem');
% AXIS([0 2 -1.1 1.1]);
xlabel('Zeitachse [s]');
set(get(AX(1),'Ylabel'),'String','Amplitude [V]');
set(get(AX(2),'Ylabel'),'String','Frequenz [Hz]');
title ('\bf u-Signal und Frequenzanstieg ber die Zeit');
end
```

4.1.2 Frequenzverlauf aus dem Spektrogram

Listing 2: frequenz durch Spektogramm

```
% Funktion zur Errechnung der Frequenzen durch das Spektogramm

function frequenz_durch_Spektogramm(x_t,t)


N=length(t);
fs=N/max(t);

wnsize=256;
wnoverlap= 250;
nr_abtastwerte_frequenz= 256;

[S,F,T]=spectrogram(x_t,wnsize,wnoverlap,nr_abtastwerte_frequenz,fs);

%spectrogram(x_t,wnsize,wnoverlap,nr_abtastwerte_frequenz,fs);

groesse = size(S);
maxfrequ = ones(1,groesse(1))*-9;
maxfrequ_umgerechnet = ones(1,groesse(1))*-9;
maxindx = ones(1,groesse(1))*-8;
maxindx_umgerechnet = ones(1,groesse(1))*-8;

for i=1:groesse(2)
    [maxfrequ(i) maxindx(i)] = max(abs(S(:,i)));
    maxfrequ_umgerechnet(i) = F(round(maxindx(i)+1));
    maxindx_umgerechnet(i) = T(round(i));
end

maxindx_umgerechnet

figure(204);
[AX H1 H2] = plotyy(t,x_t,maxindx_umgerechnet,maxfrequ_umgerechnet,'plot','stem');
% AXIS([0 2 -1.1 1.1]);
xlabel('Zeitachse [s]');
set(get(AX(1),'Ylabel'),'String','Amplitude [V]');
set(get(AX(2),'Ylabel'),'String','Frequenz [Hz]');
title('\bf Chirp-Signal und Frequenzanstieg ber die Zeit');
end
```

4.1.3 Algorithmus zur Drehfrequenzberechnung

Listing 3: Algorithmus zur Drehfrequenzberechnung

```
%MDV Praktikum 7 Vorbereitungsaufgabe 4 - Testprogramm

clear all; clc; close all;

%Messwerte laden
messung1 = load('MotorStrom_10V_100kS.mat');
messung2 = load('MotorStrom_20V_100kS.mat');
messung3 = load('MotorStrom_30V_100kS.mat');

% Strme und Tachos identifizieren
strom1 = messung1.strom;
```

```
tacho1 = messung1.tacho;
strom2 = messung2.strom;
tacho2 = messung2.tacho;
strom3 = messung3.strom;
tacho3 = messung3.tacho;

%Versorgungsspannungen
A1 = 10;
A2 = 20;
A3 = 30;

f_T  = 100000;
T_ges = 1/f_T;

%plottet  Spektren der strme
[y_DFT_abs_10V_strom f_DFT_10V_strom]= MotorStrom_Amplitudenspektrum(strom1,T_ges,f_T,10,'b',1);
[y_DFT_abs_20V_strom f_DFT_20V_strom]= MotorStrom_Amplitudenspektrum(strom2,T_ges,f_T,10,'b',2);
[y_DFT_abs_30V_strom f_DFT_30V_strom]= MotorStrom_Amplitudenspektrum(strom3,T_ges,f_T,10,'b',3);

%findet Index vom hchsten Peak
[maxwert10_strom maxind10_strom] = max(y_DFT_abs_10V_strom);
[maxwert20_strom maxind20_strom] = max(y_DFT_abs_20V_strom);
[maxwert30_strom maxind30_strom] = max(y_DFT_abs_30V_strom);

%berechnet entsprechende Drehzahl
Drehzahl_Motor_10V_strom = abs(f_DFT_10V_strom(maxind10_strom))/18
Drehzahl_Motor_20V_strom = abs(f_DFT_20V_strom(maxind20_strom))/18
Drehzahl_Motor_30V_strom = abs(f_DFT_30V_strom(maxind30_strom))/18

%plottet  Spektren der tachos
[y_DFT_abs_10V_tacho f_DFT_10V_tacho]= MotorStrom_Amplitudenspektrum(tacho1,T_ges,f_T,10,'b',4);
[y_DFT_abs_20V_tacho f_DFT_20V_tacho]= MotorStrom_Amplitudenspektrum(tacho2,T_ges,f_T,20,'b',5);
[y_DFT_abs_30V_tacho f_DFT_30V_tacho]= MotorStrom_Amplitudenspektrum(tacho3,T_ges,f_T,30,'b',6);

%findet Index vom hchsten Peak
[maxwert10_tacho maxind10_tacho] = max(y_DFT_abs_10V_tacho);
[maxwert20_tacho maxind20_tacho] = max(y_DFT_abs_20V_tacho);
[maxwert30_tacho maxind30_tacho] = max(y_DFT_abs_30V_tacho);


%berechnet entsprechende Drehzahl
Drehzahl_Motor_10V_tacho = abs(f_DFT_10V_tacho(maxind10_tacho))/8
Drehzahl_Motor_20V_tacho = abs(f_DFT_20V_tacho(maxind20_tacho))/8
Drehzahl_Motor_30V_tacho = abs(f_DFT_30V_tacho(maxind30_tacho))/8


% Drehzahl_Motor_10V_tacho/Drehzahl_Motor_10V_strom
% Drehzahl_Motor_20V_tacho/Drehzahl_Motor_20V_strom
% Drehzahl_Motor_30V_tacho/Drehzahl_Motor_30V_strom

Drehzahl_Motor_10V_tacho/Drehzahl_Motor_10V_strom
Drehzahl_Motor_20V_tacho/Drehzahl_Motor_20V_strom
Drehzahl_Motor_30V_tacho/Drehzahl_Motor_30V_strom
```

4.2 Codes aus Termin 8

4.2.1 Funktion haardec.m

Listing 4: Funktion haardec.m

```
% Vorbereitungsaufgabe 1.1 Termin 8
function [u,v] = haardec_8_1_1(x)
% function [u,v] = haardec(x)
% ---------------------------------------------------------------
% filename :        haardec
% author:             g   Dogan, Timo Lausen, Boris Henckell
% organisation :    TU Berlin
% p r o j e c t :   MDV PR
% date :            04.07.2012
% ---------------------------------------------------------------
% description :      fhrt  einen Zerlegungsschritt
%                   der schnellen Haartransformation durch
% input :           x - zu zerlegendes Signal
% output :          u - Approximationen
%                   v - Details
% ---------------------------------------------------------------

N = length(x);
u = ones(1,N/2)*-99;
v = ones(1,N/2)*-88;
j=1;
for i=1:N/2
    u(i) = (x(j)+x(j+1))/sqrt(2);
    v(i) = (x(j)-x(j+1))/sqrt(2);
    j=j+2;
end
```

4.2.2 Funktion haardeclevel.m

Listing 5: Funktion haardeclevel.m

```
% Vorbereitungsaufgabe 1.2 Termin 8
function S = haardeclevel_8_1_2(x,lvl)
% function [S] = haardeclevel(x, lvl )
% ---------------------------------------------------------------
% filename :        haardeclevel
% author:             g   Dogan, Timo Lausen, Boris Henckell
% organisation :    TU Berlin
% p r o j e c t :   MDV PR
% date :            04.07.2012
% ---------------------------------------------------------------
% description :      fhrt  die schnelle Haartransformation
%                   bis zu einem vorgegebenen
%                   Skalierungslevel durch
% input: x    ?     zu zerlegendes Signal
%        lvl  ?     Skalierungslevel
% output: S   ?     Matrix mit Skalierungen und Details
%                   Dimensionen: lvl+1: Signalle
%                   Jede Zeile   enth   die Approximationen
%                   gefolgt von den Details eines Levels
% ---------------------------------------------------------------
N = length(x);
```

```
S = ones(lvl+1,N)*−77;

S(1,:) = x;
u = x;
for i=2:lvl+1
    [u v] = haardec_8_1_1(u);
    N2 = length(u);
    S(i,:) = [u,v,S(i−1,2*N2+1:N)];
end
```

4.2.3 Funktion getAppDet.m

Listing 6: Funktion getAppDet.m

```
% Vorbereitungsaufgabe 1.3 Termin 8
function [u,v] = getAppDet_8_1_3(S, lvl )
% function [u,v] = getAppDet(S, lvl )
% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
% filename :          getAppDet
% author:              g    Dogan, Timo Lausen, Boris Henckell
% organisation :      TU Berlin
% p r o j e c t :     MDV PR
% date :              04.07.2012
% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
% description :        extrahiert die Approximationen
%                      und details eines Levels
% input :              S    ?   Matrix mit Signalzerlegeung
%                      lvl  ?   Skalierungslevel
% output :             u    ?   Approximationen
%                      v    ?   Details
% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
[M N] = size(S);
u = S(lvl+1,(1:N/(2^lvl)));
% v = S(lvl+1,(N/(2^lvl)+1:N));
v = S(lvl+1,(N/(2^lvl)+1:2*N/(2^lvl)));
```

4.2.4 Funktion Daubechies_Wavelets.m

Listing 7: Funktion Daubechies_Wavelets.m

```
%MDV Praktikum 8 Vorbereitungsaufgabe 3
%Daubechies−Wavelets
%Bemerkung: Anzahl der verschwindenden Momenten variierbar durch
%db1,db4,db15


function [approx details] = Daubechies_Wavelets(x,lvl, db)

%Skalierungslevel variabel
N = lvl;

%mittels wavedec: Daubechies−Wavelet ('db1'−wavename)
% C      = [app. coef.(N)|det. coef.(N) |...  |det. coef.(1)]
% L(1)   = length of app. coef.(N)
% L(i)   = length of det. coef.(N−i+2) for i = 2,..., N+1
% L(N+2) = length(X)
```

```
[C,L] = wavedec(x,N,db);

%mittel appcoef: Berechnung der App.Koeffizienten in Level N
%gibt Vektor approx der  Lnge  L(1) mit den Koeffizienten aus
approx = appcoef(C,L,db,N);

%mittels detcoef: Berechnung der Det.Koefizienten in Level N
%gibt Vektor details der  Lnge  L(1) mit den Koeffizienten aus
%braucht keine Wavelet-Vorgabe mehr, nur Ergebnisse aus wavedec
details = detcoef(C,L,N);
```

4.2.5 Vorbereitungsaufgaben Plots

Listing 8: Aufrufe für die Plots

```
% Vorbereitungsaufgaben 8

close all; clc, clear;

load('strom');

Bilder_abspeichern = 0;

S = haardeclevel_8_1_2(x,5);

fignum=801;
for lvl=1:5
    %% errechnen der Approximation und Details des jeweiligen Levels
    [u v] = getAppDet_8_1_3(S, lvl);

    [ud vd] = Daubechies_Wavelets(x,lvl,'db3');
    [ud1 vd1] = Daubechies_Wavelets(x,lvl,'db1');
    [ud5 vd5] = Daubechies_Wavelets(x,lvl,'db5');
    [ud10 vd10] = Daubechies_Wavelets(x,lvl,'db10');
    [ud15 vd15] = Daubechies_Wavelets(x,lvl,'db15');

    %% Strecken des Approximation und der Details
    N=length(u);
    u2=ones(1,N*2^lvl);
    v2=ones(1,N*2^lvl);

    ud2  = ones(1,N*2^lvl);
    vd2  = ones(1,N*2^lvl);
    ud21 = ones(1,N*2^lvl);
    vd21 = ones(1,N*2^lvl);
    ud25 = ones(1,N*2^lvl);
    vd25 = ones(1,N*2^lvl);
    ud210 = ones(1,N*2^lvl);
    vd210 = ones(1,N*2^lvl);
    ud215 = ones(1,N*2^lvl);
    vd215 = ones(1,N*2^lvl);


    j=1;
%   Die Approximation und die Details auf die  Le  von x normieren
    for i=1:N
        for k=1:2^lvl
            u2(j)    = u(i);
```

```
            v2(j)     = v(i);
            ud2(j)    = ud(i);
            vd2(j)    = vd(i);
            ud21(j)   = ud1(i);
            vd21(j)   = vd1(i);
            ud25(j)   = ud5(i);
            vd25(j)   = vd5(i);
            ud210(j) = ud10(i);
            vd210(j) = vd10(i);
            ud215(j) = ud15(i);
            vd215(j) = vd15(i);
            j = j+1;
        end
    end

    %% plotten der Approximation und der Details
    %Haar-Wavelet
    figure(fignum);
    hold on
        plot(x)
        plot(u2,'r')
        plot(v2,'c')
    hold off
    title (['\bf Haar Wavlet Stufe',num2str(lvl)]);
    xlabel('Zeit [s]');
    ylabel('Amplitude [A]');
    legend('Originalsignal','Approximation','Details');

    if Bilder_abspeichern == 1
        figure(fignum);
        name=['../../ Bilder/Termin8/Haar_Wavlet_lvl_',num2str(lvl),'.pdf'];
        print('-painters','-dpdf','-r600',name)
    end

    fignum= fignum+1;

    % Daubechies-Wavelet

    figure(fignum);
    hold on
        plot(x)
        plot(ud2,'r')
        plot(vd2,'c')
    hold off
    xlabel('Zeit [s]');
    ylabel('Amplitude [A]');
    legend('Originalsignal','Approximation','Details');

    title (['\bf Daubechies Wavlet Stufe ',num2str(lvl)]);

    if Bilder_abspeichern == 1
        figure(fignum);
        name=['../../ Bilder/Termin8/Daubechies_Wavlet_lvl_',num2str(lvl),'.pdf'];
        print('-painters','-dpdf','-r600',name)
    end

    fignum= fignum+1;

    % 1db Daubechies-Wavelet
```

```
figure(fignum);
hold on
    plot(x)
    plot(ud21,'r')
    plot(vd21,'c')
hold off
xlabel('Zeit [s]');
ylabel('Amplitude [A]');
legend('Originalsignal','Approximation','Details');


 title (['\bf Daubechies Wavlet 1dB Stufe ',num2str(lvl)]);

 if Bilder_abspeichern == 1
     figure(fignum);
     name=['../../Bilder/Termin8/Daubechies_Wavlet_1db_lvl_',num2str(lvl),'.pdf'];
     print('-painters','-dpdf','-r600',name)
end

fignum= fignum+1;

% 5db Daubechies-Wavelet

figure(fignum);
hold on
    plot(x)
    plot(ud25,'r')
    plot(vd25,'c')
hold off
 title (['\bf Daubechies Wavlet 5dB Stufe ',num2str(lvl)]);
xlabel('Zeit [s]');
ylabel('Amplitude [A]');
legend('Originalsignal','Approximation','Details');

 if Bilder_abspeichern == 1
     figure(fignum);
     name=['../../Bilder/Termin8/Daubechies_Wavlet_5db_lvl_',num2str(lvl),'.pdf'];
     print('-painters','-dpdf','-r600',name)
end

fignum= fignum+1;

% 10db Daubechies-Wavelet

figure(fignum);
hold on
    plot(x)
    plot(ud210,'r')
    plot(vd210,'c')
hold off

xlabel('Zeit [s]');
ylabel('Amplitude [A]');
legend('Originalsignal','Approximation','Details');

 title (['\bf Daubechies Wavlet 10dB Stufe ',num2str(lvl)]);

 if Bilder_abspeichern == 1
```

```
            figure(fignum);
            name=[' ../../ Bilder/Termin8/Daubechies_Wavlet_10db_lvl_',num2str(lvl),'.pdf' ];
            print( '-painters', '-dpdf','-r600',name)
        end

        fignum= fignum+1;

        % 15db Daubechies-Wavelet

        figure(fignum);
        hold on
            plot(x)
            plot(ud215,'r' )
            plot(vd215,'c' )
        hold off

        xlabel(' Zeit [s] ' );
        ylabel(' Amplitude [A]');
        legend('Originalsignal' , 'Approximation','Details ' );



         title ([ ' \bf Daubechies Wavlet 15dB Stufe ',num2str(lvl)]);

         if  Bilder_abspeichern == 1
            figure(fignum);
            name=[' ../../ Bilder/Termin8/Daubechies_Wavlet_15db_lvl_',num2str(lvl),'.pdf' ];
            print( '-painters', '-dpdf','-r600',name)
        end

        fignum= fignum+1;


end


wn = ones(1,length(x));
fs  = 1;

Spektrum(x, wn, fs,  1, 'b',-0.6,0.6,-100,30, fignum);

         if  Bilder_abspeichern == 1
            figure(fignum);
            name=[' ../../ Bilder/Termin8/Spektrum.pdf'];
            print( '-painters', '-dpdf','-r600',name)
        end

fignum=fignum+1;
figure(fignum);
spectrogram(x,25,24)

         if  Bilder_abspeichern == 1
            figure(fignum);
            name=[' ../../ Bilder/Termin8/Spectrogam.pdf'];
            print( '-painters', '-dpdf','-r600',name)
        end
```