AET 5420 Homework 3

Audio Signal Processing

Due: March 29, 2021

1 IMPLEMENTING IIR FILTERS IN THE TIME DOMAIN

As we have discussed in class, IIR filters can be implemented using time-domain processing based on a difference equation:

$$y[n] = \frac{1}{a_0}\Big(b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] + \ldots - a_1 y[n-1] - a_2 y[n-2] - \ldots\Big)$$

The coefficients for the feed-forward gains of the filter can be stored in an array:

```
b = [b0 b1 b2 b3 ..];
```

The coefficients for the feed-back gains of the filter can be stored in an array:

```
a = [a0 a1 a2 a3 ..];
```

The built-in MATLAB function for doing this processing has the syntax:

```
y = filter(b,a,x);
```

For this problem, you will recreate the behavior of this function by writing your own. In particular for Problem 1, you should use an implementation based on the time-domain difference equation. For Problem 2, you will repeat the task by using an implementation based on the frequency domain transfer function.

1.1 PROBLEM

Create and save a **function** (m-file) in MATLAB that performs frequency-domain IIR filtering:

- Name the function: `timeFilter.m`
- It should have the following input and output variables

  `[y] = timeFilter(b,a,x)`

  - `b`: array of feedforward coefficients
  - `a`: array of feedback coefficients
  - `x`: input signal
  - `y`: output signal
- Here are some guidelines and hints:
  - The output signal should end up being the same length as the input signal
  - The function should work with any length input signal
  - The function should work with arrays `b` and `a` of arbitrary length
  - It is not necessary for `b` and `a` to be the same length
- Use the same examples from Problem 1 to test your function
- It is recommended to create arrays to store the delayed samples of `x` and `y`.
- Use the provided test script to verify the performance of the function
  - Experiment with taking the impulse response of the function
  - Plot the results and compare to `freqz(b,a)`
  - Filter white noise and listen to the result

For this problem, you will submit the function file, **timeFilter.m**, and also **testScript.m**.

2 IMPLEMENTING IIR FILTERS IN THE FREQUENCY DOMAIN

IIR filters can also be implemented using frequency-domain processing based on a tranfer function:

$$Y[z] = H[z] \cdot X[z]$$
$$\text{Where: } H[z] = \frac{b_0 z^{-0} + b_1 z^{-1} + b_2 z^{-2} + \dots}{a_0 z^{-0} + a_1 z^{-1} + a_2 z^{-2} + \dots}$$

To perform frequency-domain filtering, a complex-valued filter, $H[z]$, must be created. This can be done by using the relationship between the Z-transform and the Fourier Transform:

$$z^{-d} = e^{-j \cdot 2\pi \cdot k \cdot d \cdot \frac{1}{N}}$$

Essentially, this substitution should be made in the transfer function for each term. As an example: $z^{-3} = e^{-j \cdot 2\pi \cdot k \cdot 3 \cdot \frac{1}{N}}$. Keep in mind, N is the total number of samples used for the FFT of $X[k]$ and k is an array of integers from $[0, 1, 2, ..., N-1]$. Therefore, the numerator of $H[k]$ is an array with length N after summing all of the terms. The same is true for the denominator. After determining the transfer function, element-wise multiplication can be used between $H[k]$ and $X[k]$ to calculate $Y[k]$, which is the frequency-domain version of the processed output signal.

2.1 Problem

Create and save a **function** (m-file) in MATLAB that performs frequency-domain IIR filtering:

- Name the function: `freqFilter.m`
- It should have the following input and output variables

  `[y] = freqFilter(b,a,x)`

  - `b`: array of feedforward coefficients
  - `a`: array of feedback coefficients
  - `x`: input signal
  - `y`: output signal
- Here are some guidelines and hints:
  - The output signal should end up being the same length as the input signal
  - The function should work with any length input signal
  - The function should work with arrays `b` and `a` of arbitrary length
  - It is not necessary for `b` and `a` to be the same length
- My recommendation would be to start this problem with a specific case, then build to the general solution. Examples:
  - `b = [1 -1], a = [1]`
  - `b = [1], a = [1 0.5]`
  - `b = [1 1], a = [1 0.5]`
  - `b = [0.2929, -.5858,.2929], a = [1, 0 , 0.1716]`
- The function should perform frequency-domain processing
  - X = fft(x)
  - y = real(ifft(Y))
  - N = length(x)
  - k has values 0, 1, 2, ... , N-1
- The function will need to create, `H`, based on `b` and `a`

    - It is recommended to create a numerator, `Hnum`, and denominator, `Hden`
    - $H = \frac{Hnum}{Hden}$

- You may choose to process each frequency bin, k, one at a time. Or, you could process the entire spectrum in one command.

- Use the provided test script to verify the performance of the function
    - Experiment with taking the impulse response of the function
    - Plot the results and compare to `freqz(b,a)`
    - Filter white noise and listen to the result

For this problem, you will submit the function file, **freqFilter.m**, and also **testScript.m**.

2.2 Submission

When completed, put the script, sound files, and your function in a compressed zip folder. Name the zip file: X_AET5420_Homework3.zip, where X is your last name. Then email the zip file to: eric.tarr@belmont.edu.