Chapter 7 Project Problem: Noise Equivalent Bandwidth

Background

In the design of communications and signal processing systems we often deal with filters implemented as an analog electrical circuit or digital filters implemented as algorithms in a computer program. Filters are generally used to remove an undesired signal(s), say interference and noise, and retain the desired signal(s).

Noise Equivalent Bandwidth

- When *white noise* (flat spectrum of frequencies like white light) is passed through a filter having a frequency response $H(f)$, some of the noise power is rejected by the filter and some is passed through to the output.

- The noise equivalent bandwidth is defined in the following picture.

![Diagram of noise equivalent bandwidth](image_url)

- $|H(f)|^2$
- Equal Areas Under Each Curve
- Noise Equivalent Bandwidth


---

Chapter 7: Project Problem
• For a digital filter, which operates with sampled data type signals, the maximum frequency of operation is limited to $F_s/2$, where $F_s$ is the sampling frequency in Hz.

• Taking into account the symmetry of the frequency response we can write that the noise equivalent bandwidth of a digital lowpass filter with unity gain at D.C. is given by

$$B_n = \int_{0}^{F_s} |H(f)|^2 df \quad (1)$$

where $H(f)$ is the digital filter frequency response with $f$ in Hz.

**Problem Statement**

Plot the noise equivalent bandwidth in Hz of a first-order digital lowpass filter versus the filter parameter $a$. The filter frequency response magnitude is given by

$$|H(f)| = \frac{1 - a}{\sqrt{1 - 2a \cos\left(\frac{2\pi f}{F_s}\right) + a^2}} \quad (2)$$

In plotting $B_n$ consider the filter parameter just on the interval $0.4 \leq a \leq 0.9$ and assume a sampling rate of $F_s = 44.1$ KHz (CD audio sampling rate). Perform the integration numerically using the MATLAB function `quad8`. 
Implementation Suggestions

• Write a MATLAB function for the integrand that takes \( f, a \) and perhaps \( F_s \) as inputs, similar to the example in the notes (top of page 7-14), e.g.,

\[
\text{function } y = \text{lowpass_inte}(f,a,Fs)
\]

\[
\ldots
\]

• Write a second MATLAB function that returns \( B_n \) given \( a \) and \( F_s \) as inputs (similar to the call to \texttt{quad} in the middle of page 7-14)

\[
\text{function } Bn = \text{noise_eq}(a,Fs)
\]

\[
\ldots
\%
\text{quad8()} \text{ embedded in here somewhere}
\]

\[
\ldots
\]

• Place the second function in a \texttt{for} loop that steps over the desired range of \( a \) values using a step size appropriate for plotting

– In this \texttt{for} loop store the \( a \) values in a vector and the \( B_n \) values in a vector

\[
a = \text{zeros(1,100)}; \ Bn = \text{zeros(1,100)};
\]

\[
\text{for } k=1:100
\]

\[
\ldots
\]

\[
a(k) = \ldots
\]

\[
Bn(k) = \text{noise_eq}(a(k),Fs);
\]

\[
\ldots
\]

\[
\text{end}
\]

• When the loop is complete plot \( B_n \) versus \( a \) and label the plot accordingly
For Your Spare Time?

- A digital bandpass filter was my first thought, but is more complex
- A second-order digital bandpass filter is of the form

\[
H(f) = \frac{(1 - r) \sqrt{1 + r^2 - 2r \cos \left(2 \left(\frac{2\pi f_o}{F_s}\right)\right)}}{1 - 2r \cos \left(\frac{2\pi f_o}{F_s}\right)e^{-j\left(\frac{2\pi f}{F_s}\right)} + r^2 e^{-j2\left(\frac{2\pi f}{F_s}\right)}}
\]

where \(0 < f < F_s/2\) is the filter center frequency and \(0 < r < 1\) controls the filter bandwidth (recall \(j = \sqrt{-1}\))