Designing Oscillators

DSP Lab 13

Table of contents

1 Objective

Students should be able to design basic filters and oscillators in Matlab and implement them in Simulink.

2 Theoretical notions

Oscillators are unstable systems which have at least one pole exactly **on the unit circle**, i.e. the modulus of the pole value is exactly 1.

$$z = 1 \cdot e^{j\omega}$$

In order to have a real-valued system (real-valued impulse response), if the pole is outside of the real axis, there must also be a complex conjugate pole, i.e. there will be a pair of complex conjugate poles.

The phase of the poles, i.e. the angle ω of the poles in the graphical representation, is the pulsation $\omega = 2\pi f$ of the oscillator. For example, if we have a pair of poles $e^{\pm j\frac{\pi}{2}}$ on the

unit circle, situated at an angle of $\omega = \frac{\pi}{2}$, the oscillator will produce a sinusoidal signal $y[n] = \cos(\frac{\pi}{2}n)$.

3 Exercises

1. Use the Filter Design tool in Matlab (`fdatool`) to design a IIR high-pass filter with order 3, with cutoff frequency 0.07.

    1. Implement the filter in Simulink
    2. Apply at the input the signal $x[n] = \cos(2\pi 0.03n)$ (use a `Sine Wave` block) and visualize the output $y[n]$. Is this signal rejected or not by the filter?
    3. Change the input frequency from 0.03 to 0.2 and visualize the output. Is this signal rejected by the filter?

2. Use the Filter Design tool in Matlab (`fdatool`) to design an oscillator with frequency 0.05. Implement it in Simulink, visualize & play the output signal.

    Use the following steps to design the oscillator:

    1. Design a system of order 2 with 2 conjugate poles placed **on the unit circle** at the correct frequency, and 2 zeros at low & high frequencies
    2. Export the filter coefficients in the Matlab workspace
    3. Implement the system in Simulink, **omitting the input signal** (not necessary)
    4. Set a non-zero initial condition in the system, to start-up the oscillator

3. Implement in code the same operations like in the Simulink model, using the following procedure:

    1. Prepare a long vector full of zeros
    2. Set a non-zero value on one (or both) of the first two values
    3. Use a loop to compute every value of the vector based on the preceding two values, starting from the third.
    4. Plot the resulting signal

4 Final questions

1. Why do we need a non-zero initial condition in Simulink? What happens if we don't set it?
2. What happens if we have double poles on the unit circle, instead of single poles?
3. Do the position of the zeros influence the behavior or the implementation of the oscillator?