2.161 Signal Processing: Continuous and Discrete
Fall 2008

Massachusetts Institute of Technology
Department of Mechanical Engineering

2.161 *Signal Processing - Continuous and Discrete*
Fall Term 2008

Lecture 18[1]

Reading:

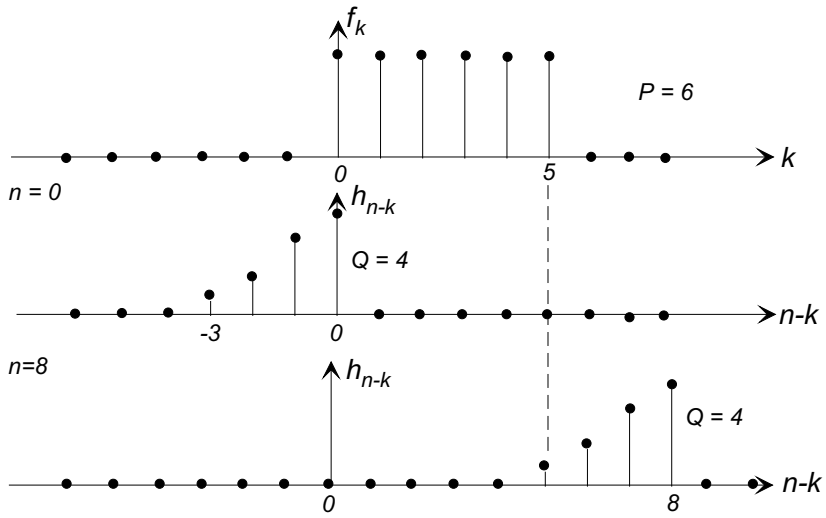
- Proakis and Manolakis: 7.3.1, 7.3.2, 10.3
- Oppenheim, Schafer, and Buck: 8.7.3, 7.1

1 FFT Convolution for FIR Filters

The response of an FIR filter with impulse response $\{h_k\}$ to an input $\{f_k\}$ is given by the *linear* convolution

$$y_n = \sum_{k=-\infty}^{\infty} f_k h_{n-k}.$$

The length of the convolution of two finite sequences of lengths P and Q is $N = P + Q - 1$. The following figure shows a sequence $\{f_n\}$ of length $P = 6$, and a sequence $\{h_n\}$ of length $Q = 4$ reversed and shifted so as to compute the extremes of the convolved sequence y_0 and y_8.

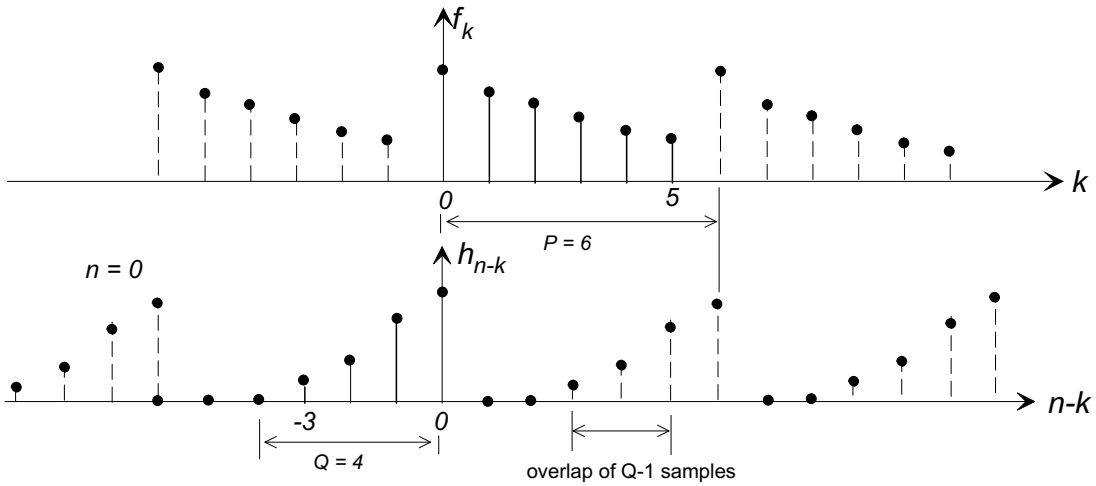


The convolution property of the DFT suggests that the FFT might be used to convolve two equal length sequences

$$y_n = \text{IDFT}\left\{\text{DFT}\left\{f_n\right\}.\text{DFT}\left\{h_n\right\}\right\}.$$

[1]copyright © D.Rowell 2008

However, DFT convolution is a *circular* convolution, involving periodic extensions of the two sequences. The following figure shows the circular convolution of length 6, on two sequences $\{f_n\}$ of length $P = 6$ and $\{h_n\}$ of length $Q = 4$. The periodic extensions cause overlap in the first $Q - 1$ samples, generating "wrap-around" errors in the DFT convolution.



DFT convolution of two sequences of length P and Q ($P \geq Q$) in DFTs of length P

1. Produces an output sequence of length P, whereas linear convolution produces an output sequence of length $P + Q - 1$.
2. Introduces wrap-around error in the first $Q-1$ samples of the output sequence.

The solution is to zero-pad both input sequences to a length $N \geq P+Q-1$ and then to use DFT convolution with the length N sequences.

For example, if $\{f_n\}$ is of length $P = 237$, and $\{h_n\}$ is of length $Q = 125$, for error-free convolution we must perform the DFTs in length $N \geq 237 + 125 - 1 = 461$. If the available FFT routine is radix-2, we should choose N=512.

The use of the FFT for Filtering Long Data Sequences: The DFT convolution method provides an attractive alternative to direct convolution when the length of the data record is very large. The general method is to break the data into manageable sections, then use the FFT to to perform the convolution and then recombine the output sections. Care must be taken, however, to avoid wrap-around errors. There are two basic methods used for convolving long data records. Let the impulse response $\{h_n\}$ have length Q.

Overlap-Save Method: (Also known as the *overlap-discard*, or *select-savings* method.) In this method the data is divided into blocks of length P samples, but with successive blocks overlapping by $Q - 1$. The DFT convolution is done on each block with length P, and wrap-around errors are allowed to contaminate the first $Q - 1$ samples of the output. These initial samples are then discarded, and only the error-free $P - (Q - 1)$ samples are saved in the output record.

With the overlap of the data blocks, in the mth block the samples are

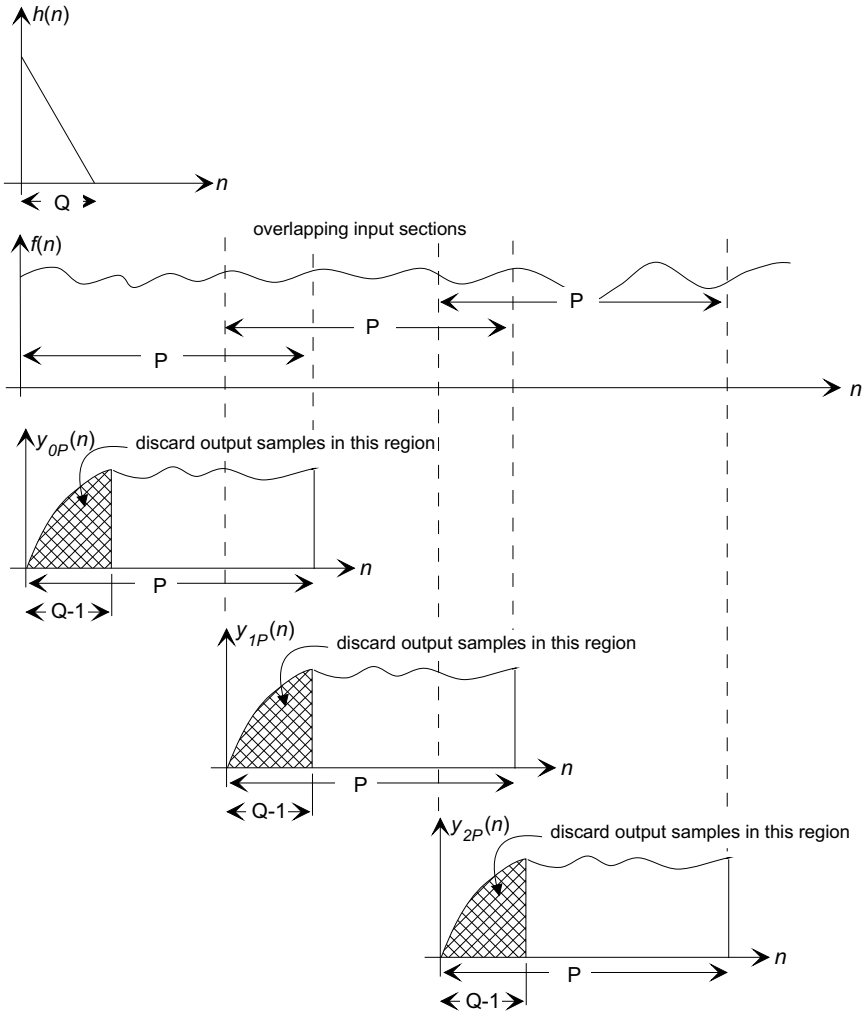
$$f_m(n) = f(n + m(P - (Q-1))), \qquad n = 0, \ldots, P-1,$$

and after DFT convolution in length P, giving $y_{mP}(n)$, the output is taken as

$$y_m(n) = \begin{cases} y_{mP}(n + (Q-1)), & n = 0, \ldots, P - (Q-1) \\ 0, & \text{otherwise.} \end{cases}$$

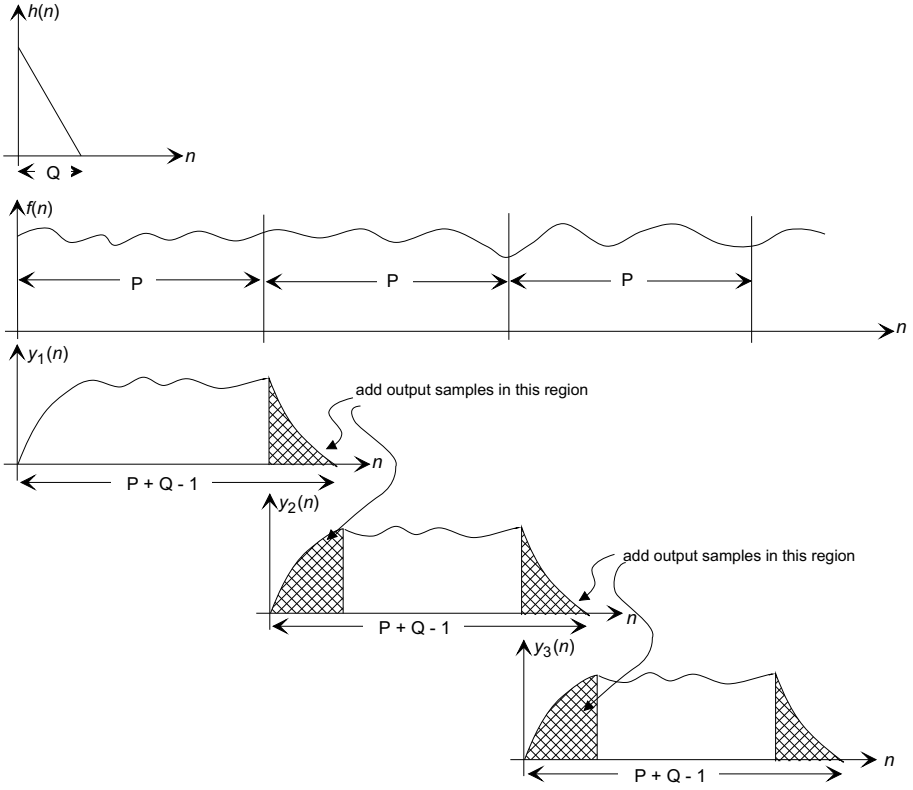
and the output is formed by concatenating all such records:

$$y(n) = \sum_{m=0}^{\infty} y_m(n - m(P - Q + 1)).$$



Overlap-Add Method: In this method the data is divided into blocks of length P, but the DFT convolution is done in zero-padded blocks of length $N = P + Q - 1$ so that

wrap-around errors do not occur. In this case the output is identical to the linear convolution of the two blocks, with an initial rise of length $Q-1$ samples, and a trailing section also of length $Q-1$ samples. It is easy to show that if the trailing section of the mth output block is overlapped with the initial section of the $(m+1)$th block, the samples add together to generate the correct output values.



MATLAB's `fftfilt()` function performs DFT convolution using the overlap-add method.

2 The Design of IIR Filters

An IIR filter is characterized by a recursive difference equation

$$y_n = \sum_{k=1}^{N} a_k y_{n-k} + \sum_{k=0}^{M} b_k f_{n-k}$$

and a rational transfer function of the form

$$H(z) = \frac{b_0 z_0 + b_1 z^{-1} + \ldots + b_M z^{-M}}{z_0 + a_1 z^{-1} + \ldots + a_N z^{-N}}$$

IIR filters have the advantage that they can give a better cut-off characteristic than a FIR filter of the same order, but have the disadvantage that the phase response cannot be well controlled.

The most common design procedure for digital IIR filters is to design a continuous filter in the s-plane, and then to transform that filter to the z-plane. Because the mapping between the continuous and discrete domains cannot be done exactly, the various design methods are at best approximations.

2.1 Design By Approximation of Derivatives:

Perhaps the simplest method for low-order systems is to use backward-difference approximation to continuous domain derivatives.

■ Example 1

Suppose we wish to make a discrete-time filter based on a prototype first-order high-pass filter

$$H_p(s) = \frac{s}{s+a}.$$

The differential equation describing this filter is

$$\frac{\mathrm{d}y}{\mathrm{d}t} + ay = \frac{\mathrm{d}f}{\mathrm{d}t}$$

The backward-difference approximation to a derivative based on samples taken at intervals T apart is

$$\frac{\mathrm{d}x}{\mathrm{d}t} \approx \frac{x_n - x_{n-1}}{T}$$

and substitution into the differential equation gives

$$\frac{y_n - y_{n-1}}{T} + ay_n = \frac{f_n - f_{n-1}}{T}$$

or

$$y_n = \frac{1}{1+aT}y_{n-1} + \frac{1}{1+aT}(f_n - f_{n-1})$$

The transfer function is

$$H(z) = \frac{1 - z^{-1}}{(1+aT)1 + z^{-1}} = \frac{z-1}{(1+aT)z + 1}$$

This example indicates that the method uses the transformation

$$s \to \frac{1 - z^{-1}}{T}$$

in $H_p(s)$. For higher order terms

$$s^n \to \left(\frac{1 - z^{-1}}{T}\right)^n.$$

■ Example 2

Convert the continuous low-pass Butterworth filter with $\Omega_c = 1$ rad/s to a digital filter with a sampling time $T = 0.5$ s. The transfer function is

$$H_p(s) = \frac{1}{s^2 + \sqrt{2}s + 1}.$$

The discrete-time transfer function is

$$\begin{aligned} H(z) &= \frac{1}{\left(\frac{1-z^{-1}}{T}\right)^2 + \sqrt{2}\left(\frac{1-z^{-1}}{T}\right) + 1} \\ &= \frac{T^2}{(1+\sqrt{2}T+T^2) - (2+\sqrt{2}T)z^{-1} + z^{-2}} \end{aligned}$$

and with $T = 0.5$ s,

$$H(z) = \frac{0.25}{1.9571 - 2.7071z^{-1} + z^{-2}}$$

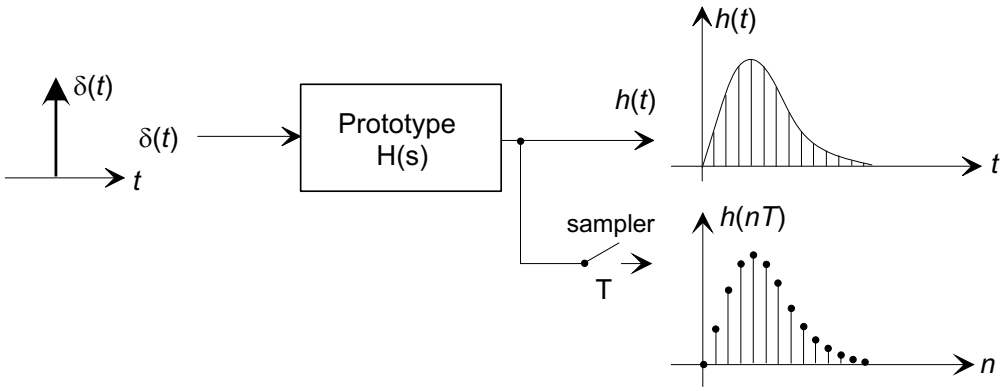
The frequency response of this filter is plotted in Example 4.

In general, the backward-difference does not lead to satisfactory digital filters that mimic the prototype filter characteristics. (See Proakis and Manolakis, Sec. 10.3.1).

2.2 Design by Impulse-Invariance:

In the *impulse-invariant* design method the impulse response $\{h_n\}$ of the digital filter is taken to be proportional to the samples of the impulse response $h_p(t)$ of the continuous filter $H_p(s)$ with a sampling interval of T seconds. The most common form is

$$h_n = T h_p(nT).$$



Then

$$\boxed{H(z) = T\mathcal{Z}\left\{h_p(nT)\right\} = T\mathcal{Z}_T\left\{\mathcal{L}^{-1}\left\{H_p(s)\right\}\right\}}$$

since $h_p(t) = \mathcal{L}^{-1}\left\{H_p(s)\right\}$, and where $\mathcal{Z}_T\left\{\right\}$ indicates the z-transform of a continuous function with sampling interval T.

■ Example 3

Find the impulse-invariant IIR filter from the prototype continuous filter

$$H_p(s) = \frac{a}{s+a}.$$

Solution: Using Laplace transform tables

$$h_p(t) = \mathcal{L}^{-1}\left\{\frac{a}{s+a}\right\} = a\,\mathrm{e}^{-at}.$$

and from z-transform tables

$$\mathcal{Z}_T\left\{a\,\mathrm{e}^{-at}\right\} = \frac{a}{1-\,\mathrm{e}^{-aT}z^{-1}}.$$

The IIR filter is

$$H(z) = T\mathcal{Z}_T\left\{a\,\mathrm{e}^{-at}\right\} = \frac{aT}{1-\,\mathrm{e}^{-aT}z^{-1}}$$

and the difference equation is

$$y_n = \,\mathrm{e}^{-aT}y_{n-1} + aTf_n$$

For the digital filter

$$H(\,\mathrm{e}^{\mathrm{j}\,\Omega T}) = \,H(z)|_{z=\,\mathrm{e}^{\mathrm{j}\,\Omega T}} = \sum_{k=0}^{\infty} h_k\,\mathrm{e}^{-\mathrm{j}\,k\Omega T}$$

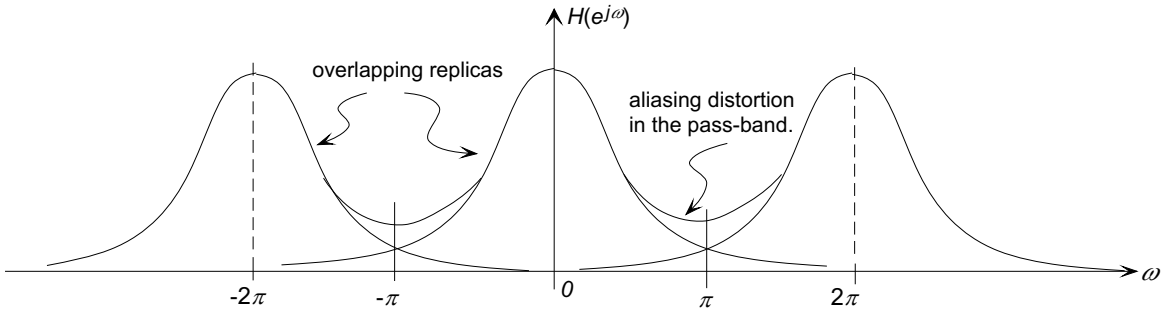
and the DTFT of the samples of the continuous prototype's impulse response is

$$\mathrm{DTFT}\left\{h_p(nT)\right\} = \sum_{k=0}^{\infty} h_p(kT)\,\mathrm{e}^{-\mathrm{j}\,k\Omega T} = \frac{1}{T}\sum_{k=-\infty}^{\infty} H_p\left(\mathrm{j}\,\left(\Omega - \frac{2\pi k}{T}\right)\right).$$

Then if $h_n = Th_p(nT)$,

$$H(\,\mathrm{e}^{\mathrm{j}\,\Omega T}) = \sum_{k=-\infty}^{\infty} H_p\left(\mathrm{j}\,\left(\Omega - \frac{2\pi k}{T}\right)\right).$$

The discrete-time frequency response is therefore a *superposition of shifted replicas* of the frequency response of the prototype. As a result, aliasing will be present in $H(\,\mathrm{e}^{\mathrm{j}\,\omega})$ if the prototype's frequency response $|H_p(\mathrm{j}\,\Omega)| \neq 0$ for $|\Omega| \geq \pi/T$.



For this reason the impulse-invariance method is not suitable for the design of high-pass or band-stop filters, which by definition require a prototype $H_p(s)$ with a non-zero frequency response at $\Omega = \pi/T$.

■ Example 4

Design an impulse-invariant filter based on the second-order low-pass Butterworth prototype used in Example 2, with $T = 0.5$ s.

$$H_p(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$$

Solution: From z-transform tables

$$\mathcal{Z}_T\left\{\mathcal{L}^{-1}\left\{\frac{\beta}{(s+a)^2+\beta^2}\right\}\right\} = \frac{\mathrm{e}^{-aT}\sin(\beta T)z}{z^2 + 2z\,\mathrm{e}^{-aT}\cos(\beta T)z + \mathrm{e}^{-2aT}}$$

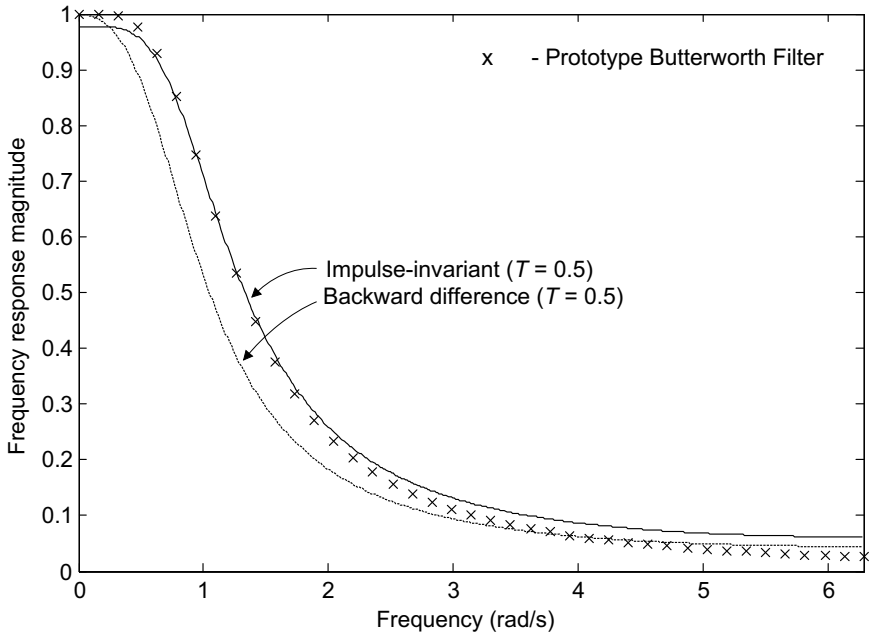
and $H_p(s)$ may be written in this form

$$H_p(s) = \frac{1}{(s+1/\sqrt{2})^2 + (1/\sqrt{2})^2}$$

so that $a = 1/\sqrt{2}$, and $\beta = 1/\sqrt{2}$. Substituting these values,

$$H(z) = T\mathcal{Z}_T\left\{\mathcal{L}^{-1}\left\{H_p(s)\right\}\right\} = \frac{0.1719z^{-1}}{1 - 1.3175z^{-1} + 0.4935z^{-2}}$$

The frequency response of the impulse-invariant, and backward-difference (from Example 2) filters are compared with the prototype below:



The MATLAB function

```
[bz, az] = impinvar(bs, as, Fs)
```

will compute the numerator `az`, and denominator `bz` coefficients for an impulse-invariant filter from the continuous prototype coefficients `bs` and `as`, with a sampling frequency `Fs`.

The filter in Example 4 can be designed in a single line:

```
[bz, az] = impinvar(1, [1 sqrt(2) 1], 2).
```