M1 „DSV mit Matlab - Anwendung der DFT/FFT“

Inhalt:

In diesem Versuch soll zunächst mit einfachen Beispielen das Anwenden der FFT mit Matlab geübt werden. Als Voraussetzung wird die Einführungsübung zu Matlab (M0) angenommen.

Allgemeine Hinweise:

1. **Login** in den Labors LSV und LUT: user: ***asdf*** und password: ***asdf***;
2. Innerhalb des Desktop-Ordners „Eigene Dateien“ sollten sie sich ein eigenes Arbeitsverzeichnis anlegen!
3. Nach dem Aufruf von Matlab (Desktopsymbol) stellen sie bitte das **Arbeitsdirectory (in Matlab)** auf das eigene, soeben erstellte, Sub-Directory ein!
4. **!!! Bitte bei den Folgeterminen USB-Stick mitbringen, damit Sie sich Ihre erstellten Files mitnehmen können!!!**
5. **NETZWERK-Laufwerk:**
   Normalerweise sollte auf den Praktikumsrechnern ein Netzverzeichnis „dsv-prakt“ oder „dsv-praktikum“ (meist LW X:, Y: oder Z:) gemountet sein! (Bei Problemen Betreuer fragen)
   Dort ist diese **Praktikumsanleitung** in einer read-only Word- bzw. Open-Office-Version übers Netz verfügbar!
6. Sie können diese Datei während des Versuchs öffnen und unter sofort unter eigenem Namen im <Eigene Dateien>\<eigener Name> Verzeichnis mit „save as..“ abspeichern. Die zu bearbeitenden Fragen können dann direkt in der Datei editiert werden und die in Matlab erreichten Ergebnisse und Plots mit **Cut&Paste** zur **Dokumentation und Protokollerstellung** einfügt werden.
7. Ebenso können die in der Anleitung enthaltenen Code-Beispiele mit Cut&Paste (<CTRL c> und <CTRL v> in den Matlab-Editor übernommen und modifiziert und angepasst werden.

V

Mit V gekennzeichnete Teile bitte vorbereiten!

1 Approximation der Fourierreihe eines periodischen Rechtecksignales mittels FFT

1.1 Vorbereitung:

Berechnen Sie zunächst den Betrag der komplexen Fourierkoeffizienten des in Bild 1 skizzierten periodischen Rechtecksignals mit normierter Amplitude A=1000 und tragen Sie den Wert **der ersten 11 Fourierkoeffizienten $|c_k|$** für k=0,1...10 in Tabelle 1 ein.

(Hinweis; F-Reihe aus dem Skript oder Formelsammlung entnehmen; komplexe Koeff. verwenden!)

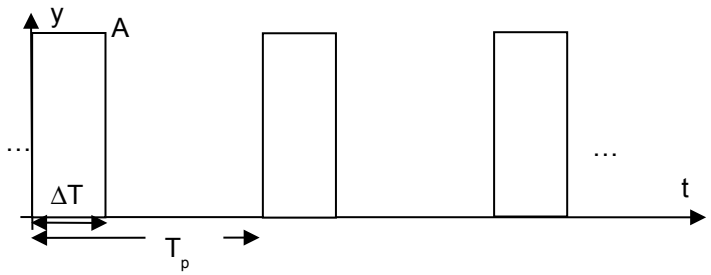


Abbildung 1 Periodischer Rechteckpuls;

$\frac{\Delta T}{T_p} = \frac{1}{4}$; $f_p = 1/T_p = 1\,kHz$

Formel $j\frac{A}{T\,k\,\omega}\left[e^{-jkT\omega}-1\right]$

Y_k= .....

V

k	0	1	2	3	4	5	6	7	8	9	10
|Y| berechnet	250	225.08	159.15	75	0	45	53	32	0	25	32

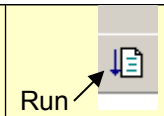
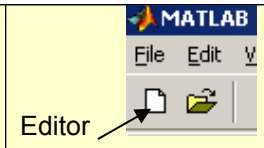
Tabelle 1

1.2 Erzeugung des diskreten Mustersignals y1 per Simulation

Zunächst soll gezeigt werden, wie ein einzelner Rechteckimpuls der oben gegebenen Folge mit MATLAB realisiert werden kann:
Es soll ein Zeitausschnitt über eine Periode der Dauer T_P (siehe Bild 1) diskret mit L=16 Samples erzeugt werden:

V

==> Welche Abtastfrequenz ist hierfür notwendig: f_{a1}= 16kHz=L/Tp

Hinweis: Für die weitere Bearbeitung ist es sinnvoll in Matlab mit einem sogenannten Skript-File zu arbeiten. Rufen Sie hierzu den in Matlab eingebauten internen Editor auf (z.B. durch Anklicken des Leeren-Blatt-Symbols links oben) und editieren Sie im Skript-file! Ein Ausführen des Skripts ist direkt im Debug-Modus mit dem „Run“ - Symbol möglich		

```
% MATLAB: File  rechteck
fa1= …….;
L=16;
k=[0:L-1];     %Index
t=k*1/fa1;      % Zeitvektor d. Länge 1 ms
%               Ausgangsvektor y, erzeugt durch logische
% Verknüpfung mit dem Zeitvektor t:
y=[t<0.25e-3];
% heißt: es wird ein y Vektor mit gleicher
% Länge wie t erzeugt und für jedes Element
% in t die Bedingung t<0.25e-3 geprüft; an den
% Indizes, wo Bedingung erfüllt ist, wird y=1; sonst 0
y=A*y;  %Amplitude berücksichtigen
stem(t,y);   xlabel('t/ms \rightarrow');   ylabel('y_k');
```

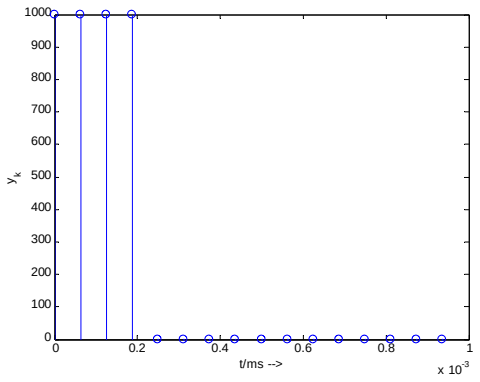


Abbildung 2 Erzeugung des diskreten Mustersignals

In dem Beispiel ist nur *eine* Möglichkeit aufgezeigt, wie das diskrete Signal y erzeugt werden kann. Welche Möglichkeiten könnten Sie sich noch vorstellen? (Grundlegende Matlab-Kenntnisse vorausgesetzt!)

.→ Erzeugen Sie das zeitdiskrete Signal y_k nach obenstehender oder eigener Methode.
(Hinweis: Erzeugen Sie den gewünschten Code im Editor und speichern Sie das file unter dem Namen Rechteck.m) im eigenen Directory ab.

V

Wenn diese diskrete Folge der Länge L=16 nun zur FFT Analyse verwendet wird, welches diskrete Signal $y_{d1}(t)$ wird dann eigentlich analysiert, und wie sieht das zugehörige Linienspektrum (Betrag) in einem erweiterten Frequenzbereich aus?

abgetasteter Rechteckimpuls ↔ periodische Si Funktion

Ta = 1ns/16 = 62.5us

fa = 1/Ta = 16kHz

.....

Bitte skizzieren das Ergebnis dieser Überlegung in untenstehendem Diagramm (Abb. 3) und tragen Sie Abtastperiode T_{A1} und -frequenz f_{A1} ein.

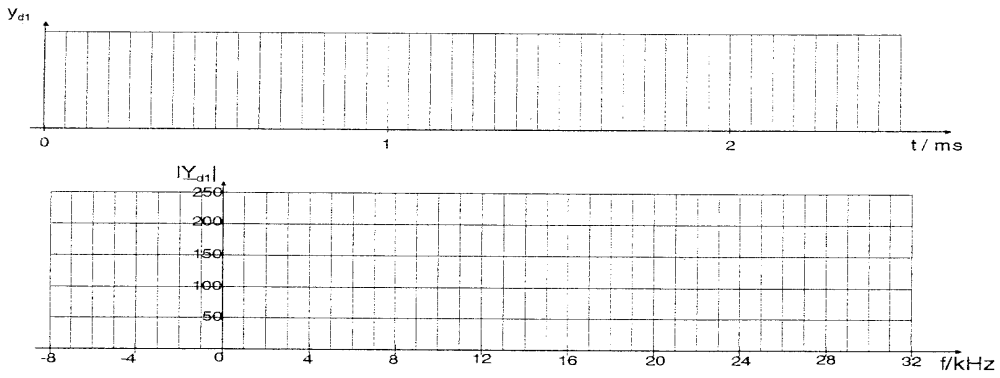


Abbildung 3 Zeitdiagramm und Linienspektrum des abgetasteten Rechteckpulses

1.3 Analyse des Mustersignals y1 (eine Periode) mit (DFT) FFT

Da eine Länge **von L=16=2^4** für das Mustersignal gewählt wurde, wird hier der FFT Algorithmus verwendet.
Es sollen die in Tabelle 1 schon theoretisch ermittelten Fourierkoeffizienten nun durch FFT approximiert werden:

Syntax: *Yd1=**korr***abs(fft(y)); f= 16000; stem(f,Yd1)*

→ Wie groß ist der ***Korrektur- bzw. Skalierungsfaktor***, der hier noch berücksichtigt werden muss, um physikalisch richtige Amplitudenwerte im Linienspektrum zu erhalten?
korr=1/N=0,0625
.....
Zur korrekten graphischen Ausgabe ist es zweckmäßig, einen Frequenzvektor f zu verwenden; mit welcher Syntax kann man diesen erzeugen?:
f= 0 … 15
.....
→ Erzeugen sie das mittels FFT ermittelte Linienspektrum und plotten Sie es in Abb 4; (Hinweis: Mit Hilfe der Befehle *„axis([0 fa 0 250]);grid;“* erhalten Sie eine sinnvolle Darstellung im Bereich 0…..fa)
→ Übertragen Sie die als Approximation erhaltenen Fourier-Koeffizienten $|Y_{d1}|$ in Tabelle 2:
(Verwenden sie entweder den "Workspace-Browser" oder die Graphik zum Ablesen der Werte)
→ Wo und warum weichen die Koeffizienten aus Tabelle 1 $|Y_{d1}|$ und $|Y_k|$ aus Tabelle 2 voneinander ab?
Weichen ab wegen Aliasing Fehler ab ca. zweitem Wert

Ausfüllen während des Praktikums

```
figure;
korr= 0.0625;
fa2 = 16;
Yd1=korr*abs(fft(y));     f= [0:15];
stem(f,Yd1);
xlabel('f/kHz \rightarrow');
ylabel('y_k');
axis([0 fa2 0 250]);
grid;
```

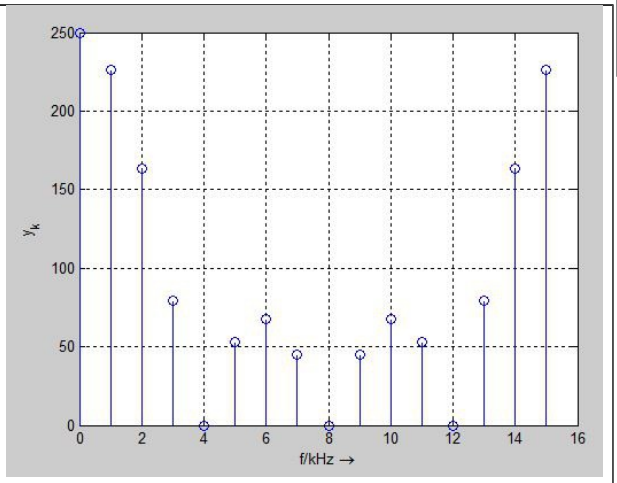


Abbildung 4

k	0	1	2	3	4	5	6	7	8	9	10
|Yd1| simuliert	250	226.53	163.32	79.55	0	53.15	67.65	45.06	0	45.06	67.65
|Yd2| aus Aufg. 1.4 simuliert	250	225.17	159.41	75.30	0	45.47	53.83	32.80	0	25.24	33.15

Tabelle 2 Durch Simulation gewonnene Foourierkoeffizienten

1.4 Veränderung der Abtastfrequenz: Mustersignal y2 (1 Periode)

→ Mit welcher Maßnahme lassen sich eben festgestellte Abweichungen von $|Y_{d1}|$ reduzieren?

Abtastfrequenz erhöhen

.....

→ Modifizieren Sie den erstellten Programmcode (s. Abb.2) so dass eine bessere Approximation der Koeffizienten erreicht wird. ***(Änderung des Parameters um einen Faktor/Divisor 4!)***

- andere Abtastfrequenz fa2=4*fa1
- welche Variable muss sonst verändert werden? y, korr, t
-

→ Stellen Sie die neue Musterfolge $y_2(n)$ und $|Y_2|$ unten dar und ergänzen Sie Tabelle 1 an den relevanten Stellen.

```
fa2= 64000;  L2=64;
A=1000;
k2=[0:L2-1];      %Index
t2=k*1/fa2;
y2=[t2<0.25e-3];
y2=A*y2;

figure;
korr3 = 0.015625;
fa3 = 64;
Yd3=korr3*abs(fft(y2));
f= [0:63];
stem(f,Yd3);
xlabel('f/kHz
\rightarrow');
ylabel('y_k');
axis([0 fa3 0 250]); grid;
```

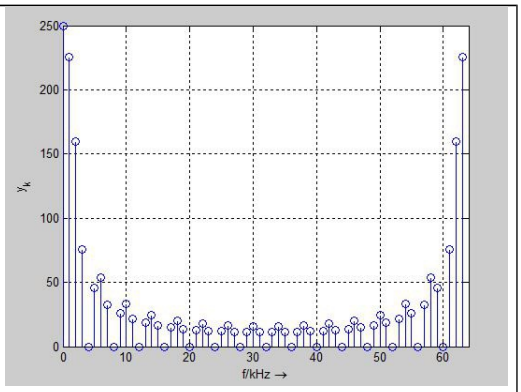


Abbildung 5 (Code , Zeitfolge , Frequenz

1.5 Verlängerte Messzeit über 4 Perioden : Mustersignal y3

→ Was bringt es, wenn die ursprüngliche Folge aus Abb. 1 statt mit einer Periode beispielsweise mit 4 Perioden dargestellt wird? (Abtastfrequenz f_a wie ganz am Anfang!!)

→ Erzeugen Sie hierzu noch mal die Folge y1 und verlängern sie durch periodisches hintereinander setzen. In Matlab kann dies ganz einfach folgendermaßen erreicht werden:

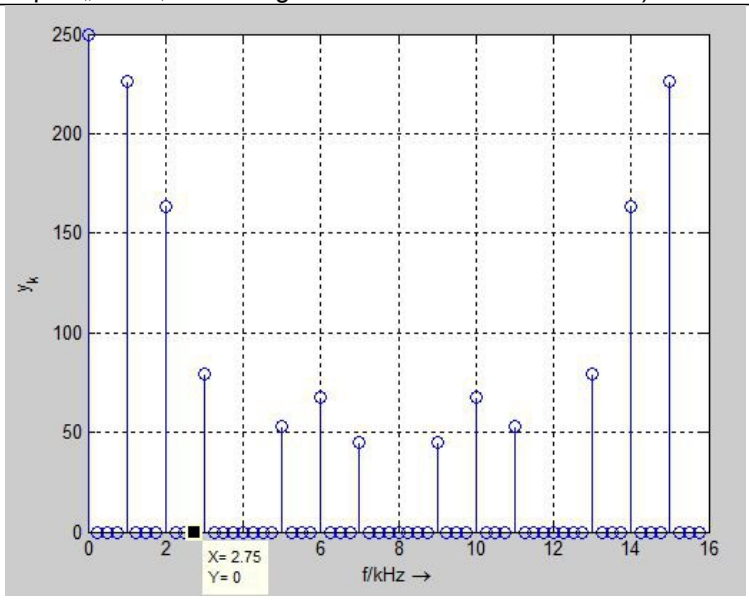
y=... *% Ausgangsfolge*
y3=[y y y]; *% viermal hintereinandergesetzt; Länge jetzt 4*L (Zeit- und Frequenzvektoren müssen nach der Verlängerung nochmals neu berechnet werden!!*

(Hinweis: Kontrolle der Folge durch Zeitplot „stem“; Änderung des Korrekturfaktors für FFT!)

```
y3=[y y y y];
fa2 = 16;
L=4*L;
k=[0:L-1];      %Index
f=k*fa1/L/1000;

figure;
korr3 = 1/L;

Yd3=korr3*abs(fft(y3));
% f= [0:63];
stem(f,Yd3);
xlabel('f/kHz
\rightarrow');
ylabel('y_k');
axis([0 fa2 0 250]);
grid;
```



Ausfüllen während des Praktikums

→ Welche Auswirkungen ergeben sich im Frequenzbereich ?

- allgemein: höhere Auflösung im Frequenzbereich,

- für die approximierten Fourierkoeffizienten: keine Verbesserung der Genauigkeit (wie bei Bsp.1)

1.6 Auswertung einer nicht-ganzzahligen Anzahl von Perioden/ y4

Verwenden Sie die Musterfolge y3 vom vorherigen Abschnitt. Sie sollte die Länge 64 haben!

→ Verkürzen sie die Folge um vier Samples, indem Sie einfach die 4 hinteren Werte ausschneiden!
Mögliche Syntax: L=length(y3) %(sollte 64 sein) ; L=L-4 ; y4=y3(1 :L)

→Verändert sich etwas an der Energie des Signals?

→ Analysieren sie y4 ebenfalls mittels FFT Befehl, berücksichtigen Sie die neue Länge des Auswertefelds L=L-4 ; und generieren Sie für den Plot einen neuen Frequenzvektor !

Hinweis: da L nun keine 2-er Potenz ist, rechnet Matlab automatisch mit dem DFT Algorithmus; vielleicht bemerken Sie die erhöhte Rechenzeit!?

Syntax: *Yd4=korr*abs(fft(y4)); f=k*fa1/L; ; stem(f,Yd4);*

Welche Veränderung ergibt sich im Linienspektrum?

Leckeffekt, da Anzahl der Samples nicht mehr ein Vielfaches der Periode ist.
Amplituden bei f = 0 ist größer, da die letzten 4 Werte fehlen (die Null sind)

Versuchen Sie die Werte für die Fourierkoeffizienten aus dem Linienspektrums-Plot herauszulesen und vergleichen Sie mit Tabelle 1!

Welcher Effekt zeigt sich? → Leckeffekt

Ausfüllen während des Praktikums

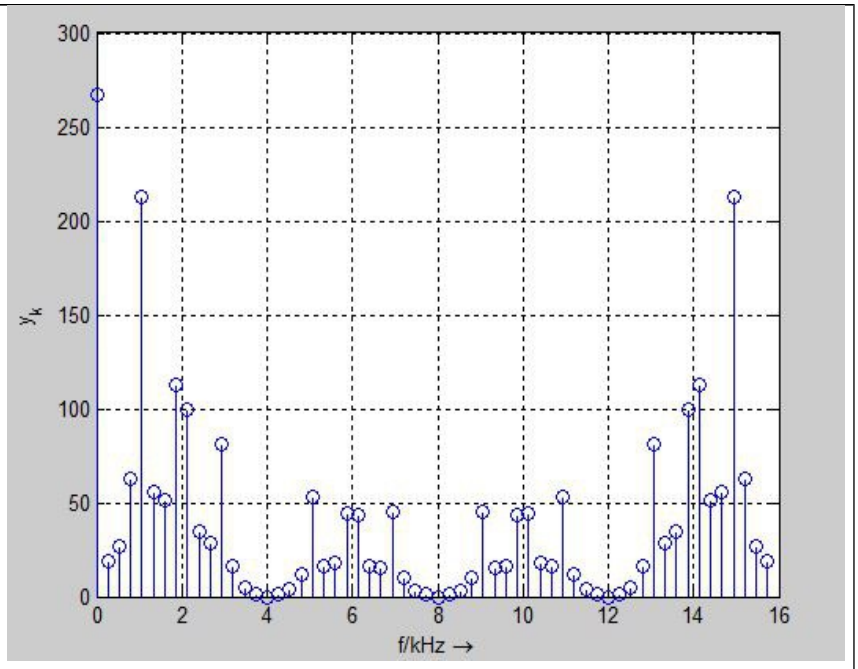
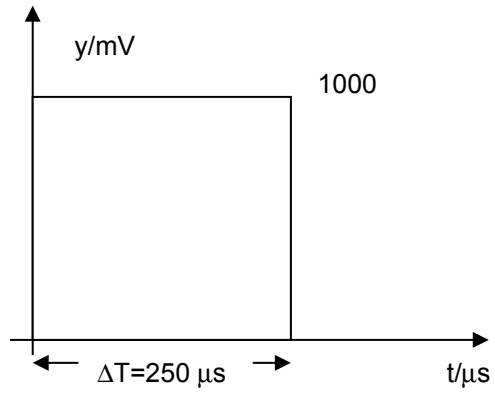


Abbildung 6 Code/ Betragsspektrum y4

2 Fourier-Spektrum eines einzelnen Rechteckimpulses

2.1 Vorbereitung

Geben Sie das komplexe Fourierintegral Y(f) des nebenstehenden kausalen Rechteckimpulses mit den richtigen Maßeinheiten an und skizzieren untenstehend Sie das Amplitudendichtespektrum |Y(f)|



V

Y(0)=.....µV/Hz

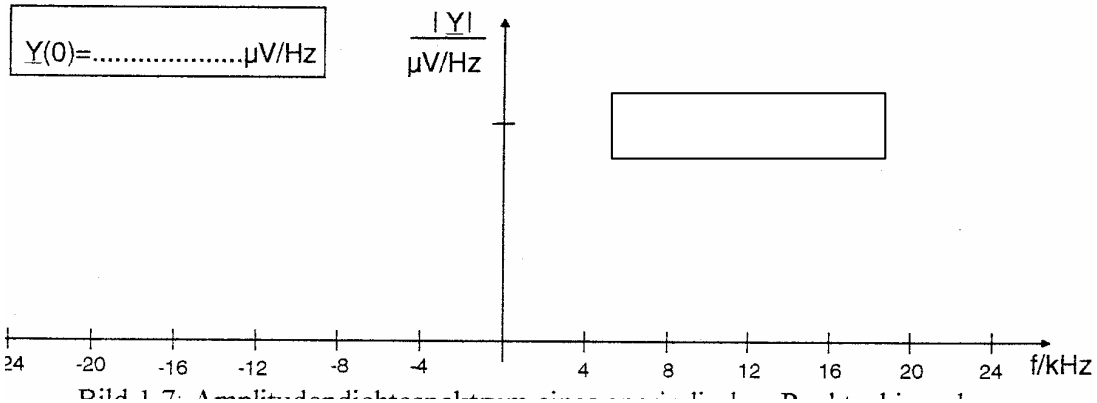


Abbildung 7: Theoretisch ermitteltes Betragsspektrum des Impulses

2.2 Approximation des kontinuierlichen Amplitudenspektrums mittels FFT

Verwenden Sie 16 Abtastwerte zur Simulation des Impulses!

→ daraus folgt fa= 64kHz

→ Was muss man tun, damit man nach der FFT in der graphischen Darstellung des approximierten kontinuierlichen Spektrums eine gute Auflösung erzielt ?

Zeropadding

→ Wie muss das FFT-Ergebnis skaliert werden, um eine physikalisch korrekte Approximation zu erreichen:

Darstellung des halben Frequenzbereichs, da Spiegelung an fa/2.

Verwenden Sie eine FFT der Ordnung r=9; das heißt N=2^9=512!
Plotten Sie immer nur die ersten 256 Werte! (z.B. *plot(f(1:256),y(1:256))*

Versuchen Sie zum Vergleich die theoretisch berechenbare Amplitudendichte in dem selben Plot darzustellen. (si-Funktion)

```
Fa1= 64000;  L=2^9;
A=1000;  k=[0:L-1];      %Index
t=k*1/fa1;
y=[t<0.25e-3];
y=A*y;
plot(t,y);
axis([0 1/fa1*100 0 1050]);
grid;

%% Grafik 1
figure;
korr= 1/fa1;
Yk=korr*abs(fft(y));
df= fa1/L;
f=k*df;

Ytheo = abs(A*0.25e-
3*sinc(f*0.25e-3));

zz= [Yk(1:L/2);Ytheo(1:L/2)]';
plot(f(1:L/2)',zz);
grid;
```

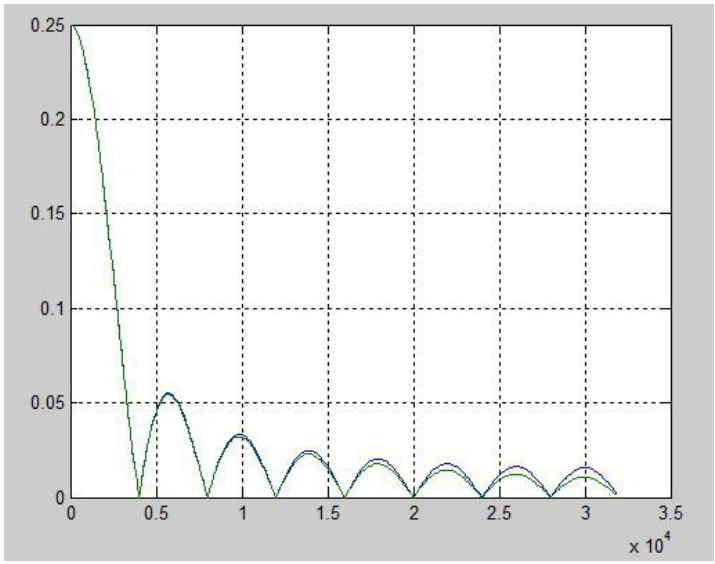


Abbildung 8 Code und simuliertes und theoretisch berechnetes Spektrum

3 Detektion von Sinussignalen, Fensterung

Es soll demonstriert werden, mit welchen Methoden man bei der FFT Analyse z.B. 2 unterschiedlich starke Sinustöne mit leicht unterschiedlicher Frequenz noch sinnvoll analysieren kann.

3.1 Vorbereitung

Es soll ein Signal mit folgender Syntax erzeugt werden: (rand ist ein Zufallszahlengenerator)

N=256; fa=2560.;
k=[0:N-1];
y=100*sin(pi*1980*k/fa)+2*sin(pi*2020*k/fa)+2*rand(1,N)-1;

V ? Was für ein Signal wird hier erzeugt? Frequenzen? Amplituden?
Sinus f1 = 990Hz, A1 = 100
f2 = 1010, A2 = 2
DC A3 = -1
Rauschen A4 = 2

V ? Welcher Pegelunterschied der zwei Sinusanteile in dB ergibt sich?

Pegeldifferenz in dB = 20log10(100/2)dB = 34,0 dB

3.2 Fall A; Abtastfrequenz ist frei wählbar

Die Signalfrequenzen f1 und f2 sind bekannt; Abtastfrequenz fa ist passend wählbar.

→Wählen Sie die kleinstmöglichen Werte für fa und Länge N der FFT (bzw. Ordnung (r=ldN)), so dass f1 und f2 auf Auswertefrequenzen zu liegen kommen und bei Rechteckfenster **kein** Leckeffekt auftritt!!
fA= N * Δf (> 2020) = 2560 Hz N=256 Δf = 10 Hz

→ Simulation mit oben vorgeschlagenem Code und Ermittlung des Pegelunterschieds aus dem Spektrum! (Hierzu Darstellung logarithmisch und auf das Maximum bezogen)

```
Yk=abs(fft(y));
Ymax=max(Yk)            %Maximum feststellen
Yk=20*log10(Yk/Ymax);
f=k*fa/N;
plot(f,Yk,'bo-'); grid; axis([0 fa/2 -100 0]);
```

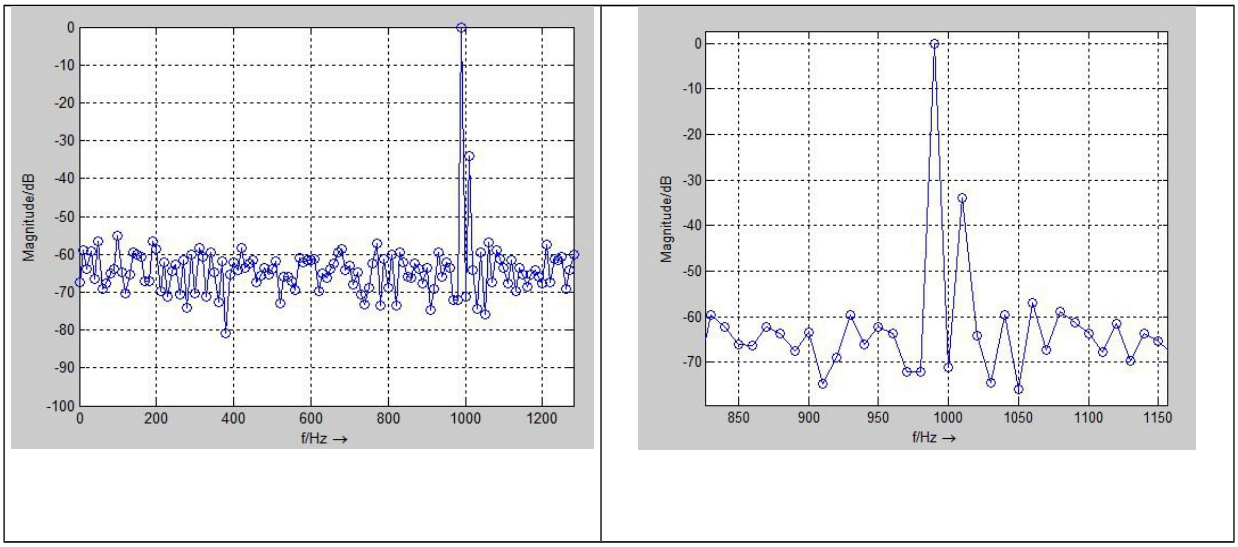


Abbildung 9 Betragsspektrum und Vergrösserung um den Peak

3.3 Fall B; Abtastfrequenz ist fest vorgegeben

Die Signalfrequenzen f1 und f2 sind nach wie vor bekannt; Abtastfrequenz **fa** ist nun **fest** vorgegeben**: fa=4000 Hz**

→ Experimentieren Sie mit der Länge des FFT Feldes (Ordnung) und verschiedenen Fensterfunktionen indem sie das Signal mit Fensterfunktionen gewichten, bis im Amplitudenspektrum f1 und f2 deutlich getrennt detektierbar sind!

Folgende Fensterfunktionen stehen zur Verfügung:
BARTLETT, BLACKMAN, BOXCAR, CHEBWIN, HAMMING, HANN, KAISER, TRIANG.
(Hinweis: Die verschiedenen Fenster können,mit dem GUI „wintool“ analysiert werden!)

Syntax z.B.: *y=y.*hann(N)' (Achtung : hier muss der Fenstervektor transponiert werden (Apostroph); sonst gibt es Fehlermeldung, da y bisher Zeilenvektor war und window(N) eine Spaltenvektor ergibt)*

Ergebnisse:

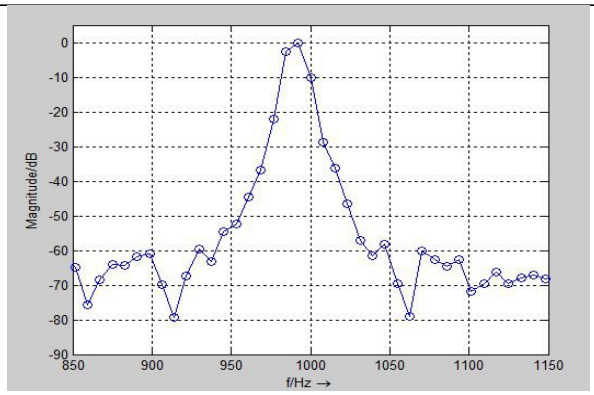
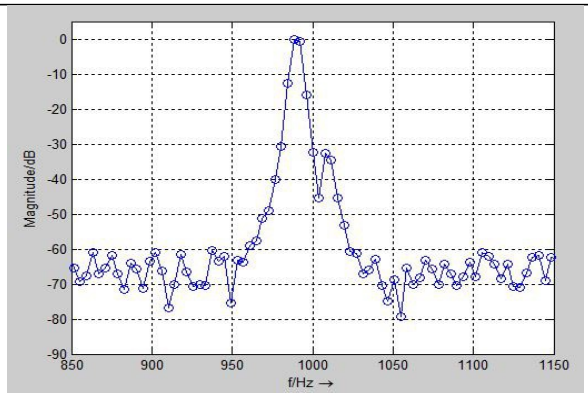
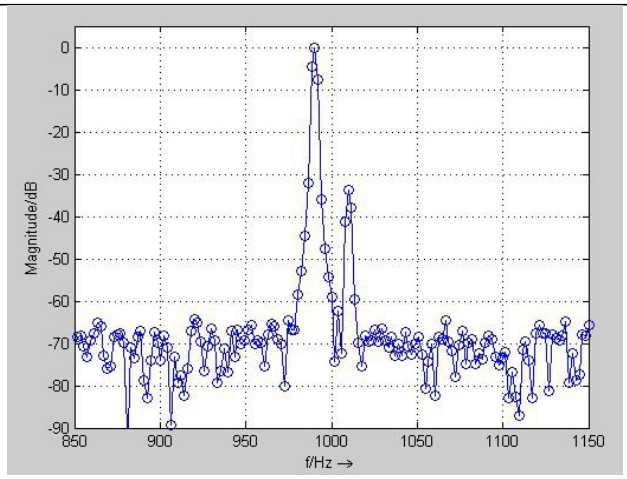
N=512	Hannfenster ..	
N=1024	Hannfenster	
N=2048	Hannfenster	

Abbildung 10 Betragsspektrum für verschiedene Varianten von N und der Fensterfunktion

3.4 Matlab Animation (Ev. Demo in der Vorlesung):

Folgende Demo erzeugt eine kleine Matlab-Animation:
Es wird ein Sinussignal erzeugt und mittels einer 128 Punkt – FFT und wahlweise mit Fensterung spektral ausgewertet. Dabei wird in 65 Schritten die Signalfrequenz von 1000 bis 1200 Hz erhöht. Abtastfrequenz fa=4000 Hz ist fest für alle Schritte. Für jeden Schritt wird ein Spektrumsplot erzeugt. Alle 65 Plots können dann mittels des „*movie*“ Befehls als Animation abgespielt werden.

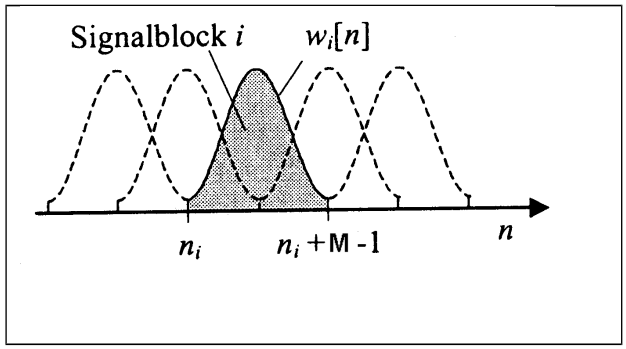
Folgenden Code können Sie direkt mit Copy&Paste in ein Matlab-Editor Fenster übernehmen:

```
% Movie zur FFT Analyse
n=65; fa=4000; N=128; k=[0:N-1]; t=k/fa; f=k*fa/N;
%
f0=fa/4;                                %Startfrequenz
for j=1:n
    fs=f0 +fa/(N*10)*(j-1);
   y=sin(2*pi*fs*t)+0.00001*rand(1,N);
%
% hier Fenster wahlweise auskommentieren
y=y.*hann(N,'periodic')';
%y= y.*hamming(N,'periodic')';
    Yk=1/N*abs(fft(y));
    Ymax=max(Yk);                       %Maximum feststellen
    Yk=20*log10(Yk);
   plot(f,Yk,'bo-'); grid;
   axis([0 fa/2 -100 0]);
   M(j) = getframe;
end
movie(M,1,5)  % movie wird zum Schluss 3 Mal gespielt mit 5 Frames per second
```

4 Kurzzeit-Spektralanalyse

Zufällige Signale, wie zum Beispiel Audiosignale, weisen oft charakteristische Merkmale zum Beispiel im Frequenzbereich, nur über kurze Zeit auf. Im Gegensatz zu einem „mittleren Leistungsdichtespektrum“ kann man relativ kurze Signalausschnitte des Signals mittels FFT analysieren und somit den zeitlichen Verlauf des spektralen Inhalts verfolgen. Eine wichtige Anwendung findet sich zum Beispiel in der Sprach (bzw. Audio)-Codierung oder in der Spracherkennung.

Eine theoretisch unendlich lange Eingangsfolge wird in „Segmente“ bzw. Blöcke der Länge M unterteilt (M = normalerweise 2’er Potenz), wobei von jedem der Teilsegmente über die FFT ein Betragsspektrum ermittelt wird. Die FFT-Verarbeitung der Segmente geschieht üblicherweise mit einer der bekannten Fensterfunktionen; wobei die verarbeiteten Segmente sich zeitlich überlappen können! (Bild) Die zeitliche Überlappung kann z.B. sinnvollerweise M/2 betragen!



Die derart ermittelten Teilspektren werden nun „hintereinander“ im so genannten 3-Dimensionalen **Wasserfall-Diagramm** dargestellt.
Eine alternative Darstellung ist die Darstellung als farblich codierte Balkenspektren in 2 Dimensionen. Beides nennt man ***Spektrogramm***.

Am folgenden Beispiel wird dies nicht mit einem zufälligen Signal, sondern an einem Sinus-förmig modulierten FM-Signal demonstriert:

→ Sie können das Code-Beispiel direkt in den Matlab-Editor übernehmen und selbst ausführen.

Ein FM Moduliertes Signal mit Sinussignal als Information wird erzeugt. Um das (im Mittel) zu erwartende Besselspektrum zu erreichen, werden die Trägerfrequenz ft und die Signalfrequenz fs so gewählt, dass alle Linien auf einer FFT-Auswertefrequenz zu liegen kommen!
Die im ersten Programmteil erfolgte FFT Auswertung über den gesamten langen Ausschnitt des Signals ergibt das erwartete Besselspektrum ohne Leckeffekt!!
Der im Spektrogramm dargestellte Verlauf zeigt den typischen Verlauf der „Augenblicksfrequenz“ mit sinusförmiger Schwankung im Zeitbereich!

→ Versuchen Sie sich im Beispielprogramm zurechtzufinden! Variieren Sie zum Beispiel die Trägerfrequenz auf ft=7/16*fa!

- Welcher Effekt zeigt sich im Spektrogramm??
-

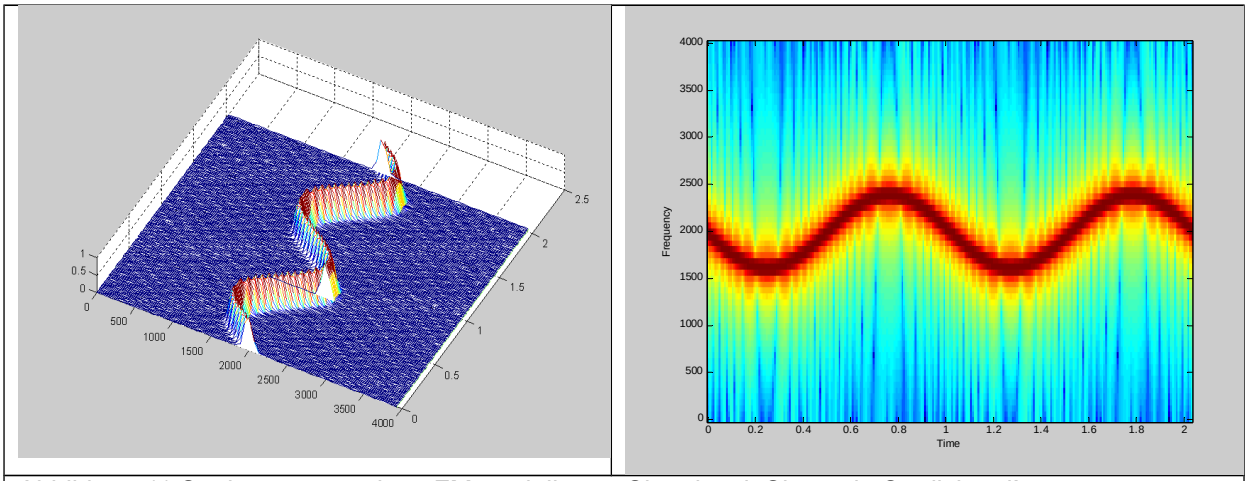


Abbildung 11 Spektrogramm eines FM modulierten Signals mit Sinus als Quellsignal!

```
%Beispiel-Code:
fa= 8000; ta=1/fa; N=2^14 ; time=[0:N-1]*ta;      % Abtastfrequenz und Zeitraster für N Werte
ft=fa/4;                                          %Trägerfrequenz in der Mitte des Basisbereichs
df=fa/N;              f=[0:N-1]*df ;                      % Frequenzvektor

% Bestimmung einer geeigneten Signalfrequenz fs:
% Da das erwartete Spektrum ein Bessellinienspektrum um ft herum
% mit Linien im Abstand fs sein soll, ist es sinnvoll die vorkommenden
% Frequenzen auf Auswertefrequenzen zu legen, d.h. fs ist ein ganzzahliges
% Vielfaches von df
fs=1;                         % Vorgabe Signalfrequenz
fs=df*round(fs/df)                  % Auf Auswertefrequenzen angepasste Signalfrequenz
%
% Signalvektor erzeugen:
etha=400;                     % Modulationsindex
hub=etha*fs                   % Daraus resultierender Hub; einseitig!!
yfm=cos(2*pi*ft*time + etha*cos(2*pi*fs*time));

% Betrags-Spektrum
Yspek=abs(fft(yfm));    Yspek=Yspek/max(Yspek);
%
% Zweigeteilten Plot erzeugen
subplot(2,1,1);  plot(time,yfm); subplot(2,1,2);  plot(f,Yspek);
%
%  jetzt Ermittlung der Kurzzeitspektren jeweis über
%  eine Segmentlänge von N2=512 Werten;
N2=128;   w2=hanning(N2);   % Fensterfunktion für Kurzzeitspektren
i2=0:N2-1;   df2=fa/N2;     f2=i2*df2;
% Zeitlicher Überlapp  der verschiedenen Fenster M= N2/2;
M=N2/2;
% somit können
Nseg=N/(N2/2) -1;              % Segmente ausgewertet werden
%
Y3D=zeros(Nseg, N2/2);     % Speicherplatz reservieren!!
%
% Schleife über Nseg Segmente;
for k=1:Nseg;
   i_min=(k-1)*M+1; i_max=i_min+N2-1;       % Start - Stop - Index
   YY= fft(w2'.*yfm(i_min:i_max));
   Y3D(k,:)=abs(YY(1:M));                   % nur pos. Basisbereich weiterverarbeiten
end;
time2=[0:Nseg-1]*M*ta;
Ynorm=max(max(Y3D));                        % maximum suchen
Y3D=Y3D/Ynorm;       f2b=f2(1:M);
figure;      h=waterfall(f2b,time2,Y3D);    view(30,30);    %Wasserfall-Plot
```

```
%
figure;    specgram(yfm,N2,fa);    % Matlab Internes Spectogramm
```