# Modul 5033 Digitale Signalverarbeitung Dozent: Rolf Vetter

### *Praktische Übung No8*

# Entwurf digitaler Filter

### Lernziele

* Nach dem Bearbeiten dieser praktischen Übung können Sie
  + Entwurfsvorschriften für digitale Filter im Frequenzbereich mittels Toleranzschemas erläutern
  + die Fenstermethode zum Filterentwurf von FIR Filtern erklären und programmieren
  + die IIR-Filter mittels Bilinear Transformation entwerfen
  + mit dem MATLAB Programm fdatool oder Linienbefehlen Filter entwerfen
  + den Unterschied zwischen Butterworth-, Chebyshev- und Cauer (elliptischen)-Tiefpassfilter erklären

### Übung

Interessierte Studierende können den Filterentwurf selber programmieren, d.h. die Aufgaben 1) und 2) Schritt per Schritt ausführen. Die anderen können diese Aufgaben mit dem fdatool und den Linienbefehlen fir1 und butter von MATLAB lösen (siehe Aufgabe 3).

1. **IIR-Filterentwurf Entwurf FIR Filter mittels Fenstermethode**

Schreiben Sie ein MATLAB Skript FIR\_Entwurf\_IFFT\_Fensterung für den Entwurf eines FIR-Filters mittels Abtastung des idealen Frequenzgangs, der inversen FFT und anschliessender Kaiser-Fensterung mit dem Parameter β und einer Länge N für das folgende Toleranzschema:

Durchlassbereich

* + Normierte Grenzfrequenz 0.1
  + Rippel 6dB
* Sperrbereich
  + Normierte Grenzfrequenz 0.2
  + Abschwächung 60dB

Überprüfen Sie den Frequenzgang des entworfenen Filters mit der Funktion freqz().

*Frequenzgang und Impulsantwort Ihres Filters. Ist das Toleranzschema erreicht?*

Wie kann man mit dieser Entwurfsmethode ein Filter mit einem Rippel im Durchlassbereich von 3dB erzeugen?

*Antwort in einem Satz.*

1. **IIR-Filterentwurf mit Bilinear-Transformation**

Schreiben Sie ein MATLAB Skript IIR\_Entwurf\_Bilinear für den Entwurf eines digitalen IIR-Tiefpassfilter der 2. Ordnung mit einer normierten Grenzfrequenz von 0.25.

1. Berechnen Sie die Grenzfrequenz fc,a des zu transformierenden analogen Filters bei einer Abtastfrequenz 8000Hz.

*Berechnung der korrigierten analogen Grenzfrequenz.*

1. Entwerfen Sie ein kontinuierliches Butterworth Filter der 2. Ordnung mit Grenzfrequenz fc,a.

*Geben Sie H(s) an*

1. Transformieren Sie das kontinuierliche Filter in ein digitales Filter mit Hilfe der Bilinear-Transformation bilinear()und überprüfen Sie die Grenzfrequenz.

*Frequenzgang Ihres Filters. Ist die Grenzfrequenz korrekt?*

1. Entwerfen Sie direkt ein digitales Butterworth Filter mit dem Befehl butter().
2. Berechnen Sie die Bilinear-Transformation eines analogen Filters der 1. Ordnung mit Grenzfrequenz fc,a und einer Abtastfrequenz fa analytisch. Überprüfen Sie Ihr Resultat für fc,a=3Hz und fa=21Hz mit dem der MATLAB Funktion bilinear().

1. **IIR-Filterentwurf mit fdatool**

Führen Sie den Filterentwurf der Aufgaben 1 und 2 mit dem fdatool und den Linienbefehlen fir1 und butter von MATLAB lösen aus. Erzeugen Sie ein gaussverteiltes weisses Rauschsignal x(n), wenden Sie die Filter auf x(n) und beobachten Sie die spektrale Leistungsdichte des Ausgangssignals.

*Darstellung der Leistungsdichte des gefilterten Signals und Erklärung*

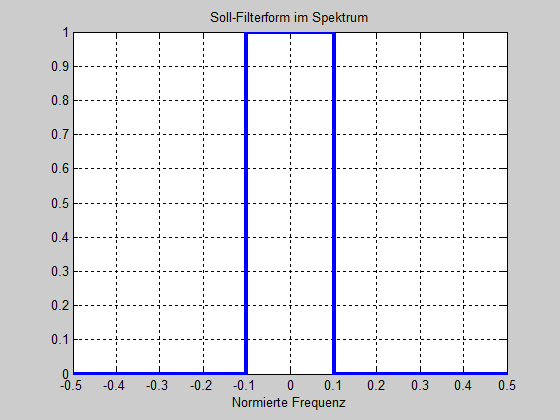
Ergebnisse

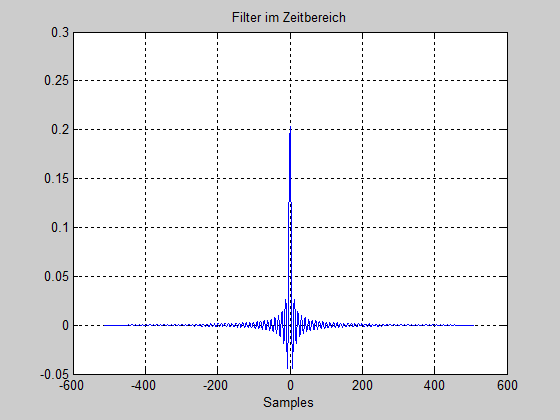
# 1

Code:

|  |
| --- |
| close all;  clear;  N = 1024;  f = linspace(-0.5, 0.5, N)';  f\_grenz\_filtering = 0.102;  f\_grenz\_soll = 0.1;    dw= 1/N;  dt = 2\*pi/(N\*dw);  t = linspace(-0.5\*N, 0.5\*N-1, N)';    filterSpektrum = (abs(f) <= f\_grenz\_filtering);    filterZeit = real(ifftshift(ifft(fftshift(filterSpektrum))));    plot(t, filterZeit)  grid on  title('Filter im Zeitbereich');  xlabel('Samples');      b = 4;  n\_fenster = 80;  fenster = kaiser(n\_fenster, b);    filterGefenstert = zeros(N,1);  filterGefenstert(N/2-(n\_fenster/2-1):N/2+n\_fenster/2) = filterZeit(N/2-39:N/2+40) .\* fenster;    figure  plot(filterGefenstert)  grid on  title('Filter im Zeitbereich (mit Nullen ergänzt)');  xlabel('Samples');    filterkern = fftshift(fft(fftshift(filterGefenstert)));    figure  plot(f, 20\*log10(filterkern))  grid on  title('Filter im Spektrum');  xlabel('Genormte Frequenz');  ylabel('dB');    % Überprüfung des Spektrums mit der internen Funktion Freqz  figure  freqz(filterGefenstert, 1, 1024, 1)  title('Filter im Spektrum (mit freqz');  %% Filterqualität beurteilen aufgrund der Bandspezifikationen  ripple\_durchlassband = min(real(20\*log10(filterkern(find(abs(f) <= f\_grenz\_soll)))))  daempfung\_stopband = max(real(20\*log10(filterkern(find(abs(f) >= f\_grenz\_soll + 0.1))))) |

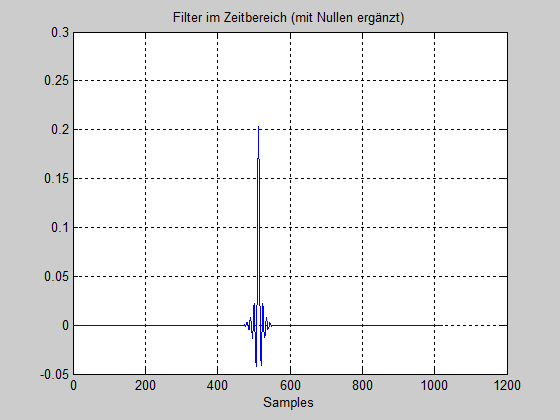
Zuerst definieren wir eine soll-filterform im Spektrum:





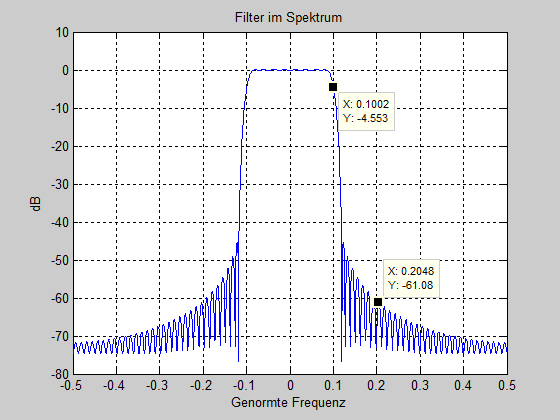
Das in den Zeitbereich transformierte filter ist sehr „lang“ -> wir müssen es abschneiden.

* wir schneiden es mit einem Filter ab -> kaiser



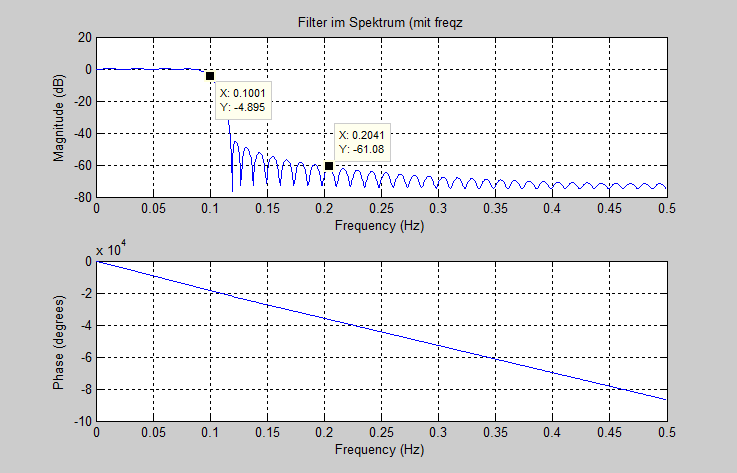
Dann haben wir das Filter noch mit Nullen ergänzt.

Jetzt wird das Filter wieder in das Spektrum transformiert:



Wir erfüllen die Anforderungen. wir haben einen Rippel von -4.5530 dB und eine Dämpfung im Stopband von 61.0755 dB.

Die Kontrolle mit freqz ergibt genau das gleiche:



### 3dB Rippel Anforderungen:

Wenn wir statt 6dB 3dB wollen, müsste wahrscheinlich die Filterlänge erhöht werden. Dadurch werden die Nebenlappen unterdrückt und das Filter kann etwas mehr nach aussen geschoben werden. Dann fängt das Abfallen der Filterkurve erst NACH den 0.1Hz an.

# 2

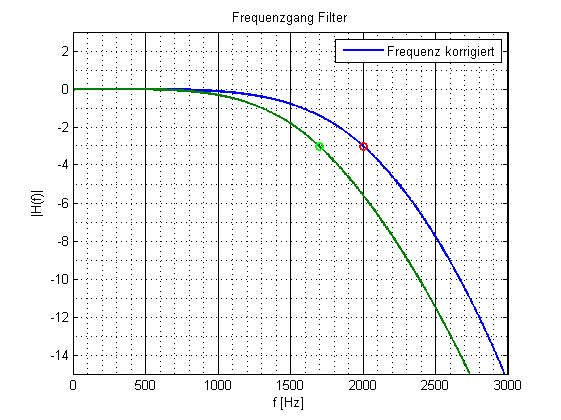
Korrigierte Grenzfrequenz: fpass=fa/pi\*tan(pi\*fdpass/fa); = 2547Hz (statt 2000Hz)

H(s) = 2.56e008

----------------------------

s^2 + 2.263e004 s + 2.56e008

Kontrolle Grenzfrequenz:

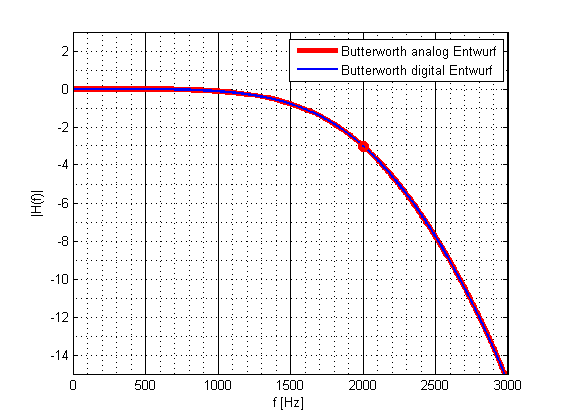


Bemerkung: wenn wir die Frequenz mit der Tangensfunktion korrigieren, ist die Grenzfrequenz genau bei den gewünschten 2000Hz. Ohne Korrektur haben wir eine Abweichung von 300Hz.

Lösung direkt digitales Filter mit butter():

|  |
| --- |
| %------ ---------Toleranzschema des digitalen Filters----------------------------  Apass=3;fdpass=2000; % Abschwächung und Grenzfrequenz des Durchlassbereichs  Astop=10;fdstop=3000; % Abschwächung und Grenzfrequenz des Sperrbereichs  fa=8000; % Abtastfrequenz  % Bestimmung der analogen Grenzfrequenzen (Kompensation der  % Frequenzverzerrung der Bilinear-Transformation)  fpass=fa/pi\*tan(pi\*fdpass/fa);  fstop=fa/pi\*tan(pi\*fdstop/fa);  %-------------------Berechnung der Ordnung des analogen Filters---------------------  % siehe Kurs Signale und Systeme mit MATLAB  Ordnung\_toleranzschema=buttord(fpass,fstop,Apass,Astop,'s');  Ordnung\_aufgabenstellung = 2;  %-------------------Bestimmung der Koeffizienten des analogen Filters---------------  [b,a]=butter(Ordnung\_aufgabenstellung,2\*pi\*fpass,'s');  tf(b,a)  %-------------------Bilinear-Transformation-----------------------------------------  [bd,ad]=bilinear(b,a,fa);  %-------------------Überprüfung und Anzeige-----------------------------------------  nfft=4\*256;  [H,f]=freqz(bd,ad,nfft,fa);  plot(f,20\*log10(abs(H)),'r', 'Linewidth',4)  set(gca,'ylim',[-Astop-5 3]);grid minor;ylabel('|H(f)|');xlabel('f [Hz]')    %-------------------Bilinear-Transformation für digitales Filter-----  [bd,ad]=butter(Ordnung\_aufgabenstellung,fdpass/(fa/2));  hold all  grid on  [H,f]=freqz(bd,ad,nfft,fa);  plot(f,20\*log10(abs(H)),'Linewidth',2)  legend('Butterworth analog Entwurf', 'Butterworth digital Entwurf') |

Ergebnis:



Die Variante direkt über die digitalen koeffizienten (ohne den Parameter ‚s‘ bei butter() ) liefert nun genau den selben Frequenzgang wie die analoge Variante. ACHTUNG: wir mussten als Frequenz für die digitale Filtersynthese fdpass/(fa/2) einsetzen, da MATLAB ein Interval von [0 1] (entspricht 0 bis *Nyquistfrequenz*) erwartet.

Analytische Bilineartransformation:

fc = 3Hz, fa = 21Hz

s = 2\*fa\*(1-z^-1)/(1+z^-1)

H(s) = 1/(sRC+1) (s einsetzen)

H(z) = 1( pi\*fc/fa\*(1-z^-1)/(1+z^-1) ) (noch ausmultiplizieren)