Filter Design for Down/Up sampling

Last lecture we saw Downsampling and Upsampling techniques which concludes-
Down sampling
1.Information is Lost when signal is downsampled
2.down sampler causes aliasing
Up sampling
1.information is NOT lost when signal is up sampled
2.Up sampler produces Spectral images.

Solutions:
Down Sampler
Input Signal is passed through a low pass filter and Band limited to (π/D).
Then it is downsampled at the rate (Fx/D) where Fx=Sampling rate of input signal

Up Sampler
In Up sampler signal is upsampled at the rate Fx*I and then passed through a low pass filter which eliminates the spectral images.(by rejecting all the values above ($\omega=\pi$/I).

Efficiency in Filter design
Upsampler
In a typical Upsampler followed by Low pass filter (LPF) :
LPF works at the rate of : I*Fx
But we know that Upsampler inserts I-1 zeros in-between two samples so lots of filter coefficients will be zero which implies that I-1 multiplication and addition in the filter gives out output zero which is perhaps same as input i.e. upsampled signal). This leads some kind of redundancy.

So we come up with next solution
We can combine the Upsampler and Filter kernel such that input signal is multiplied by filter coefficients and then Upsampled to insert zeros (in effect instead of just bypassing zeros of upsampled data in filter we first pass each input sample through filter and insert (I-1) zeros at the output and combine the result of each sample of input signal)
Thus now Filter operates at input sampling rate Fx.

Thus reduction in Filter frequency is of 1/I.

Similarly it can be designed for downsampler i.e. by combining the filter kernel and downsampler, we first select the Dth sample and multiply it with filter coefficient. So filter works at Fx/D where in a typical case it would be operating at Fx.
Thus reduction in Filter frequency is of 1/D.

USE of symmetric property of Filter kernel

We can still reduce the multiplications in filtering operation by NOT calculating the h(i+M/2)*Xj which will be same as h(i)*Xj (M is size of filter kernel) .Thus we can reduce no. of multiplications by ½.It will also reduce space required for storing these multiplications.

Polyphase filter design for Upsampler

If we observe the Upsampling process ,it introduces I-1 zeros thus if filter neglects(actually holding I-1 samples) first I-1 inputs from upsampler then we will get a output of I samples in next time slot.

Now if M is size of filter kernel (that means it can hold M inputs before giving outputs) then in each of next time slot we will get FLOOR (M/I) no. Of NON-ZERO output samples, which are then multiplied with filter coefficients.
We are interested in finding out these NON-ZERO samples.
Hence we can consider Upsampler as a parallel combination of I filters which starts giving I output samples after first I-1 samples.
Working goes like this:
Assuming M as a multiple of I, in first I samples there are (M/I) non-zero sample output which are multiplied with a downsampled version of h(n)(filter sequence) h(0),h(I),h(2I).. For next input sample these samples will get shifted (delayed) by one and will still have M/I no. of non-zero samples which will get multiplied by h(1),h(I+1),h(2I+1)..
Size of each such filter bank will be (M/I).

If we observe the first filter bank it is
Downsampled original filter kernel h(n) with D=I
And
Each such next filter i is downsampled signal of shifted filter kernel h(n+i)
Where 1<=i<=I-1.
Thus each filter differs in phase hence called as POLYPHASE filters
Thus there are I different filters acting on I samples.
Similar kind of design can be done for Downsampling.

Now I-1 filters work at frequency Fx and Output of each filter is collected at rate I*Fx
Hence reduction in filter calculations we obtain is : (I-1)/I

Rational Sampling rate conversion (p/q)
There are two approaches for doing rational sampling.
DOWN-> UP sampling
Downsampling signal first and then upsampling looses information in signal.
Since downsampling selects every qth sample and upsampler then inserts p zero the output signal do not contain same information.
UP->DOWN sampling
Upsampling signal first (inserting zeros i.e. no information loss) and then downsampling do not have information loss but signal is aliased. But aliasing can be eliminated by anti-aliasing filter.

Hence in Rational sampling rate conversion Upsampling is done before Downsampling. This kind of sampling can be obtained by cascading two polyphase filters of an upsampler and downsampler.

Thus combining the 2 filter kernels of we can get the desired result.

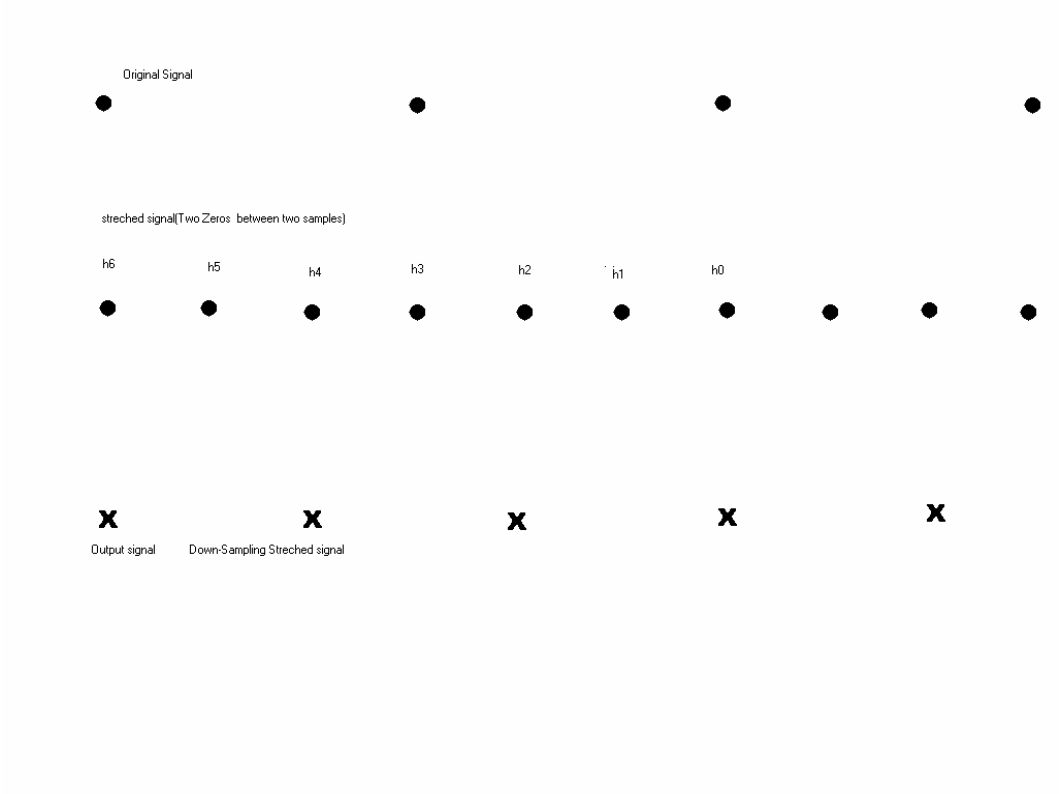
TIME -VARIANT Filters.
If p/q is the ratio we wanted …then for q inputs we want p outputs…

Getting polyphase filters from filter kernel we just downsample it at rate D=I
For example let us assume that we want to sample the input at rate 3/2

So when upsampler is followed by downsampler..
In upsampler we have filter banks of (h0,h3,h6..) (h1,h4,h7) (h2,h5,h8..) …..
So first non-zero output we get is from 1st bank….Second from 2nd bank and so on

When we down sample these stretched signal from upsampler we are interested in non-zero values of stretched signal only.
And since we want every second sample to be selected we design the filter using above filter banks …
So combininng effect will be arranging filter banks with gap of 2 as
(h0, h3,h6..) then (h2,h5,h8..)….. (h4,h7,h10..) and so on
But we need to take care of the previous coefficient of the filter also..
Assuming h(n) is causal i.e. h(n)=0 for n<0
We get the banks as
(0,h0, h3, h6..) then (0,h2, h5,h8…)then(h1,h4,h7…) and so on

Following diagram will best show this example

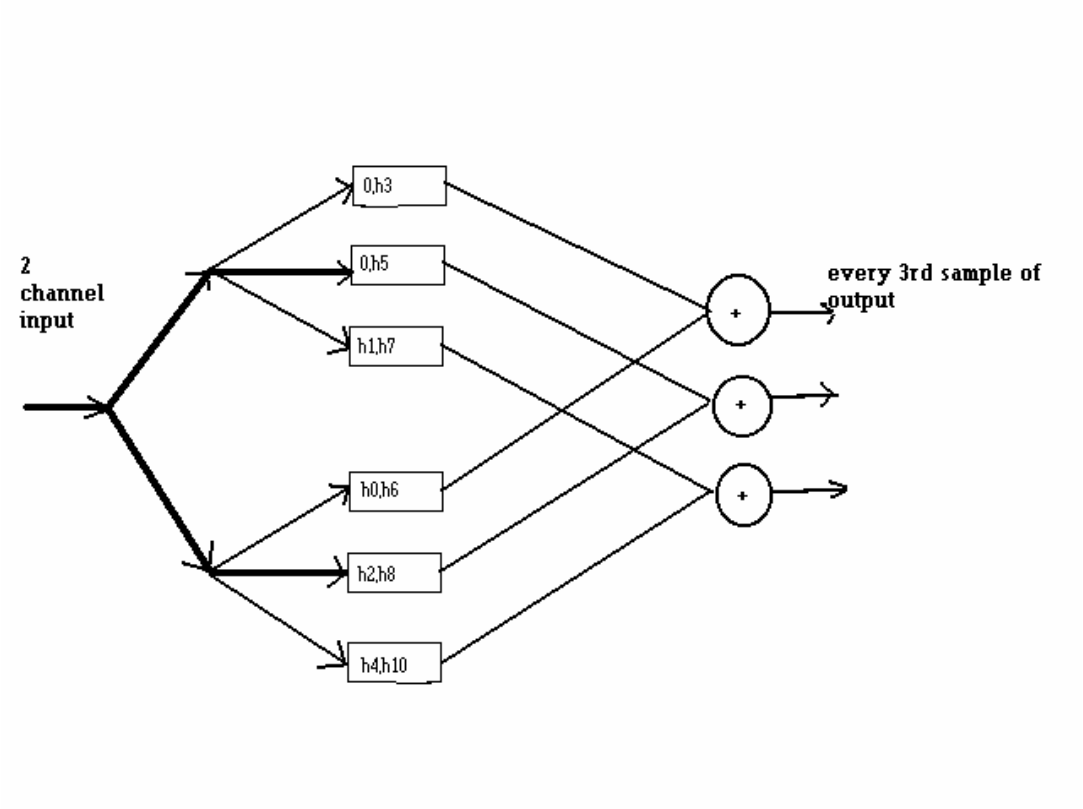


If u observe the diagram
first output sample(4th from left) can be obtained by using filter coefficients h(-3)=0,h0,h3,h6

Next one (3rd from left) can be obtained by h (-1)=0,h2, h5, h8
Similarly next one (2nd from left) can be obtained from h1, h4, h7,h10
Same sequence will get repeated after each 6 samples of stretched signal

Reduction we get by this is 1/(D*I).

Implementation of 3/2 sampler for 2 channel input can be shown as



Real-value upsampling
Suppose we want to upsample signal by 23.7.
In this case we can find p and q such that p/q best approximate to 23.7
But p and q may be too huge to design upsampler OR downsampler.
Hence method is not always useful.
Hence we first upsample signal by 23 and use approximation.
We can use linear filter to find the neighbor sample, which is close to 0.7 and use the "error" in approximation to predict next sample to approximate.

Applications of multirate sampling
1.speech scaling and speech shifters
2.compression/decompression algorithms
3.graphics