Filters_and_Noble_Identities

March 17, 2017

0.1 Example:

Take a simple filter, in Matlab or Octave notaton: B=[1,1]; (a running average flter), an input signal x=[1,2,3,4,...] , in Python:

```
In [1]: import numpy as np
        x =np.arange(1,10, dtype='float') #lfilter needs float
        N=2
```

Now we would like to implement the **first block diagram** of the Noble Identites, the down-sampling (the pair on the frst line, with outputs y1 and y2). First, for y1, **down sampling followed by filtering**, the down-sampling by a factor of N=2:

```
In [2]: xd = x[::N]
        print xd
```

```
[ 1.  3.  5.  7.  9.]
```

Then apply the flter B=[1,1],

```
In [3]: import scipy.signal
        B = [1,1]
        y1 = scipy.signal.lfilter(B, 1, xd)
        print y1
```

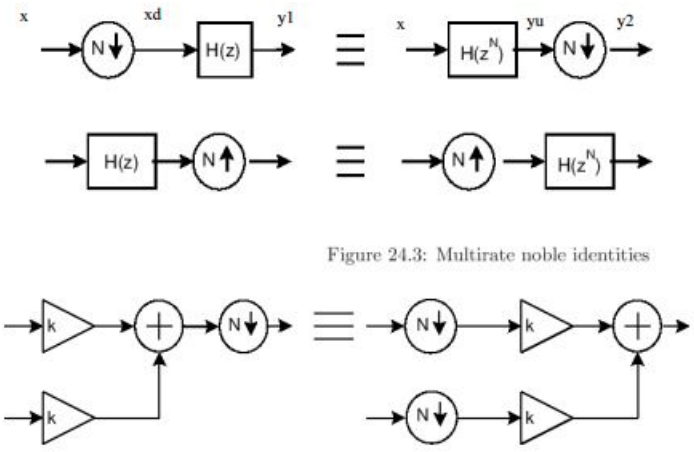

Figure 24.3: Multirate noble identities

Multirate Noble Identities

```
[  1.   4.   8.  12.  16.]
```

This yields the sum of each pair in xd: $y1 = 1, 4, 8, 12, 16$

Now we would like to implement the corresponding righthand side block diagram of the noble identity, **filtering followed by down sampling**. Our filter is now up-sampled by N=2:

```
In [4]: Bu = np.zeros(3)
        Bu[::N] = B
        print Bu
```

```
[ 1.  0.  1.]
```

Now filter the signal before down-sampling:

```
In [5]: yu = scipy.signal.lfilter(Bu, 1, x)
        print yu
```

```
[  1.   2.   4.   6.   8.  10.  12.  14.  16.]
```

Now down-sample it:

```
In [6]: y2 = yu[::N]
        print y2
```

```
[  1.   4.   8.  12.  16.]
```

Here we can now see that they are **indeed identcal**, y1=y2!