State-space implementations of digital IIR filters

Lab 7, SDP

Objective

The students should become familiar with *state-space* type realization structure used for implementing IIR filters.

Theoretical notions

Exercises

1. Consider the IIR system with the system function

$$H(z) = \frac{1 + 2z^{-1} + 3z^{-2} + 2z^{-3}}{1 + 0.9z^{-1} + 0.8z^{-2} + 0.5z^{-3}}$$

   a. Write the equations and draw the type I and type II state-space implementations of this system
   b. Compute the first 5 values of the step response, considering the initial conditions $v[0] = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

2. Consider the system with the following state-space equations:

$$v[n+1] = \begin{bmatrix} 1 & -0.81 \\ 1 & 0 \end{bmatrix} v[n] + \begin{bmatrix} 1 \\ 0 \end{bmatrix} x[n]$$

$$y[n] = \begin{bmatrix} 1 & -1.81 \end{bmatrix} + x[n]$$

   a. Find the system function of this system

b. Compute the first 5 values of the step response, considering the initial conditions $v[0] = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

c. Draw the the type I and type II state-space implementations of this system, as well as the direct form II implementation

3. Using the Octave software, use the `ellip()` function to design one of the following filters:

   a. A low-pass IIR filter of order 4, with cutoff frequency of 6kHz at a sampling frequency of 8kHz;
   b. A high-pass IIR filter of order 4, with cutoff frequency of 2.5kHz at a sampling frequency of 8kHz;
   c. A band-pass IIR filter of order 4, with passband between 0.5kHz and 5.5kHz at a sampling frequency of 8kHz;
   d. A stop-band IIR filter of order 4, with stop band between 1kHz and 3kHz, at a sampling frequency of 8kHz.

   Name the coefficient vectors `b` and `a`.

4. In Octave, implement a function `filter_stsp(b, a, x)` which filters a signal `x` with the filter defined by the coefficients `b` and `a`. Implementation shall follow the type I state-space equations in matrix form.

5. Use the function above to load and filter the audio signal `Sample.wav`.

   a) Load the file using `audioread()`
   b) Filter the signal using `filter_stsp()`, with the previously designed filter
   c) Play the resulting signal, and display it

Final questions

1. TBD