MATLAB - Filterentwicklung

Themen: Filterentwurf und Filtereigenschaften

1. Entwurf eines FIR-Filters nach der Fenstermethode

a) Es ist ein FIR-Filter zu entwickeln (fdatool) mit folgenden Kennwerten:
- f_{Sample} = 44.1kHz
- Tiefpass mit Grenzfrequenz f_G=1500 Hz
- Bei f_{Stop} = 1600Hz soll die Dämpfung mindestens 40dB betragen.
- Probieren Sie verschiedene Window-basierte Entwurfsmethoden (rect, hamming, chebychev) aus. Nehmen Sie den Entwurf, der die Spezifikation mit möglichst geringer Filterordnung (= Länge der FIR-Faltungsmaske) erfüllt.
- Speichern Sie das TP-Filter als „FIR_1500_40dB“ ab (File→Export→als MAT-File).
- Dokumentieren Sie die Einstellungen und den Frequenzgang (Screenshot).
- Geben Sie die Filterordnung an.
- Ist das Filter „linearphasig“ (konstante Gruppenlaufzeit)?

b) Schreiben Sie ein Matlab-Script mit folgenden Merkmalen:

Über eine Variable Select (= 1 … 4) kann zwischen 4 Signalquellen ausgewählt werden.
Select == 1: wav-Datei „ACDClike.wav“
Select == 2: chirp-Signal (44.1kHz Samplerate, 0…5s, 50 …5000 Hz)
Select == 3: 5s Rauschen (Befehl „rand(…)“ mit 44.1 kHz)
Select == 4: Rechteckimpulsfolge

```
elseif Select == 4
    t = 0 : 1/44100 : 1;          % 44.1kHz sample freq für 1s
    d = 0 : 1/20 : 1;             % 20 Hz Wiederholrate für 1s
    s1 = pulstran(t, d, 'rectpuls', 0.005); % Rect der Breite 5ms
end
```

Das Filter kann in Matlab geladen werden mit:

```
load('FIR_1500_40dB');
```

Das ausgewählte Signal wird dann gefiltert mit:

```
% Anwenden des FIR-Filters
s2 = filter(FIR_1500_40dB, 1, s1);
```

Das Ergebnis wird geplottet.
Die Signale werden normiert.
Das Originalsignal und das Ergebnis werden mit sound(..) ausgegeben.

c) Dokumentieren Sie das Ausgangssignal (Screenshot) für das Chirpsignal.
Dokumentieren Sie das Ausgangssignal für die rect-Impulsfolge (Auschnittvergrößerung).

Anforderungen:
- Gute Variablennamen sowie Kommentare
- keine „*Magic Numbers*“

Abnahme: Code-Vorführung,
Protokoll mit Screenshots und kurze Erklärungen

2. Entwurf eines IIR-Filters mit Second-order-Sections

a) Es ist ein IIR-Filter zu entwickeln (fdatool) mit folgenden Kennwerten:
- f_{Sample} = 44.1kHz
- Tiefpass mit Grenzfrequenz f_G=1500 Hz
- Bei f_{Stop} = 1600Hz soll die Dämpfung mindestens 80dB betragen.
- Probieren Sie verschiedene Filterkonzepte (Butterworth, Chebychev, Elliptic)
- Nehmen Sie den Entwurf, der die Spezifikation mit möglichst geringer Filterordnung erfüllt.
- Speichern Sie das TP-Filter als „IIR_1500_80dB“ ab (File→Export).
  Das Filter wird in Form von L Second-Order-Sections (SOS) abgespeichert. Jede Zeile in der Filtermatrix beschreibt eine SOS.

$$sos = \begin{bmatrix} b_{01} & b_{11} & b_{21} & 1 & a_{11} & a_{21} \\ b_{02} & b_{12} & b_{22} & 1 & a_{12} & a_{22} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{0L} & b_{1L} & b_{2L} & 1 & a_{1L} & a_{2L} \end{bmatrix}$$

- Dokumentieren Sie die Einstellungen (Screenshot) und den Frequenzgang.
- Geben Sie die Filterordnung an.
- Ist das Filter „linearphasig“ (konstante Gruppenlaufzeit)?

b) Schreiben Sie ein Matlab-Script mit folgenden Merkmalen:

Über eine Variable *Select* (= 1 … 4) kann zwischen 4 Signalquellen ausgewählt werden.
Select == 1: wav-Datei „ACDClike.wav“
Select == 2: chirp-Signal (44.1kHz Samplerate, 0…5s, 50 …5000 Hz)
Select == 3: 5s Rauschen (Befehl „rand(…)“ mit 44.1 kHz)
Select == 4: Rechteckimpulsfolge

Das ausgewählte Signal wird dann gefiltert mit:

```
%s2 = sosfilt(IIR_SOS, s1);
s2 = sosfilt(SOS_IIR_1500_80dB, s1);
```

Das Ergebnis wird geplottet.
Die Signale werden normiert.
Das Originalsignal und das Ergebnis werden mit sound(..) ausgegeben.

c) Dokumentieren Sie das Ausgangssignal (screenshot) für das Chirpsignal.
Dokumentieren Sie das Ausgangssignal für die rect-Impulsfolge (Auschnittvergrößerung).

Abnahme: Code-Vorführung
Protokoll mit Screenshots und kurze Erklärungen

3. Entwurf eines IIR-Filters und rekursive Realisierung

a) Es ist ein IIR-Filter zu entwickeln (fdatool) mit folgenden Kennwerten:
- f_{Sample} = 44.1kHz
- Tiefpass mit Grenzfrequenz f_G=1400 Hz
- IIR 2. Ordnung, Elliptisch, A_{Stop}=40dB
- Speichern Sie die Filterkoeffizienten ab (File→Export).
  Das Filter wird in SOS-Form abgespeichert.

b) Über eine Variable *Select* (= 1 … 4) kann zwischen 4 Signalquellen ausgewählt werden.
Select == 1: wav-Datei „ACDClike.wav“
Select == 2: chirp-Signal (44.1kHz Samplerate, 0…5s, 50 …5000 Hz)
Select == 3: 5s Rauschen (Befehl „rand(…)“ mit 44.1 kHz)
Select == 4: Rechteckimpulsfolge

Das ausgewählte Signal soll dann mit Hilfe eines selbstgeschriebenen rekursiven Algorithmus (s. Vorlesung) gefiltert werden:

Das Ergebnis wird geplottet.
Die Signale werden normiert.
Das Originalsignal und das Ergebnis werden mit sound(..) ausgegeben.

c) Dokumentieren Sie das Ausgangssignal (screenshot) für das Chirpsignal.
Dokumentieren Sie das Ausgangssignal für die rect-Impulsfolge (Auschnittvergrößerung).

Abnahme: Code-Vorführung
Protokoll mit Screenshots und kurze Erklärungen

MATLAB - Filterentwicklung

4. Entwurf eines Notch-Filters zur Unterdrückung sinusförmiger Störsignale

a) Es ist ein Notch-Filter zu entwickeln (fdatool) mit folgenden Kennwerten:
- f_{Sample} = 44.1kHz
- Störunterdrückung bei 880Hz (nur dort)
- Bandbreite 100Hz
- Speichern Sie die Filterkoeffizienten ab (File→Export).
  Das Filter wird in Direktform II abgespeichert.
- Dokumentieren Sie die Einstellungen (Screenshot) und den Frequenzgang (Zoom).

b) Schreiben Sie ein Matlab-Script mit folgenden Merkmalen:
- Einlesen des Gitarrentons „*GitRiff_880Hz.wav*“.
- Das gestörte Signal wird mit dem Notch-Filter gefiltert.
- Die Signale werden normiert.
- Das Originalsignal und das Ergebnis werden mit sound(..) ausgegeben.

Abnahme: Code-Vorführung
Protokoll mit Screenshots und kurze Erklärungen