Exercise 3
Filters

Aalto University
School of Electrical
Engineering

ELEC-E5410 Signal processing for communications

General Guidelines

- min 50% points of the exercises required in the scale 0-100%
- Groups of 1-2 persons
  - Sign up in MyCourses
- If several groups return the same code without declaring cooperation (or similar code, changing names of the variables etc. doesn't count), the max. number of points/group becomes 100%/ #similar submissions
- Approved Exercise 1 required to be able to get the RTL-SDR
  - You can still use your own hardware if you like

General Guidelines

- Download Matlab from download.aalto.fi
- Install RTL-SDR hardware support package from Add-Ons in the Matlab menu bar
- Return .m files
- Return the .pdf made by Matlab's publish() with the results
    - Not .html + several .pngs or some other awkward format

Known issues in installation

- Admin rights needed to install driver
- On Windows, installation of the toolbox changes the rights of Matlab to admin and the path to toolbox is hidden from regular users
- (Always use the same USB port for RTL-SDR on Windows)
- Kernel drive active on Linux
  - sudo echo "blacklist dvb_usb_rtl28xxu" >/etc/modprobe.d/rtlsdr.conf and reboot
- Mac requires Xcode, which takes 7GB of disk space
  - Install command line tools (130MB) only: osxdaily.com/2014/02/12/install-command-line-tools-mac-os-x/

Matlab tips

- **Vectorize**
  - If x is a row vector and y is a column vector, do x*y instead of sum=0; for ii = 1:N, sum=sum+ x(ii)*y(ii); end
  - x(:) makes x a row vector, x(:) .’ makes x a column vector
- **Watch vector dimensions**
  - If x is a column vector and y is a row vector x*y is a NxN matrix instead of a scalar

Matlab tips

- **All functions can be included in one file as local functions**
- **These local functions are not visible outside the file**
- **The functions can be made visible outside by defining anonymous functions in the same file**
    - May be useful in debugging

```
  p_fun = @(x) fun(x);
function y = fun(x)

  ..

  end
```

- **The parameters inside the anonymous function freeze their values when the function is defined**
    - y=1; foo = @(x) x+y; y = 2; foo(1) = 2

Tasks

Tasks

1. **Download the Matlab script template rtl_filt.m from Aalto Git**
   - Functions/scripts starting by "rw_" are the ones you have to write yourself
2. **Tune to an FM station, e.g. Yle Puhe 103.7 MHz. Compensate the dongle's frequency offset in the function call (ppm value) as well as possible since the filters are narrowband and require exact location of 19 KHz carrier**
3. **Read carefully the paper harris, Egg: "Forming narrowband filters at a fixed sample rate with polyphase down and upsampling filters"**
   - Check that the lengths of the filters you design are on the same ballpark as harris' filters

Tasks

4. **Implement the baseline FIR filter with the specifications as in Fig. 4 in harris&Egg paper**
    - Use 60 dB stop band attenuation
    - Note that 400 KHz sampling rate is not possible with RTL-SDR such that down-sampling factor is different from 20
5. **Implement interpolated FIR filter as explained in Section 4 in harris&Egg& paper**
    - The prototype of the periodic filter has passband edge Lω_p and stopband edge Lω_c where $\omega_p$and ω_c refer to passband and stop band edges of the filter in 4., and L is the sparsity of the filters' impulse response (here also the down-sampling factor)

Tasks

6. **Design the polyphase decimator, low-pass filter, and polyphase interpolator as in Fig. 7 in harris&Egg paper**
   - Squaring of the signal need not be applied, because we do not use 38 KHz pilot for anything
   - Low-pass filter has 1 KHz passband and 2 KHz transition band

Polyphase decimator

- **Design a low-pass filter *h* with 1 KHz passband and 18 KHz transition band**
- **Reshape filter coefficients to matrix *H* (**reshape()**). Rows of the matrix *H* are now polyphase components $E_k(z)$.**
  - The dimensions of ***h*** and ***H*** must match, so add zeros to the end of ***h*** if necessary
- **Filter delayed and down-sampled signals with the polyphase components. Each polyphase component filters different samples**
  - For i=1:M, filter(H(i,:),1, x(i:M:end)); end
- **Modulate the outputs of the polyphase filters and sum the outputs together**
- **This procedure corresponds to the commutator, see Fig 7. in harris&Egg**

Polyphase interpolator

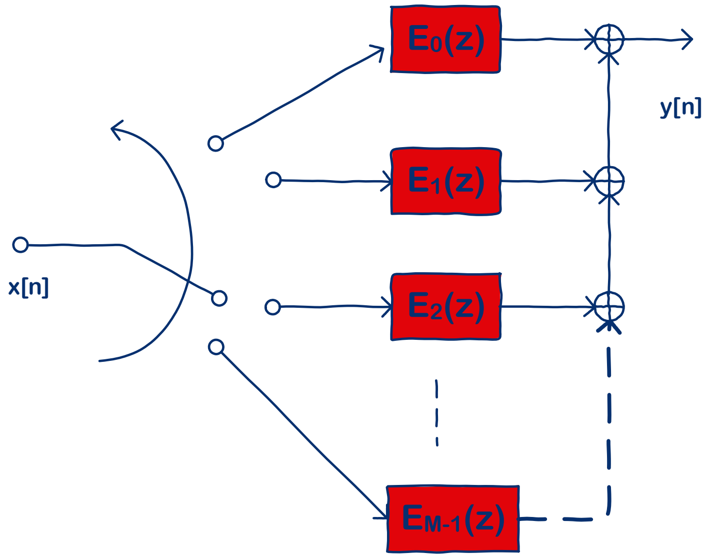
- **Use the same polyphase matrix** *H*
- **Filter the samples by polyphase components, up-sample**
    - For i=1:M, upsample(filter(H(i,:),1,x), M,i-1); end
- **The last argument in** upsample() **delays the outputs such that at one time instant only one of the branches has non-zero output**
- **Modulate the outputs and sum the signals together**
- **This procedure corresponds to the commutator, see Fig. 7**

Report

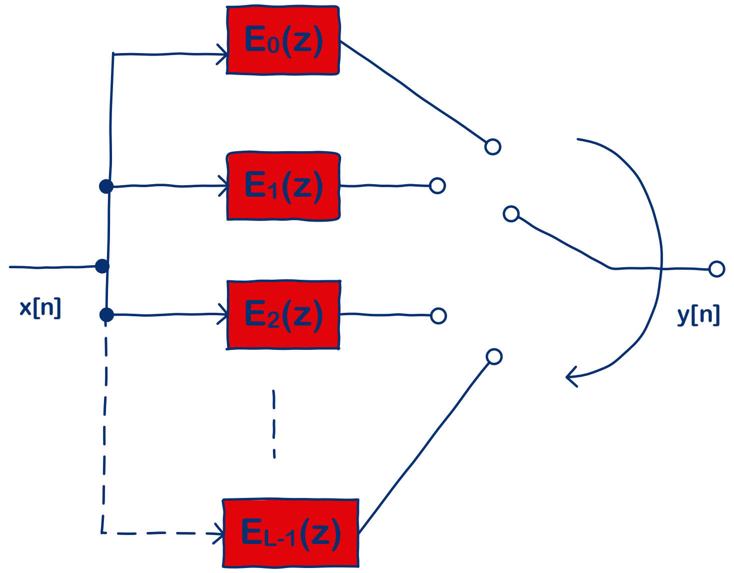
- **Return your Matlab code (.m) and a pdf made by Matlab's publish function The document must contain the algorithms and the parameters**
- **In addition, compare the complexities of the three filters**
  - How many multiplications per second the filters have
- **In addition, determine, how many stages in cascaded-integrator comb filter are needed to achieve 60 dB attenuation in stop-band**
  - The length of the filter in one stage in CIC is the same as the down-sampling factor. Harris uses down-sampling from 400 to 20 KHz, which we cannot use due to the RTL-SDR dongle
  - Design one-stage CIC given the down-sampling factor. Add stages until the desired attenuation is met.

Appendix

Commutator representation of decimation with polyphase structure

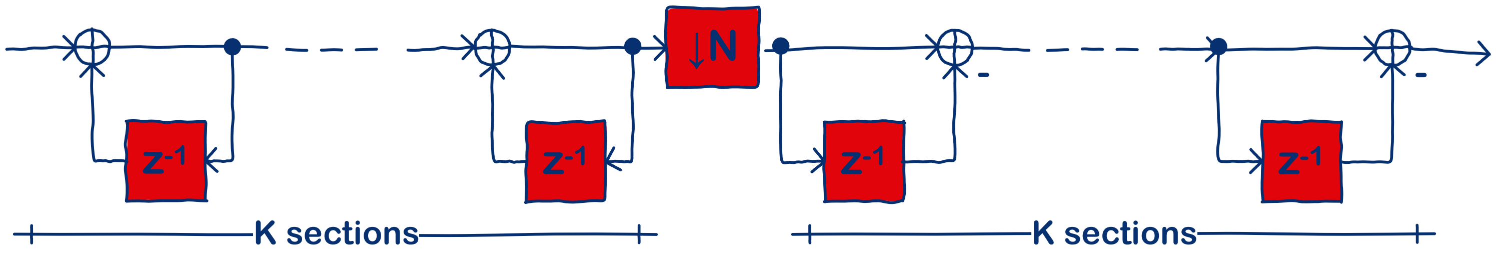


Commutator representation of interpolation with polyphase structure



CIC decimator

- **CIC decimator with K sectors, down-sampling factor N**



z^{-1}
z^{-1}
$\downarrow N$
z^{-1}
z^{-1}
-
-
K sections
K sections