Effects of finite word length representation of the filter coefficients

Lab, SDP

Objective

Students should observe the effects of having fixed point coefficients in a digital filter, and be able to mitigate the effects.

Theoretical notions

Exercises

1. Convert in binary fixed point format (signed, 6 integer bits, 6 fractionary bits - 1S6Î6F the following numbers:

   a. 273
   b. 273.21875

2. Convert in binary fixed point format (signed, 6 integer bits, 6 fractionary bits - 1S6Î6F the following negative numbers. Negative numbers shall be represented in sign-value, 1's complement (C1) and 2's complement (C2) formats.

   a. -273
   b. -273.21875

3. Quantize the samples $x_1 = 0.42625$ and $x_2 = -0.4333$ the fixed point format 1S0Î4F via:

   a. Truncation
   b. Rounding
   c. Truncation in absolute value

   The negative values shall be represented in C2 format.

4. Using the Octave software, use the `cheby1()` function to design one of the following Chebyshev type I filters:

   a. A low-pass IIR filter of order 7, with cutoff frequency of 6kHz at a sampling frequency of 8kHz;
   b. A high-pass IIR filter of order 7, with cutoff frequency of 2.5kHz at a sampling frequency of 8kHz;
   c. A band-pass IIR filter of order 7, with passband between 0.5kHz and 5.5kHz at a sampling frequency of 8kHz;
   d. A stop-band IIR filter of order 7, with stop band between 1kHz and 3kHz, at a sampling frequency of 8kHz.

5. Use the quantization function `cuant()` provided in the lab data files to quantize the coefficients of the filter, and display the transfer function with `freqz()` in three scenarios:

   - coefficients not quantized (maximum precision)
   - coefficients quantized with 15 fractionary bits
   - coefficients quantized with 6 fractionary bits.

   Use rounding as the quantization method.

   Display all transfer functions on the same figure, to better evaluate the changes.

6. Display the poles and zeros of in each of the three cases above, using `zplane()`

7. Evaluate the effect of quantization considering the **parallel implementation** of the filter:

   - Compute the coefficients for the parallel implementation using the function `rpfd()` provided in the lab files
   - Use the provided function `qfr()` to compute the frequency response with quantized coefficients, in parallel implementation. Read inside the function `qfr()` to see how the input and output arguments.
   - Plot the transfer functions in three cases:
     - coefficients not quantized (maximum precision)
     - coefficients quantized with 15 fractionary bits
     - coefficients quantized with 6 fractionary bits.
   - Also display the pole-zero plot in each case, using `zplane()`

   Note: The function `rpfd()` is used as `[c,nsec,dsec] = rpfd(b,a)` and provides the parallel implementation coefficients as follows:

   - `c` = the coefficients of the quotient polynomial
   - `nsec` = the numerator coefficients of all the subsections, in each line
   - `dsec` = the denominator coefficients of all the subsections, in each line

8. Similar to the above exercise, evaluate the effect of quantization considering the **series implementation** of the filter, which is obtained with the function `tf2sos()`.

9. Which is the implementation which is most robust to quantization?

Final questions

1. TBD