2.161 Signal Processing: Continuous and Discrete
Fall 2008

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
DEPARTMENT OF MECHANICAL ENGINEERING
2.161 Signal Processing – Continuous and Discrete

Frequency-Sampling FIR Filters [1]

1 Introduction

In the frequency-sampling filters the parameters that characterize the the filter are the values of the desired frequency response $H(e^{j\omega})$ at a discrete set of equally spaced sampling frequencies. In particular, let

$$\omega_k = \frac{2\pi}{N}k \qquad k = 0, \ldots, N-1 \tag{1}$$

as shown in Fig. 1 for the cases of N even, and N odd. Note that when N is odd there is no sample at the Nyquist frequency, $\omega = \pi$. The frequency-sampling method guarantees that the resulting filter design will meet the given design specification at each of the sample frequencies.

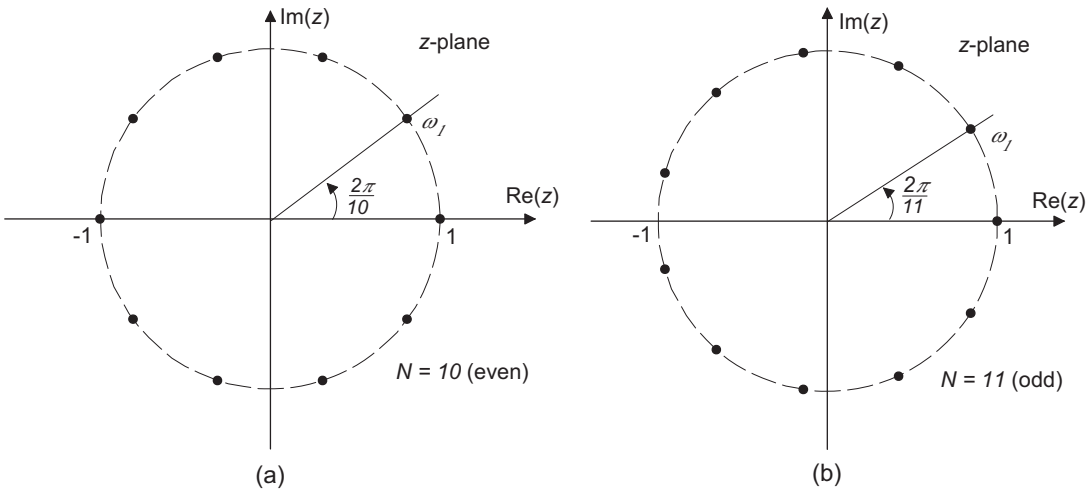


Figure 1: Representative z-plane location of frequency samples for (a) N even, and (b) N odd.

For convenience denote the complete sample set $\{H_k\}$ as

$$H_k = H(e^{j\omega_k}) \qquad k = 1, \ldots, N-1.$$

For a filter with a real impulse response $\{h_n\}$ we require conjugate symmetry, that is

$$H_{N-k} = \bar{H}_k \tag{2}$$

[1] D. Rowell November 10, 2008

and further, for a filter with a real, even impulse response we require $\{H_k\}$ to be real and even, that is

$$H_{N-k} = H_k. \tag{3}$$

Within these constraints, it is sufficient to specify frequency samples for the upper half of the z-plane, that is for

$$\omega_k = \frac{2\pi}{N}k \qquad \begin{cases} k = 0, \ldots, \frac{N-1}{2} & N \text{ odd} \\ k = 0, \ldots, \frac{N}{2} & N \text{ even.} \end{cases} \tag{4}$$

and use Eqs. (2) or (3) to determine the other samples.

If we assume that $H(e^{j\omega})$ may be recovered from the complete sample set $\{H_k\}$ by the cardinal sinc interpolation method, that is

$$H(e^{j\omega}) = \sum_{k=0}^{N-1} H_k \frac{\sin(\omega - 2\pi k/N)}{\omega - 2\pi k/N} \tag{5}$$

then $H(e^{j\omega})$ is completely specified by its sample set, and the impulse response, of length N, may be found directly from the inverse DFT,

$$\{h_n\} = \text{IDFT}\,\{H_k\}$$

where

$$h_n = \frac{1}{N}\sum_{k=0}^{N-1} H_k e^{j\frac{2\pi kn}{N}} \qquad n = 0, \ldots, N-1 \tag{6}$$

As mentioned above, this method guarantees that the resulting FIR filter, represented by $\{h_n\}$, will meet the specification $H(e^{j\omega}) = H_k$ at $\omega = \omega_k = 2k\pi/N$. Between the given sampling frequencies the response $H(e^{j\omega})$ will be described by the interpolation of Eq. (5).

1.1 Linear-Phase Frequency-Sampling Filter

The filter described by Eq. (6) is finite, with length N, but is non-causal. To create a causal filter with a linear phase characteristic we require an impulse response that is real and symmetric about its mid-point. This can be done by shifting the computed impulse response to the right by $(N-1)/2$ samples to form

$$H'(z) = z^{-(N-1)/2} H(z)$$

but this involves a non-integer shift for even N. Instead, it is more convenient to add the appropriate phase taper to the frequency domain samples H_k before taking the IDFT. The non-integer delay then poses no problems:

- Apply a phase shift of

$$\phi_k = -\frac{\pi k(N-1)}{N} \tag{7}$$

to each of the samples in the upper half z-plane

$$H'_k = H_k e^{j\phi_k} \qquad \begin{cases} k = 0, \ldots, (N-1)/2 & (\text{for } n \text{ odd}) \\ k = 0, \ldots, N/2 & (\text{for } n \text{ even}) \end{cases}$$

- Force the lower half plane samples to be complex conjugates using Eq. (2).

$$H'_{N-k} = \bar{H}'_k \qquad \begin{cases} k = 1, \ldots, (N-1)/2 & \text{(for } n \text{ odd)} \\ k = 1, \ldots, N/2 - 1 & \text{(for } n \text{ even)} \end{cases}$$

- Then the linear-phase impulse response is

$$\{h_n\} = \text{IDFT}\,\{H'_k\}$$

1.2 A Simple MATLAB Frequency-Sampling Filter

The Appendix contains a MATLAB script of a tutorial frequency-sampling filter

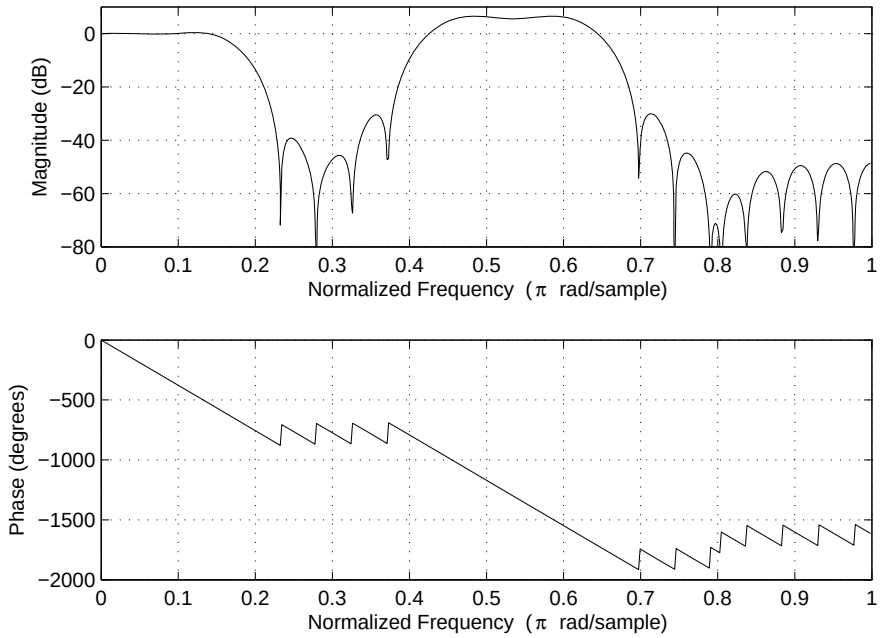
```
h = firfs(samples)
```

that takes a vector `samples` of length N of the desired frequency response, and returns the linear-phase impulse response `h` of length $2N - 1$.

The following MATLAB commands were used to generate a filter with 22 frequency samples, generating a length 43 filter.

```
h=firfs([1 1 1 1 0.4 0 0 0 0 0.8 2 2 2 2 0.8 0 0 0 0 0 0 0 ]);
freqz(h,1)
```

The filter has two pass-bands; a low-pass region with a gain of unity, and a band-pass region with a gain of two. Notice that the band-edges have been specified with transition samples, this is discussed further below. The above commands produced the following frequency response for the filter.



1.3 The Effect of Band-Edge Transition Samples

One of the advantages of the frequency-sampling filter is that the band-edges may be more precisely specified than the window method. For example, low-pass filters might be specified by

```
h = firfs([1 1 1 1 1 0.4 0 0 0 0 0 0]);
```

with one transition value of 0.4, or

```
h = firfs([1 1 1 1 0.7 0.2 0 0 0 0 0 0]);
```

with a pair of transition specifications. The frequency-sampling filter characteristic will pass through these points, and they can have a significant effect on the stop-band characteristics of the filter.

Figure 2 shows the effect of varying the value of a single transition point in a filter of length $N = 33$. The values shown are for $t = 0.6, 0.4$ and 0.2. There is clearly a significant improvement in the stop-band attenuation for for the case $t = 0.4$. Similarly Fig. 3 compares the best of these single transition values ($t = 0.4$) with a the response using two transition points ($t_1 = 0.59$, $t_2 = 0.11$). The filter using two transition points shows a significant improvement in the stop-band over the single point case, at the expense of the transition width.

Rabiner et al. (1970) did an extensive linear-programming optimization study to determine the optimum value of band edge transition values, and tabulated the results for even and odd filters of different lengths. The results show that for one transition point $t_{opt} \approx 0.4$, and for two points $t_{opt} \approx 0.59$, and 0.11.

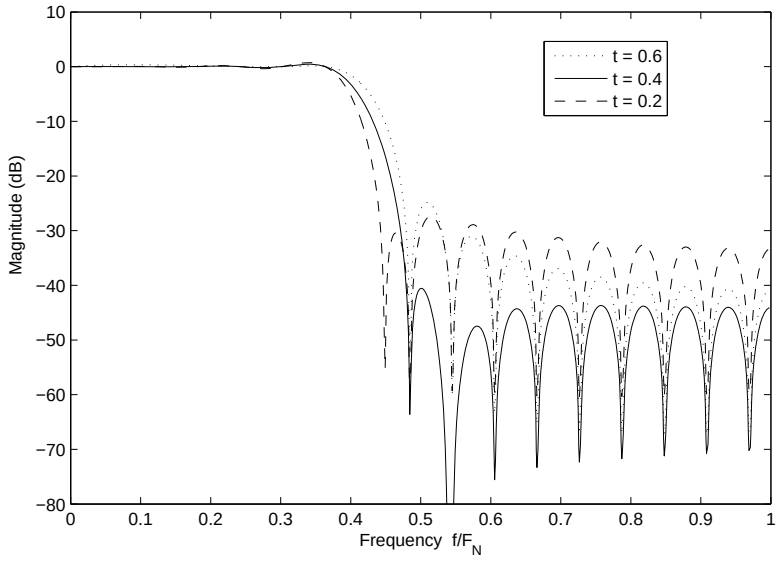


Figure 2: The effect of including a single transition value with value t on the stop-band characteristics of a low-pass (N=33) frequency sampling filter

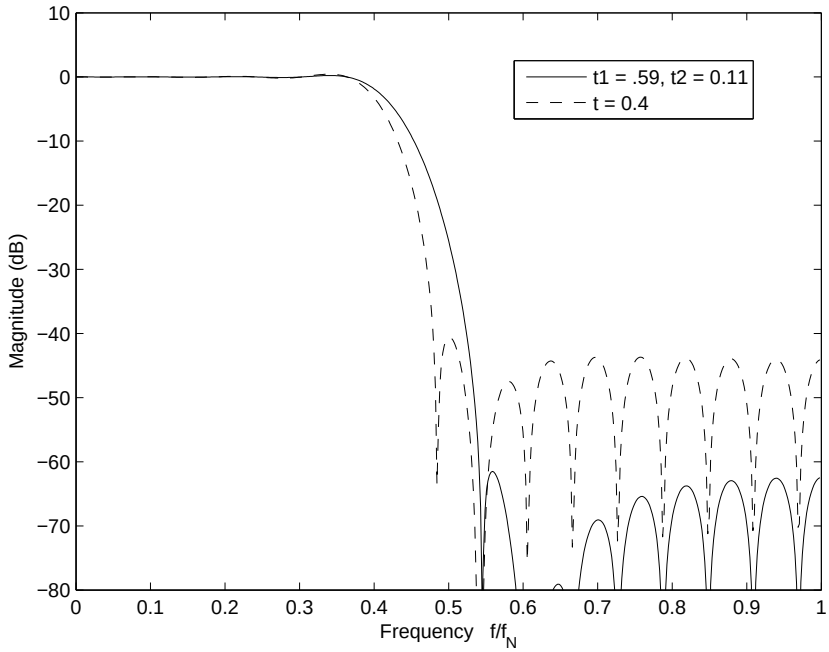


Figure 3: The effect of including a two band-edge transition values t_1 and t_2 on the stop-band characteristics of a low-pass (N=33) frequency sampling filter. In this case the comparison is with the single value $t = 0.4$ frequency response.

2 A Recursive Realization of the Frequency-Sampling Filter

We saw above that the impulse response h_n is the IDFT of the phase-shifted frequency samples or

$$h_n = \frac{1}{N} \sum_{k=0}^{N-1} H'_k e^{j\frac{2\pi n}{N}k}.$$

The z-transform is then

$$\begin{aligned} H(z) &= \sum_{n=0}^{N-1} h_n z^{-n} \\ &= \sum_{n=0}^{N-1} \left(\frac{1}{N} \sum_{k=0}^{N-1} H'_k e^{j\frac{2\pi n}{N}k} \right) z^{-n} \end{aligned} \tag{8}$$

and reversing the summation order

$$H(z) = \frac{1}{N} \sum_{k=0}^{N-1} H'_k \sum_{n=0}^{N-1} \left(e^{j\frac{2\pi k}{N}} z^{-1} \right)^n$$

The z-transform of a finite exponential sequence $x_n = a^n$ for $n = 0, \ldots, N-1$ and 0 elsewhere is

$$X(z) = \frac{1 - (az^{-1})^N}{1 - az^{-1}}$$

so that

$$H(z) = \frac{1}{N}\left(1 - z^{-N}\right) \sum_{k=0}^{N-1} \frac{H'_k}{1 - (e^{j2\pi k/N})z^{-1}} \tag{9}$$

In this form the transfer function is expressed directly in terms of the frequency samples instead of the impulse response.

Equation (9) expresses the filter as a pair of cascaded filters, $H(z) = H_1(z)H_2(z)$ where

$$H_1(z) = \frac{1}{N}\left(1 - z^{-N}\right) \tag{10}$$

is a non-recursive, all-zero filter with N zeros located on the unit-circle, at $z_k = e^{j2\pi k/N}$, $k = 0, \ldots, N-1$. The difference equation for this filter is simply

$$x_n = \frac{1}{N}\left(f_n - f_{n-N}\right)$$

The second filter is a bank of parallel first-order recursive systems

$$H_2(z) = \sum_{k=0}^{N-1} \frac{H'_k}{1 - (e^{j2\pi k/N})z^{-1}} \tag{11}$$

each of which has a pole p_k that coincides with a zero in $H_1(z)$. The difference equation for each of these filters will be

$$y_{k,n} = (e^{j2\pi k/N})y_{k,n-1} + H'_k x_n$$

which involves complex arithmetic. However if we combine two such filters corresponding to complex conjugate pole pairs, and recognize that H'_{N-k} and H'_k are complex conjugates, then a second-order filter with real coefficients results.

$$\begin{aligned} H_k(z) &= \frac{H'_k}{1 - (e^{j2\pi k/N})z^{-1}} + \frac{H'_{N-k}}{1 - (e^{j2\pi (N-k)/N})z^{-1}} \\ &= \frac{A_k - B_k z^{-1}}{1 - 2\cos(2\pi k/N)z^{-1} + z^{-2}} \end{aligned} \tag{12}$$

where

$$\begin{aligned} A_k &= H'_k + H'_{N-k} \\ B_k &= H'_k e^{-j2\pi k/N} + H'_{N-k} e^{j2\pi k/N} \end{aligned}$$

and for the linear-phase filter with $H'_k = H_k e^{-j\pi k(N-1)/N}$

$$A_k = B_k = (-1)^k 2H_k \cos\left(\frac{\pi k}{N}\right). \tag{13}$$

The difference equation for the kth second-order linear-phase filter element is therefore

$$y_{k,n} = 2\cos(2\pi k/N)y_{k,n-1} - y_{k,n-2} + A_k\left(x_n - x_{n-1}\right).$$

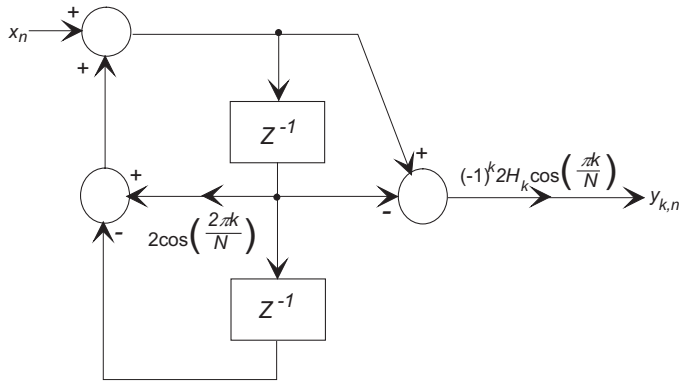


Figure 4: Single second-order filter section in $H_2(z)$ representing complex conjugate frequency samples H'_k and H'_{N-k}.

The structure of a single second-order filter section is shown in Fig. 4.

The complete filter $H_2(z)$ consists of a parallel bank of second-order blocks, supplemented fist-order blocks as necessary:

$$H_2(z) = \frac{H_0}{1-z^{-1}} + \sum_{k=0}^{(N-1)/2} \frac{A_k - B_k z^{-1}}{1 - 2\cos(2\pi k/N)z^{-1} + z^{-2}} \qquad \text{for } N \text{ odd}, \qquad (14)$$

or

$$H_2(z) = \frac{H_0}{1-z^{-1}} + \frac{H_{N/2}}{1+z^{-1}} + \sum_{k=0}^{(N/2)-1} \frac{A_k - B_k z^{-1}}{1 - 2\cos(2\pi k/N)z^{-1} + z^{-2}} \qquad \text{for } N \text{ even}. \qquad (15)$$

The advantage of this structure is in the implementation of narrow band filters where many of the frequency samples H_k are specified as zero. Then many of the second-order blocks will have zero gain and need not be included in the realization, greatly reducing the computational burden.

Example: Show the frequency-sampling realization of a $N = 32$ linear-phase FIR filter with frequency samples:

$$H_k = H(e^{j2\pi k/32}) = \begin{cases} 1 & k = 0, 1, 2 \\ 0.5 & k = 3 \\ 0 & k = 4, 5, \ldots, 15 \end{cases}$$

$H_2(z)$ will contain a single first-order block, corresponding to H'_0, and three second-order blocks corresponding to H'_1. H'_2 and H'_3 and their complex conjugates H'_{31}, H'_{30}, and H'_{29}. From Eq. (13),

$$\begin{aligned} A_1 = B_1 &= -2\cos(\pi/32) \\ A_2 = B_2 &= 2\cos(\pi/16) \\ A_3 = B_3 &= -\cos(3\pi/32) \end{aligned}$$

The complete structure is shown in Fig. 5. As shown there will be a total of 6 multiplications and 14 additions in the computation of each output value, which represents a considerable savings over the convolution with an impulse response length $N = 32$.

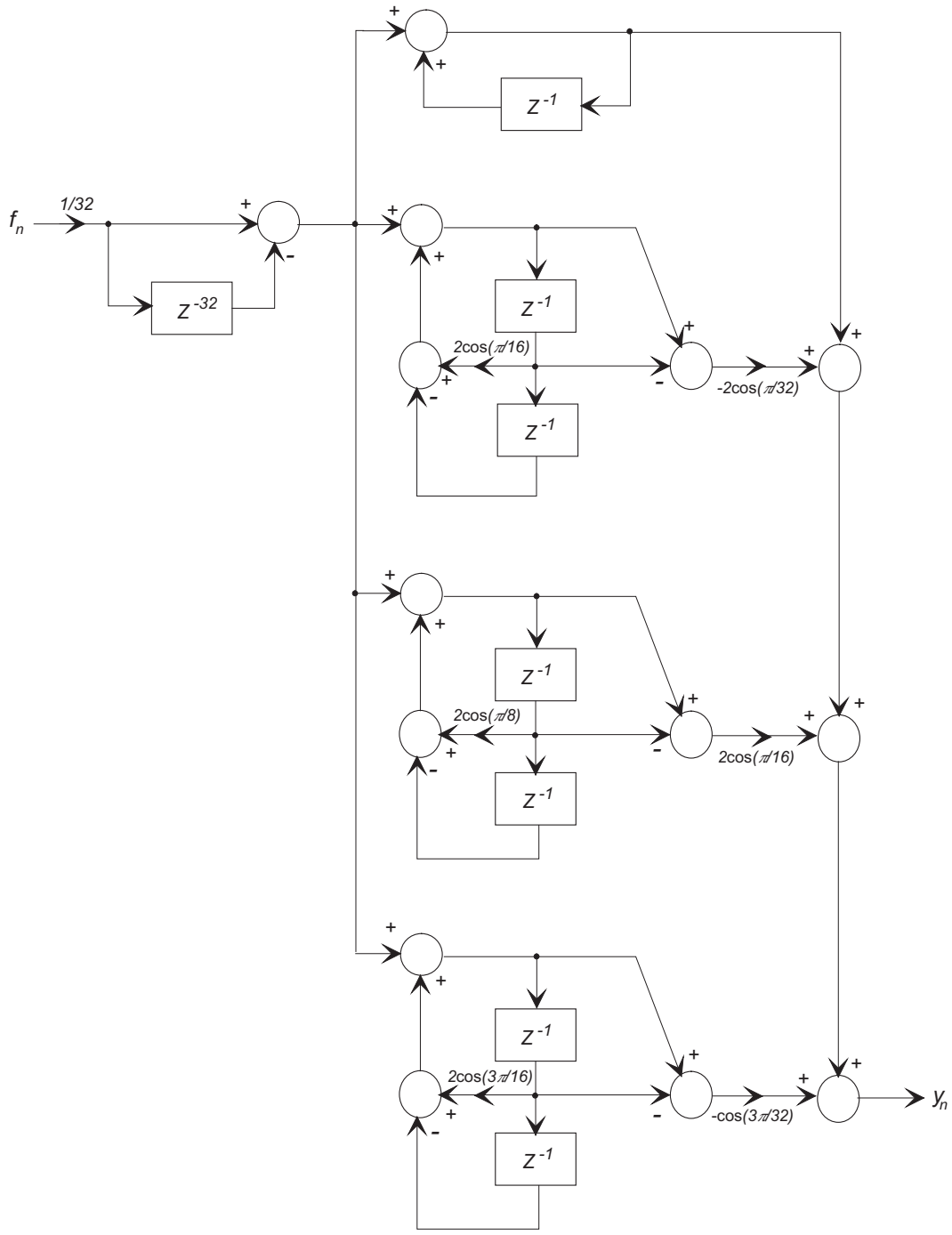


Figure 5: The complete $N = 32$ filter in the example, with three second-order blocks and one first-order block in $H_2(z)$.

Appendix: A Simple MATLAB Linear-Phase FIR Function

```
% ------------------------------------------------------------------------
% 2.161 Classroom Example - firfs - A simple Frequency-Sampling Linear-Phase FIR
%                           Filter based on DFT interpolation.
% Usage :    h = firfs(samples)
%            where samples -  is a row vector of M equi-spaced, real values
%                    of the freq. response magnitude.
%                    The samples are interpreted as being equally spaced around
%                    the top half of the unit circle at normalized (in terms of
%                    the Nyquist frequency f_N) frequencies from
%                           0 to 2(M-1)/(2M-1) x f_N,
%                    or at frequencies 2k/(2N-1)xf_N for k = 0...M-1
%                    Note: Because the length is odd, the frequency response
%                    is not specified at f_N.
%                  h - is the output impulse response of length 2M-1 (odd).
%
%           The filter h is real, and has linear phase, i.e. has symmetric
%           coefficients obeying h(k) = h(2M+1-k), k = 1,2,...,M+1.
%
% Version:  1.0
% Author:   D. Rowell   10/6/07
% ------------------------------------------------------------------------
%
function h = firfs(samples)
%
% Find the length of the input array...
% The complete sample set on the unit circle will be of length (2N-1)
%
N = 2*length(samples) -1;
H_d = zeros(1,N);
%
% We want a causal filter, so the resulting impulse response will be shifted
% (N-1)/2 to the right.
% Move the samples into the upper and lower halves of H_d and add the
% linear phase shift term to each sample.
%
Phi = pi*(N-1)/N;
H_d(1) = samples(1);
for j = 2:N/2-1
   Phase        = exp(-i*(j-1)*Phi);
   H_d(j)       = samples(j)*Phase;
   H_d(N+2-j)   = samples(j)*conj(Phase);
end
%
% Use the inverse DFT to define the impulse response.
%
h = real(ifft(H_d));
```