Designing FIR Digital Filters

DSP Lab 12

Table of contents

1 Objective

Use filter design functions in Matlab and the Filter Design and Analysis tool quick FIR filter design and evaluation.

2 Theoretical notions

3 Zero-phase filtering

The function `filtfilt()` achieves zero-phase filtering of a vector `x` by filtering it twice:

- once in the normal direction (start to end)
- then flip the result and filter again (i.e. in the opposite direction)

It operates as following:

- the amplitude response is applied twice (i.e. the signal is multiplied with $|H(\omega)|^2$ instead of $|H(\omega)|^2$)
- the phase is canceled (zero-phase filter)

4 Matlab functions for filter design

4.1 FIR filter design

- `fir1()`: This function designs a finite impulse response (FIR) filter with a specified frequency response using the windowing method
- `firpm()`: This function designs a minimum-phase (linear-phase) FIR filter using a specified magnitude response.
- and several others

4.2 Examples

Linear-phase FIR filter with `fir1()`

The windowing method computes the impulse respose of an ideal filter, then applies a window to keep only a limited number of elements.

Examples:

```
b = fir1(48,[0.65], 'low');    % Design a low-pass filter, order 48,
                               % with specified cutoff frequency
freqz(b)
```

```
b = fir1(48,[0.4 0.7], 'bandpass');    % Design a band-pass filter, order 30,
                                       % with specified cutoff frequencies
fvtool(b)
```

Linear-phase FIR filter with `firpm()`

```
f   = [500 600] / 2000;     % Frequency bands: [0 to 500]/2000, [600 2000]/2000
a   = [1 0];                % Desired (ideal) amplitudes in the bands. Not in dB.
dev = [0.001, 0.01];        % Maximum allowed deviations from the desired amplitudes.
                            % Values not in dB

% Estimate filter order
[n,fo,ao,w] = firpmord(f,a,dev);

% Design filter
b = firpm(n,fo,ao,w);

% View frequency response
fvtool(b)
```

5 Exercises

5.1 Exercise 1 - Filtering an ECG signal

Design an FIR filter and use it to filter an ECG signal.

1. Load the ECG signal from the file `ECGsignal.mat` and display it in a subplot of a window.
2. Design four linear-phase FIR band-pass filters, of order at least 20, with the following pass bands:

   - 10 Hz - 40 Hz
   - 10 Hz - 100 Hz
   - 20 Hz - 40 Hz
   - 20 Hz - 100 Hz

   Use the `firpm()` function to design the filters. The sampling frequency of the ECG signal is 360 Hz.
3. Apply each of the four filters to the ECG signal.

4. Display the original ECG signal and the four filtered versions as five separate subplots of a window.

   The resulting plot should show the effects of the different filter pass bands on the ECG signal.

5.2 Exercise 2 - Zero-phase filtering

Let's investigate the delay introduced by the filters. We work on a copy of the previous exercise.

1. Is there a delay introduced by the filters? Measure the location of the R-peaks in the original vs filtered ECG signals.
2. Make all the filter orders twice as large. What happens to the delay?
3. Replace `filter()` with `filtfilt()` and regenerate the plot. What is the delay now?

5.3 Exercise 3 - Filtering an image

We apply FIR filters on an image, operating first on volumns, then on rows.

1. Load the `Lena512.bmp` image and display it.
2. Design a low-pass filter of order 25 and cutoff frequency 0.2, using `fir1()`.
3. Filter the image with this filter, using `filter`(), and display it. Do the filtering as follows:
   - filter very row in the image, separately
   - then filter every column in the resulting matrix.
4. Repeat the filtering using `filtfilt()` and display. Is there a difference?