

Filters

Filters

- So far:
 - Sound signals, connection to Fourier Series, Introduction to Fourier Series and Transforms, Introduction to the FFT
 - Today Filters
- Filters:
 - Keep part of the signal we are interested in
- More generally:
 - Operate on a digital signal and transform it to have some desired property
- Wrap up signal processing

Convolution

Define a mathematical operation on discrete-time signals called **convolution**, represented by $*$.

Given two discrete-time signals x_1 , x_2 ,

$$\forall n , \quad (x_1 * x_2)(n) = \sum_{k=-\infty}^{\infty} x_1(k)x_2(n-k)$$

We can also define convolutions for continuous-time signals.

Given two continuous-time signals x_1 , x_2 ,

$$\forall t , \quad (x_1 * x_2)(t) = \int_{\tau=-\infty}^{\infty} x_1(\tau)x_2(t-\tau)d\tau$$

Properties of Convolution

Convolution is commutative: $x_1 * x_2 = x_2 * x_1$.

Convolution is associative: $(x_1 * x_2) * x_3 = x_1 * (x_2 * x_3)$.

Convolution also has the following properties:

$$x_1 * (x_2 + x_3) = x_1 * x_2 + x_1 * x_3$$

for scalars a , $x_1 * (ax_2) = a(x_1 * x_2)$

These stem from the fact that integration and summation have additivity and homogeneity properties.

Defining LTI Systems Using Convolution

When a system is excited it responds to the signal and the observed signal is a combination of the forcing signal and the system

Linear time invariant systems are linear, and their response does not change with time.

A Linear Time Invariant system can be defined using convolution.

Take any discrete-time function g .

The convolution

$$S(x)(n) = (g * x)(n) = \sum_{k=-\infty}^{\infty} g(k)x(n - k)$$

defines an LTI system S . The output for any input is obtained through convolution.

The properties of convolution ensure that S is LTI.

When we define a system this way, we call the defining function g the convolution **kernel**.

This method may be used to define continuous-time systems too.

Example

Consider the system defined by the kernel

$$\forall t \quad g(t) = \begin{cases} 1/3 & \text{if } t \in [0, 3] \\ 0 & \text{otherwise} \end{cases}$$

The output of this system for any input x is

$$y(t) = \int_{\tau=-\infty}^{\infty} g(\tau)x(t-\tau)d\tau \quad y(t) = \int_{\tau=0}^3 \frac{1}{3}x(t-\tau)d\tau$$

The output is the average value of x over the last 3 time units.

Impulse Response as Kernel

The output for a discrete-time Single Input Single Output LTI system can be given by the convolution of the input and the impulse response h :

$$y(n) = (h * x)(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k)$$

The impulse response is the particular system output obtained when the input is the Kronecker delta function

$$\forall n, \quad \delta(n) = \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{if } n \neq 0 \end{cases}$$

Dirac Delta Function

We define the impulse response h for a continuous-time system as the output for a particular input called the Dirac delta function.

The Dirac delta function is not really a function.

We define the Dirac delta function $\delta: \text{Reals} \rightarrow \text{Reals}$ to have the following properties:

$$\forall t \in \text{Reals} \setminus \{0\}, \quad \delta(t) = 0$$

$$\forall \varepsilon \in \text{Reals} \text{ with } \varepsilon > 0, \quad \int_{-\varepsilon}^{\varepsilon} \delta(t) dt = 1$$

Dirac Delta Function

So the Dirac delta function $\delta(t)$ has infinite value at $t=0$, and is zero everywhere else.

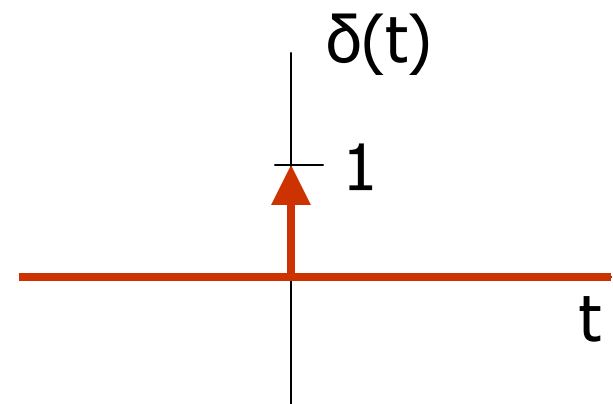
Yet, the area under the function is equal to 1.

If we scale the function by a , the function $a\delta(t)$ will still be zero for all nonzero t and infinite at $t = 0$.

But, the area under the function will now equal a .

We draw the Dirac delta function as an arrow pointing up at $t=0$ indicating its infinite value there.

The height of the arrow represents the **weight**, the area under the function.



Sifting Property

Both the Kronecker and Dirac delta functions have the following property: When a signal is convolved with a delta function, it remains unchanged. This is called the **sifting** property.

For the discrete-time case,
$$\sum_{k=-\infty}^{\infty} x(k)\delta(n-k) = x(n)$$

we see that the delta function is zero for all k except $k = n$.

So there is only one term in the sum, $x(n)\delta(n-n) = x(n)$.

For the continuous time case, $\delta(t-\tau)$ will be nonzero only for $\tau=t$.

So it doesn't matter what $\delta(t-\tau)$ multiplies for $\tau \neq t$:

$$\int_{\tau=-\infty}^{\infty} x(\tau)\delta(t-\tau)d\tau =$$

Impulse Response for Continuous-Time

The sifting property expresses x as a “sum of shifted and scaled delta functions”.

$$x(t) = \int_{\tau=-\infty}^{\infty} x(\tau)\delta(t - \tau)d\tau$$

If we apply an LTI system S to a sum of shifted and scaled delta functions, we can distribute over the sum (integral) and pull out scaling factors:

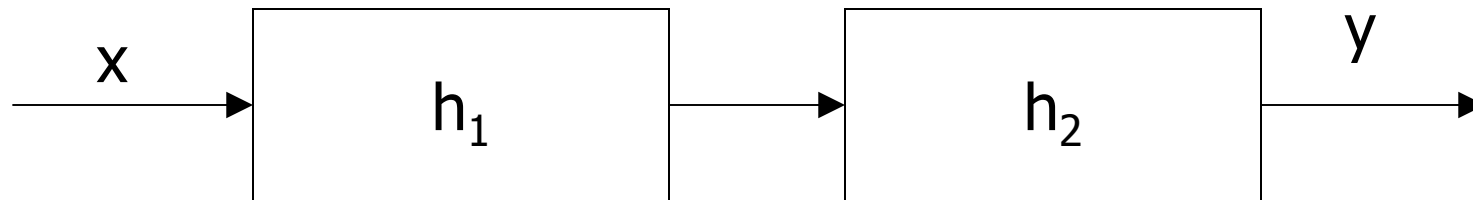
$$S(x)(t) = \int_{\tau=-\infty}^{\infty} x(\tau) S(\delta)(t - \tau) d\tau$$

$S(\delta)$ is the impulse response; the output of the system when the input is the Dirac delta function. We rename $S(\delta)$ as h , and see that the output $S(x) = x * h$, just like in discrete-time:

$$S(x)(t) = y(t) = \int_{\tau=-\infty}^{\infty} x(\tau) h(t - \tau) d\tau$$

Impulse Response for Cascade

Suppose we connect two systems with impulse responses h_1 and h_2 in cascade.



What is the impulse response of the composite system?

$$y = h_2 * (h_1 * x)$$

$$y = (h_2 * h_1) * x$$

The impulse response of the composite system is $h_2 * h_1$.

Relationship to Frequency Domain

- It was much easier to find the frequency response for a cascade system, rather than the impulse response.
- Given two systems in cascade with frequency responses H_1 and H_2 , the overall frequency response of the system was $H_1 H_2$.
- Sometimes it is easier to draw conclusions using the frequency response system description, and sometimes it is better to use the impulse response description.
- It turns out that the impulse response and frequency response are quite related, and we can find one from the other.
- We can transfer from the **time domain**, using the impulse response description, to the **frequency domain**, using the frequency response description.

Example

Find the frequency response of the system with impulse response

$$\forall t \in \text{Reals}, \quad h(t) = \begin{cases} 1/3 & \text{if } t \in [0, 3] \\ 0 & \text{otherwise} \end{cases}$$

$$H(\omega) = \int_{\tau=-\infty}^{\infty} h(\tau) e^{-i\omega\tau} d\tau$$

$$H(\omega) = \frac{i}{3\omega} \left(e^{-i3\omega} - 1 \right)$$

Convolution with Impulse Response

For both continuous and discrete-time LTI systems, we define the **impulse response** as the special output that occurs when the input is an impulse function (Dirac delta for continuous-time systems, Kronecker Delta for discrete-time systems).

This impulse response, called h , gives us the output for any input via convolution:

For x given,
 $\forall t \in \text{Reals},$

$$y(t) = (h * x)(t) = \int_{\tau=-\infty}^{\infty} h(\tau) x(t - \tau) d\tau$$

For x ,
 $\forall n$,

$$y(n) = (h * x)(n) = \sum_{k=-\infty}^{\infty} h(k) x(n - k)$$

Recall that the roles of h and x in the above may be reversed.

From Impulse Response to Frequency Response

Consider the case when the input to a continuous-time LTI system, x , is an eigenfunction given by $x(t) = e^{i\omega t}$.

We can find the output y using convolution with the impulse response h :

$$\forall t \in \text{Reals}, y(t) = \int_{\tau=-\infty}^{\infty} h(\tau) e^{i\omega(t-\tau)} d\tau = e^{i\omega t} \int_{\tau=-\infty}^{\infty} h(\tau) e^{-i\omega\tau} d\tau$$

Recall that an LTI system scales an eigenfunction $x(t) = e^{i\omega t}$ by a factor $H(\omega)$. From above, we see this scaling factor is

$$H(\omega) = \int_{\tau=-\infty}^{\infty} h(\tau) e^{-i\omega\tau} d\tau$$

The frequency response is the **Fourier transform** of the impulse response.

From Impulse Response to Frequency Response

For the discrete-time case, with $x(n) = e^{i\omega n}$ for $n \in \text{Integers}$,

We can find the output y using convolution with the impulse response h :

$$\forall n \in \text{Integers}, \quad y(n) = \sum_{k=-\infty}^{\infty} h(k)e^{i\omega(n-k)} = e^{i\omega n} \sum_{k=-\infty}^{\infty} h(k)e^{-i\omega k}$$

From above, we see the scaling factor on the eigenfunction is

$$H(\omega) = \sum_{k=-\infty}^{\infty} h(k)e^{-i\omega k}$$

The frequency response is the **discrete-time Fourier transform** (DTFT) of the impulse response.

Importance of H and h

We want to design systems with a particular frequency response H for many possible applications:

- Audio signal correction: getting rid of noise, compensating for an imperfect audio **channel**
- The equalizer on your stereo lets you modify your stereo's H
- Blurring and edge detection in images: an abrupt change in color in an image indicates high frequency, blocks of constant color are low frequencies
- AM radio reception: we want to keep a signal at one particular carrier frequency, and **filter** out neighboring carrier frequencies

Practical implementation of H , at least in digital systems, is easily accomplished by convolution with the corresponding h .

Designing FIR Discrete-Time Systems

We want to be able to specify a desired frequency response, $H(\omega)$ for all ω in Reals, and come up with the corresponding impulse response $h(n)$ for all n in Integers.

Our life will be easiest if $h(n)$ is nonzero only for a finite set of samples $n \in \{0, 1, \dots, L-1\}$. Then the length of the impulse response will be L , and the convolution sum will be finite:

$$y(n) = (h * x)(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) = \sum_{k=0}^{L-1} h(k)x(n-k)$$

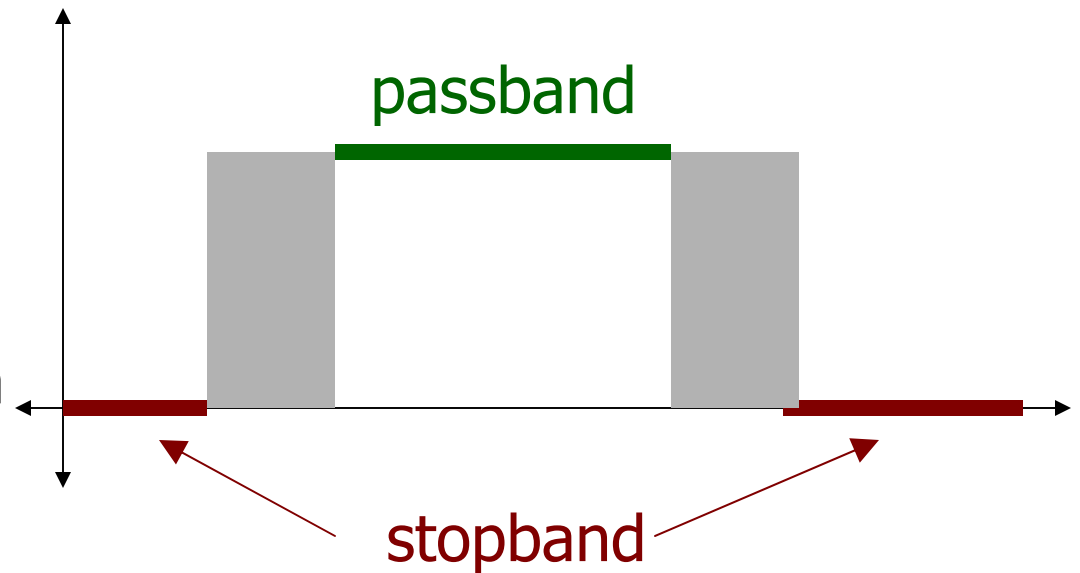
So we would like to come up with an FIR implementation of our desired frequency response function $H(\omega)$. MATLAB can help.

Designing FIR Filters Using MATLAB

The `remez()` function in MATLAB implements an algorithm called the Parks-McClellan algorithm based on the Remez algorithm.

It comes up with an impulse response of desired length L that implements a **bandpass** filter minimizing gain in the **stopband**.

You specify the **passband**, the frequencies your filter should amplify, the **stopband**, the frequencies it should zero out, and the **transition band**, the “don’t care” area of transition between 0 and 1.



Designing IIR Discrete-Time Systems

Notice that when we were designing our FIR system, with

$$y(n) = \sum_{k=0}^{L-1} h(k)x(n-k)$$

we were really picking the coefficients of a difference equation.

We can implement a wider variety of systems with the more general difference equation that involves feedback:

$$y(n) = \sum_{k=0}^{L-1} b(k)x(n-k) + \sum_{k=1}^{M-1} a(k)y(n-k)$$

These systems may have impulse responses of infinite duration (IIR), and the convolution sum would have infinite terms.

So, we design them by selecting the coefficients a and b and implement them using feedback as above.

Using MATLAB to Design IIR Filters

MATLAB has several functions that come up with the difference equation coefficients a and b to implement a system with a desired frequency response $H(\omega)$ for all ω in Reals. Using

<code>butter()</code>	Butterworth filter
<code>cheby1()</code>	Chebyshev 1 filter
<code>cheby2()</code>	Chebyshev 2 filter
<code>elliptic</code>	Elliptic filter

you can specify the passband, stopband, and order (number of terms in the difference equation) of your desired H .

MATLAB will return the corresponding coefficients in the difference equation.