Linguistics 582
Basics of Digital Signal Processing

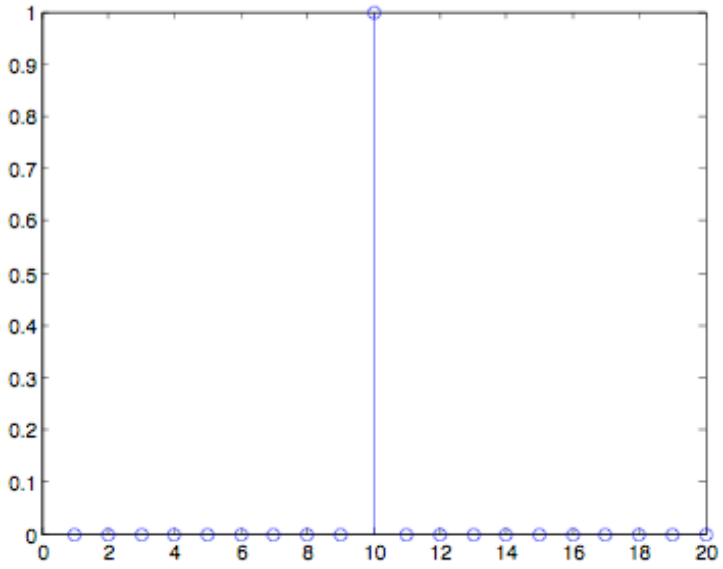
Feedback (IIR) Filters

1. Impulse response

There is yet another way of thinking about the vector, b, of coefficients that defines a moving average filter. Consider what happens when the filter is applied to a signal I(k), which is defined as zero everywhere except at a single point (usually considered k=0), where its value is equal to 1. This signal is referred to an Impulse. The response of a filter to an impulse signal is referred to as the *impulse response* of the filter. In the case of a moving average filter, the impulse response will be the sequence of filter coefficients themselves (why?) This gives us, then, a new interpretation of the filter coefficient vector—it is, in fact, the time signal produced by the filter in response to an impulse.

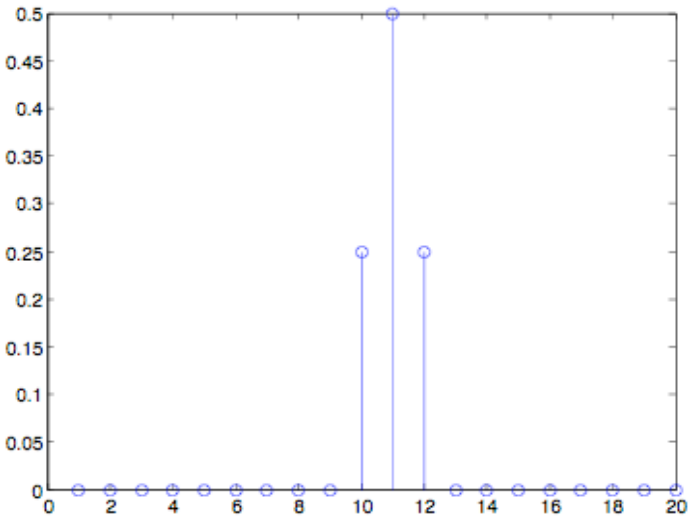
For example, the signal I(k), in (1) below, is an impulse. Instead of setting I(k) to have a value of 1 at k=0, and a value of 0 at positive and negative values of k, I have set it equal to 1 at k=10, and have set it equal to 0 from k=1 to k=9 and from k=11 to k=20. This is because vector indices in MATLAB cannot be negative, so I(-3), for example, is undefined, and cannot be set equal to 0. In order for the impulse response to be correctly produced, there must be zero sample values both before and after the non-zero value. Note that `zeros(m, n)` is a function creates a matrix with m rows and n columns and fills it with zeros.

```
»I = [zeros(1,9) 1 zeros(1,10)];
»stem (I)
```



Create the coefficients of a digital filter (b), and filter the impulse (I) through it:

```
» b = [.25 .5 .25]
b =
    0.2500    0.5000    0.2500
»Y = filter_N (b,I)
Y =
  Columns 1 through 7
         0         0         0         0         0         0         0
  Columns 8 through 14
         0         0    0.2500    0.5000    0.2500         0         0
  Columns 15 through 20
         0         0         0         0         0         0
»stem(Y)
```



Note that impulse response is equal to the filter coefficients themselves. You should confirm for yourselves, given the definition of such a filter, that this is expected.

Since the impulse response of a feedforward filter is just the coefficients themselves, the number of non-zero points in the response will be equal to the number of coefficients. Thus, the length of the impulse response will always be finite. Therefore, filters of this type (moving average filters) are also known as Finite Impulse Response filters, or FIR filters.

2. Finite Impulse Response Filters (FIR)

The output of an FIR filter is just the sum of weighted and delayed versions of the input sequence. The output will "die" when the input is exhausted (plus a few extra samples determined by the largest delay term).

An FIR filter (of order 3) is shown schematically on the top of the next page. The boxes marked with "D" indicate delays of one sample, and the circles marked with "*" indicate multiplications.

Three salient properties of a filter of this kind:

- output is a finite sequence if the input is a finite sequence
- there are no feedback pathways
- the transfer function H(z) can be represented as polynomial in negative powers of z

2. Feedback Filters (IIR)

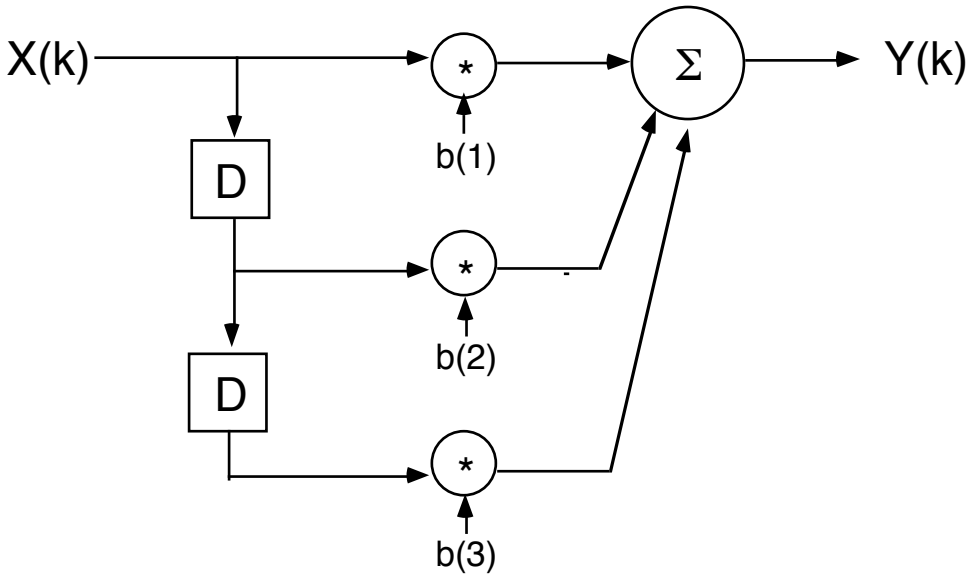
These filters involve feedback: the value of the output sample value at a given point, Y(k), is dependent on previous values of the output, as well as the present (or past) value of the input. As a result, the impulse response is theoretically (but not practically) infinite--it goes on forever, although in practice it usually reaches a condition in which its changes are smaller than the bit resolution of the system.

An IIR filter (of order 2) is shown schematically on the bottom of the next page. For now, don't worry about the "-" signs: the reason for this will be discussed below. Just compare the topologies of the two networks.

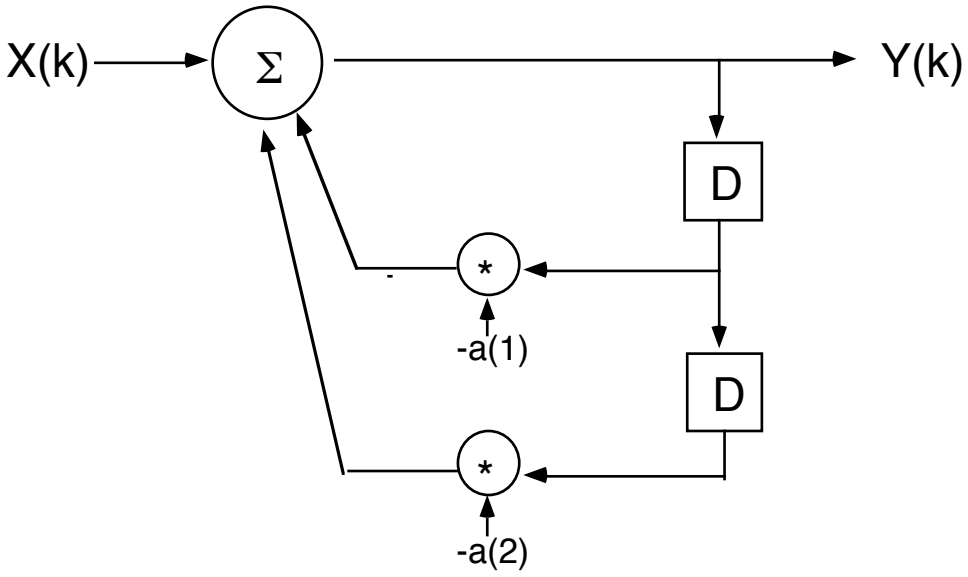
Three important properties of a filter of this kind:

- output is an infinite sequence even if the input is a finite sequence
- there are feedback pathways from the output to input
- the transfer function H(z) can be represented as a ratio of two polynomials in negative powers of z.

Finite Impulse Response Filter (FIR)



Infinite Impulse Response Filter (IIR)



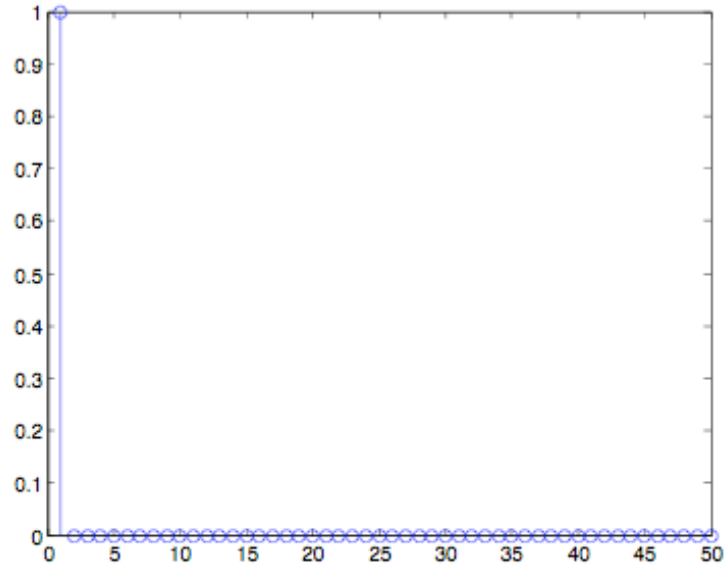
3. Simple Feedback (IIR) filters

A simple example of an IIR filter is shown in (1):

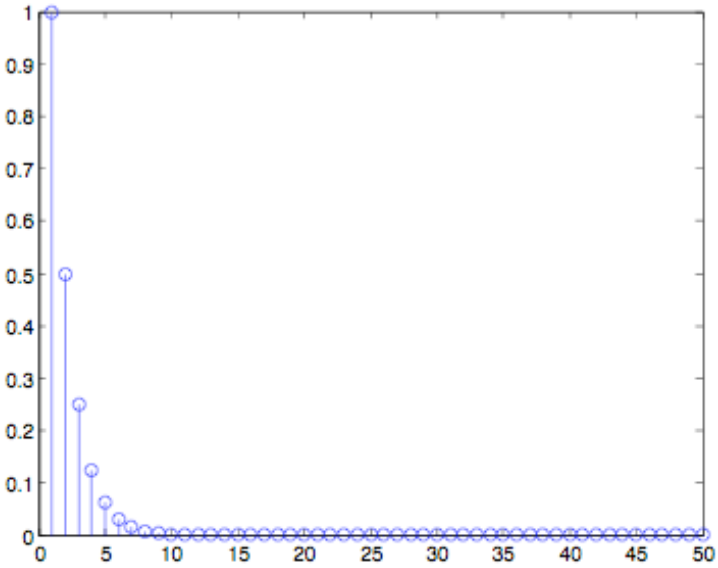
(1) $Y(k) = X(k) + .5\ Y(k-1)$

The output is equal to the input plus one-half of the previous output point.
Watch what happens when this equation is applied to an impulse (a sequence with a value of 1 at X(1), and a value of zero elsewhere , the rest of the 50 samples):

Impulse



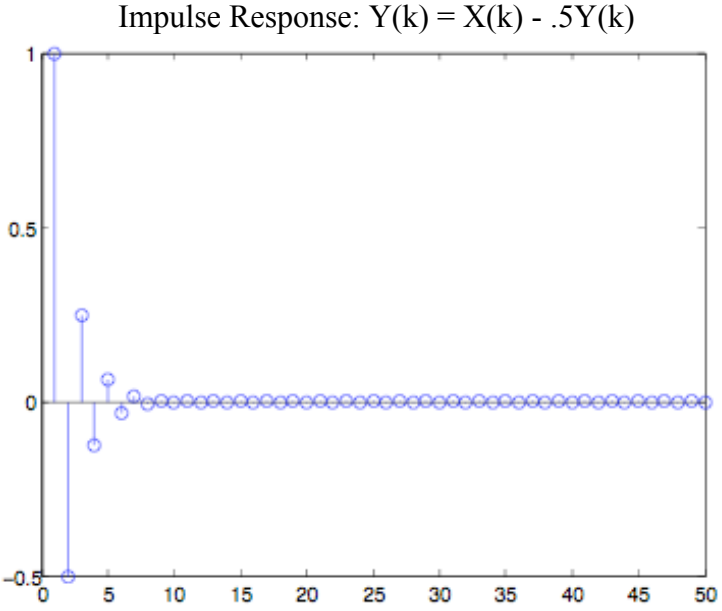
Impulse Response: $Y(k) = X(k) + .5Y(k)$



Note that the system approaches zero asymmtotically.

Now look at the behavior of a similar filter in (2):

(2) $Y(k) = X(k) - .5\ Y(k-1)$



Note that because of the negative sign, the system oscillates as it moves toward its asymtote.

4. Transfer function of a simple IIR filter

Let's derive the transfer function H(z) for the filter in (2). Assume that X is a phasor signal. Multiplication by z^{-1} means to delay by one sample, so

(3) $Y = X - .5z^{-1}Y$

Now rearrange terms:

(4) $X = Y + .5z^{-1}Y$

$X = (1 + .5z^{-1})Y$

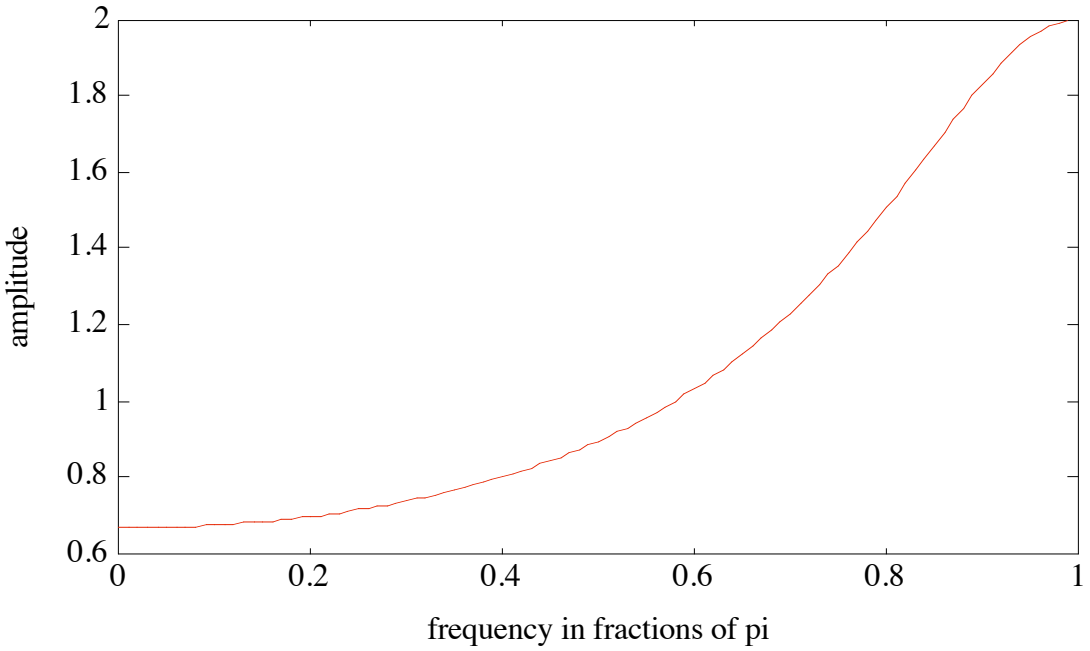
The transfer function H(z) is the ratio of the z-transform of the output to the z-transform of the input:

$$(5) \qquad H(z) = \frac{Y}{X} = \frac{1}{1 + .5z^{-1}}$$

Thus, just like moving average filters, the transfer function of this IIR filter involves a polynomial of terms in negative powers of z. In this case, however, the polynomial is the denominator of a ratio. The numerator equals 1 in this case, because there are no terms in (2) that delay and weight the input sequence. If there were such terms, as we shall see below, there would be another polynomial in z in the numerator, derived exactly as we have before for moving average filters.

Given an expression like (5) for the transfer function, we can evaluate it around the unit circle , just the way we did in our `freqresp` function. The sum of the terms of the polynomial *was* H(z(ω)) itself in `freqresp`. Here we have to take that sum of terms of the polynomial and then divide 1 by that sum to get H(z).

If we do that, the amplitude response of this filter is:

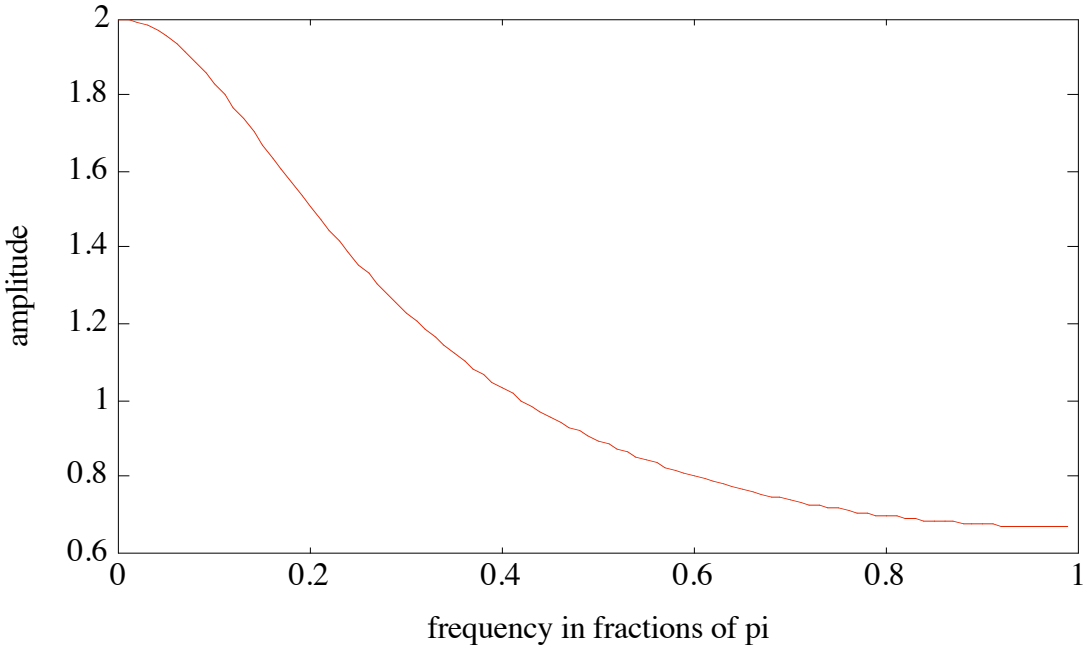


Is this a low-pass or a high pass filter? Is that consistent with impulse response of this filter that we saw above?

The transfer function for the filter in (1) will be:

(6) $$H(z) = \frac{Y}{X} = \frac{1}{1 - .5z^{-1}}$$

Again, it can be evaluated around the unit circle. Its amplitude response is shown below:



Is this a low pass or high pass filter? Is this consistent with its impulse response?

5. General expression for recursive filters

Recall that the general expression that we derived for a moving average filter was as in (7):

(7) $$Y(k) = b(1)\cdot X(k) + b(2)\cdot X(k-1) + \ldots + b(M)\cdot X(k-M+1)$$

For a general recursive filter, including terms that can depend on both past input and past output, the expression is in (8);

(8) $$\begin{aligned} Y(k) = {} & b(1)\cdot X(k) + b(2)\cdot X(k-1) + \ldots + b(M)\cdot X(k-M+1) \\ & -a(1)\cdot Y(k-1) - a(2)\cdot Y(k-2) - \ldots - a(L)\cdot Y(k-L) \end{aligned}$$

By convention, the terms that depend on previous output are subtracted, rather than added (if you want the contribution of the term to be positive, you can, of course, chose a negative value for a).

The `filter` function in MATLAB can be used to filter an input signal through a feedforward filter, a feedback filter, or both. It works like the filter_N program that we wrote, extended to feedback filters. The function works like this:

```
Y = filter(b,a,X)
Y - filtered signal
b - vector of feedforward coefficients, numerator of transfer function (9)
a - vector of feedback coefficients, denominator of transfer function (9)
X - input signal
```

To filter with feedforward only, set a=1. So `filter ([.5  0.5], 1 ,X)` would give the same result as our `filter_N([.5 0 .5], X)`. To use the filter function in recursive mode, we can set the elements of the `a` vector to correspond the coefficients of (8). There is just one complication here. The first element of the vector of `a` coefficients must be 1 (This is the coefficient of the undelayed output on the output. If you think about it, it has to be 1). So a(1) in equation (8) will be the second element of the `a` vector, a(2) the third element, etc.

The examples of impulse response shown above were obtained using MATLAB. To do the filtering according to eq (1), repeated here below, I set `a` and `b` as follows:

(1) $Y(k) = X(k) + .5\ Y(k-1)$

```
a = [1  -.5];
b = 1;
```

Note the extra 1 that is required in the `a` vector. Also, since the convention is that output terms are subtracted, I had to set the value of a(1) (the second element in MATLAB vector `a`) to -.5.

Likewise, for equation (2):

(2) $Y(k) = X(k) - .5\ Y(k-1)$

```
a = [1  .5];
b = 1;
```

6. General Expression for H(z) for a recursive filter.

Using simple algebra like we employed in (3)-(5) above we can derive the following expression for the transfer function of a recursive filter:

$$(9)\ H(z) = \frac{Y}{X} = \frac{b(1) + b(2)\,z^{-1} + \ldots + b(M)\,z^{-(M-1)}}{1 + a(1)\,z^{-1} + \ldots + a(L)\,z^{-L}}$$

Note that the "extra" 1 that is in the `a` vector shows up in the denominator here.