Doc. no. ESS140/ext:09, rev. E

# Notes on Signals and Systems

Erik Ström

September 17, 2007

## 1 Introduction

This document should be viewed as a complement to Chapter 2 of Proakis and Salehi's book [2]. Many details are omitted and some mathematical rigor has been neglected to shorten the presentation (but, hopefully, without losing relevance). For an in-depth discussion of the material, we recommend the excellent book by Oppenheim and Willsky [1].

## 2 Continuous-Time and Discrete-Time Signals

We classify signals as being either *continuous-time* (functions of a real-valued variable) or *discrete-time* (functions of an integer-valued variable). To signify the difference, we (usually) use round parenthesis around the argument for continuous-time signals, e.g., $x(t)$ and square brackets for discrete-time signals, e.g., $x[n]$. We will also use the notation $x_n$ for discrete-time signals.

One possible way to create a discrete-time signal, $x_d[n]$, is by sampling a continuous-time signal, $x_c(t)$, every $T_s$ seconds:

$$x_d[n] = x_c(nT_s), \qquad \text{for } n = 0, \pm 1, \pm 2, \dots$$

We refer to $T_s$ and $f_s = 1/T_s$ as the sample interval and sample frequency, respectively.

We will assume that signals are complex-valued if not explicitly stated otherwise.

## 3 Useful Signals

The *unit step function* is defined as

$$u(t) = \begin{cases} 1, & t > 0 \\ 0, & t < 0 \end{cases}.$$

We will leave the value of $u(t)$ at $t = 0$ undefined for the time being. If the reader feels uneasy about this, he or she can use $u(0) = 1/2$. Note that the unit step function is denoted by $u_{-1}(t)$ in the text book [2].

The discrete-time unit step function is defined as

$$u[n] = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases}.$$

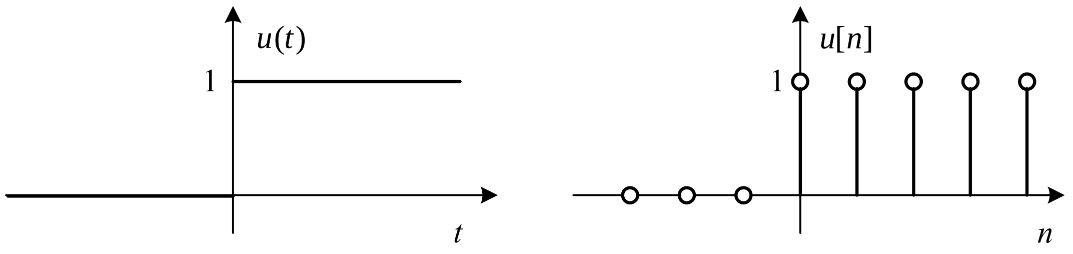

Figure 1: Unit step functions

The signal $\Pi(t)$ is a square pulse of unit width

$$\Pi(t) = \begin{cases} 1, & |t| < \frac{1}{2} \\ 0, & |t| > \frac{1}{2} \end{cases}.$$

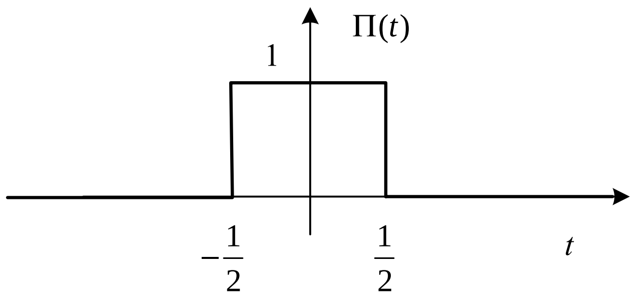

Figure 2: Rectangular function

The continuous-time impulse (or *Dirac delta function*) is a strange signal. To make a long story short, we will think of the Dirac delta function as the limit of the function $\delta_\varepsilon(t)$, which is defined as

$$\delta_\varepsilon = \begin{cases} \frac{1}{\varepsilon}, & |t| < \frac{\varepsilon}{2} \\ 0, & |t| > \frac{\varepsilon}{2} \end{cases}.$$

Hence, as seen in Figure 3, $\delta_\varepsilon(t)$ is a square pulse of height $1/\varepsilon$ and width $\varepsilon$. The Dirac delta function, $\delta(t)$, can be defined as the limit of $\delta_\varepsilon(t)$ when $\varepsilon \to 0$. We can therefore think of the Dirac delta function as an infinitely high and infinitely thin square pulse. Such a signal has the peculiar property that for any function $f(x)$ (assumed to be continuous at $x = 0$),

$$\int_{-\infty}^{\infty} f(x)\delta(x)\,dx = f(0).$$

As a matter of fact, the above equation is often taken as the definition of $\delta(t)$.

The discrete-time impulse (or *Kronecker delta function*) is an ordinary signal defined as

$$\delta[n] = \begin{cases} 1, & n = 0 \\ 0, & \text{otherwise} \end{cases}.$$

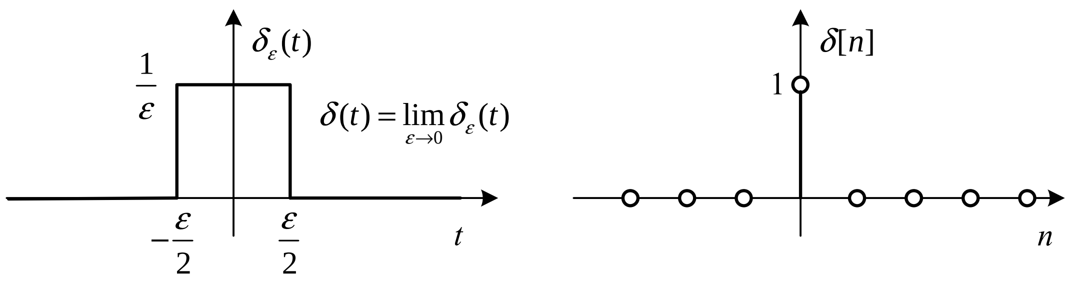

Figure 3: Delta functions

## 4 Time-Shift, Time-Compression, and Folding

We will frequently need to delay or advance signals in time. Mathematically, this is achieved by substitution of the time variable. For instance, if we want $y(t)$ to be equal to $x(t)$ delayed with $T$ seconds, the appropriate substitution is

$$y(t) = x(t)|_{t \to t-T} = x(t - T).$$

Conversely, if we want $z(t)$ to be equal to $x(t)$ advanced in time with $T$ seconds, then

$$z(t) = x(t)|_{t \to t+T} = x(t + T).$$

We will also find it useful to fold signals, i.e., to reflect or mirror the signal around the point $t = 0$ on the time-axis. This is done by the substitution $t \to -t$. Hence,

$$w(t) = x(t)|_{t \to -t} = x(-t),$$

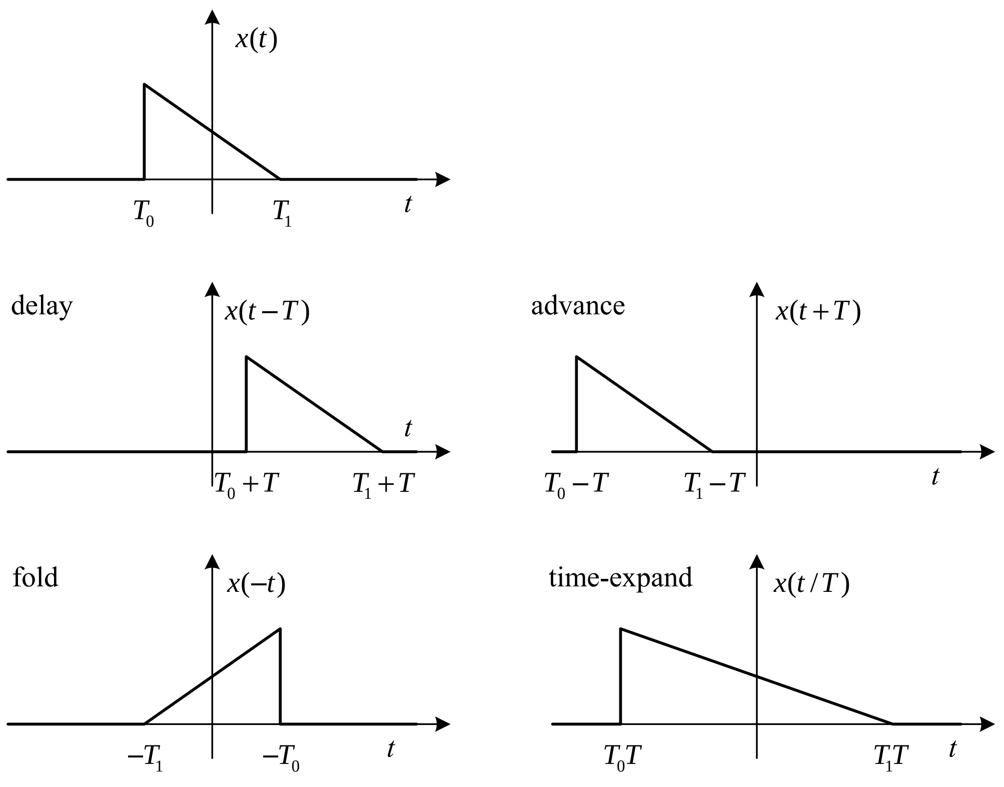

Figure 4: Delayed, advanced, folded and time-expanded versions of $x(t)$.

is a folded version of $x(t)$.

Another substitution of interest is $t \to t/T$, which corresponds to time-expanding the signal by a factor of $T$. For instance, the signal $\Pi(t/T)$ is a rectangular pulse of width $T$ seconds. Of course, we can choose $T < 1$, and then the pulse $\Pi(t/T)$ will be thinner than $\Pi(t)$.

A signal $x(t)$ and its transformations is found in Figure 4. A summary of the fundamental substitutions is found in Table 1.

| Substitution | Interpretation |
|---|---|
| $t \to t - T$ | delay (right-shift) with $T$ seconds |
| $t \to t + T$ | advance (left-shift) with $T$ seconds |
| $t \to -t$ | fold (mirror) around the point $t = 0$ |
| $t \to t/T$ | expansion with a factor $T$ |

Table 1: Fundamental substitutions of the time-variable

The fundamental substitutions can be combined to obtain more complicated transformations. For instance, if we want to fold a signal $x(t)$ and delay it by $T$

seconds, then this can be done as

$$x(t) \xrightarrow[\text{fold}]{t\to -t} x(-t) \xrightarrow[\text{delay}]{t\to t-T} x(-(t-T)) = x(T-t).$$

If we think about it for a while, we realize that we can achieve the same result by first *advancing* the signal by $T$ seconds and then folding it:

$$x(t) \xrightarrow[\text{advance}]{t\to t+T} x(t+T) \xrightarrow[\text{fold}]{t\to -t} x(-t+T)) = x(T-t).$$

Please note that a common mistake is to believe that the folded version of $x(t+T)$ is $x(-(t+T)) = x(-t-T)$, which obviously is wrong. The process is illustrated in Figure 5.

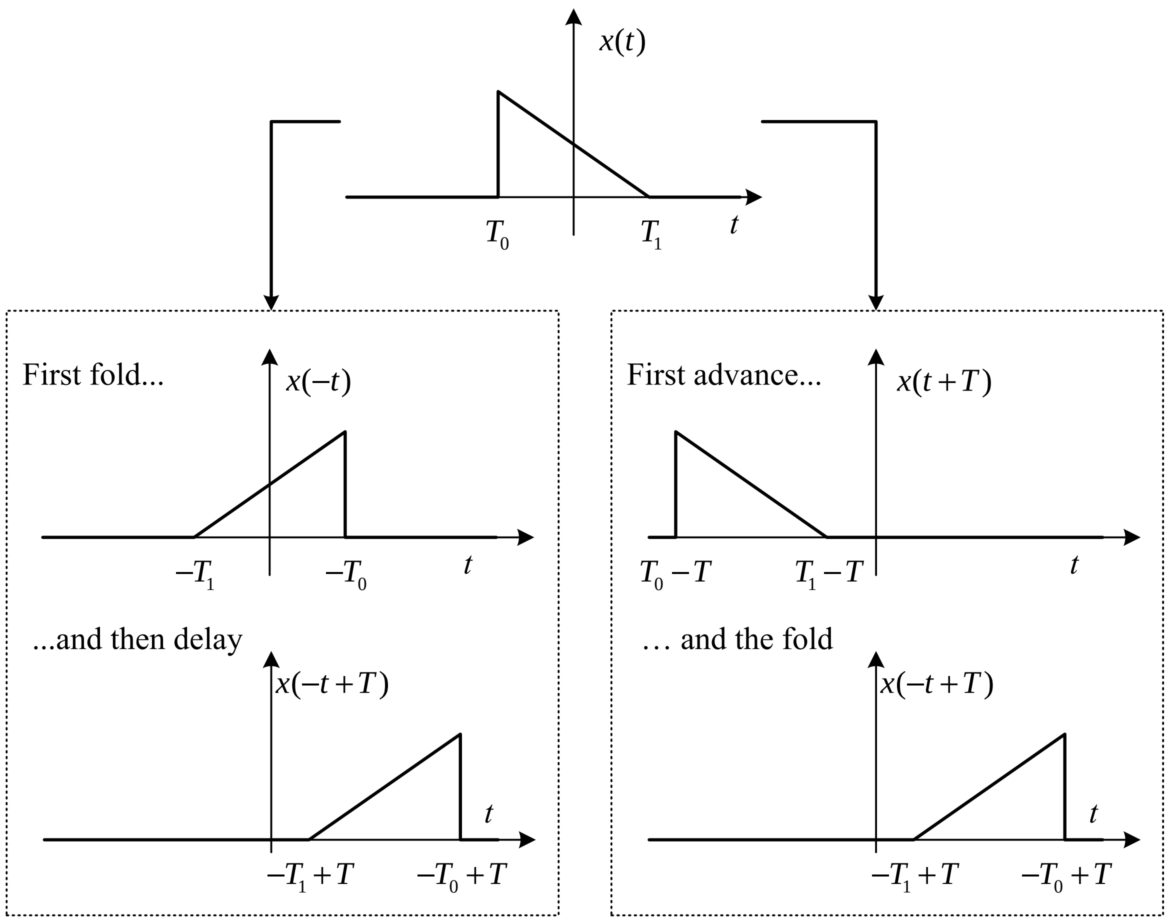

Figure 5: Two ways to arrive at $x(-t+T)$.

# 5 Systems

A system is an entity that takes an input signal and produces an output signal. Systems can be linear or nonlinear and time-invariant or time-varying.

## 5.1 Linear Systems

The system is *linear* if it follows the *superposition principle.* The superposition principle says that the output to a linear combination of input signals is the same linear combination of the corresponding output signals. To be precise, if the input signals $x_1(t)$ and $x_2(t)$ corresponds to the output signals $y_1(t)$ and $y_2(t)$, respectively, then the input signal $a_1x_1(t) + a_2x_2(t)$ should correspond to the output signal $a_1y_1(t) + a_2y_2(t)$ for any constants $a_1$ and $a_2$.

## 5.2 Time-Invariant Systems

In words, a *time-invariant system* is a system which does not change with time. Mathematically, if the input $x(t)$ gives the output $y(t)$, then the system is time-invariant if the input $x(t-T)$ gives the output $y(t-T)$ for any delay $T$. Hence, a time-shift of the input gives the same time-shift of the output.

## 5.3 Linear Time-Invariant Systems

*Linear time-invariant* (LTI) systems are commonly used in signal processing systems. LTI systems have many nice properties, e.g., an LTI system is completely described by its *impulse response.* The impulse response is the output when the input is an impulse, i.e., a Dirac delta function for continuous-time systems or a Kronecker delta function for discrete-time systems.

It is not difficult to show that if $h(t)$ is the impulse response of an LTI system, then the output to the input $x(t)$ can be found as

$$y(t) = \int_{-\infty}^{\infty} x(u)h(t-u)\,du = x(t) * h(t). \tag{1}$$

This integral is known as a *convolution* integral and $*$ is used to denote convolution. To visualize the integrand $f(u) = x(u)h(t-u)$, it is important to remember that the integrand should be considered as function of $u$ (the integration variable). Hence, $h(t-u)$ should be thought of as being $h(u)$ folded and delayed by $t$ seconds (or, equivalently, $h(u)$ advanced by $t$ seconds and then folded). See Figure 6 for an example. With a bit of practice, it will be easy to visualize the integrand, which is really helpful to illuminate convolution. For instance, we realize that for each new value of $t$, we will shift $h(-u+t)$ to a new position. The integrand and integral will therefore (probably) change.

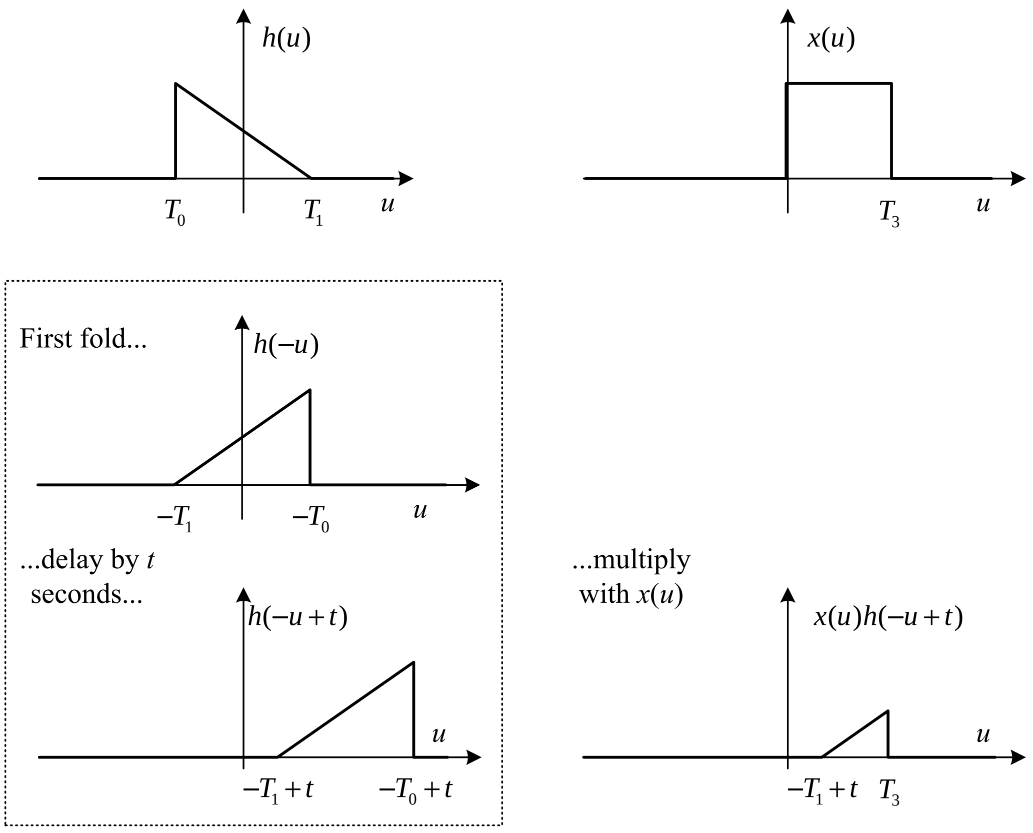

Figure 6: An example how to compute $x(u)h(t-u)$.

Similarly, the output of a discrete-time LTI system with impulse response $h[n]$ and input $x[n]$ is the convolution between $x[n]$ and $h[n]$,

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] = x[n] * h[n].$$

As seen, convolution is an operation on two signals which results in a third signal. Convolution is *commutative,*

$$x(t) * h(t) = h(t) * x(t),$$

*associative,*

$$[x(t) * y(t)] * z(t) = x(t) * [y(t) * z(t)],$$

and *distributive,*

$$x(t) * [y(t) + z(t)] = x(t) * y(t) + x(t) * z(t).$$

These properties applies for discrete-time convolution as well.

## 5.4 Correlation and Convolution

The *crosscorrelation function* between $x(t)$ and $y(t)$ is defined as

$$R_{xy}(t) = \int_{-\infty}^{\infty} x(t+u)y^*(u)\,du,$$

where $y^*(t)$ is the complex conjugate of $y(t)$. When just speaking about the *correlation* between $x(t)$ and $y(t)$, we (usually) mean $R_{xy}(0)$.

We note that the crosscorrelation function is quite related with convolution. In fact, we can compute the crosscorrelation function above by processing $x(t)$ with an LTI system with impulse response $h(t) = y^*(-t)$. Such as system is called a filter that is *matched* to $y(t)$. To show that the output is actually $R_{xy}(t)$, we recall that the filter output, $z(t)$, is the convolution between $x(t)$ and $h(t)$,

$$\begin{aligned} z(t) &= x(t) * h(t) \\ &= \int_{-\infty}^{\infty} x(u) \underbrace{h(t-u)}_{=y^*(-(t-u))} du \\ &= \int_{-\infty}^{\infty} x(u)y^*(u-t)\,du \\ &= \int_{-\infty}^{\infty} x(v+t)y^*(v)\,dv \\ &= R_{xy}(t). \end{aligned}$$

A short-hand notation for the above relation is

$$R_{xy}(t) = x(t) * y^*(-t).$$

The connection between the crosscorrelation function and convolution is quite useful for understanding how digital communication receivers can be implemented.

The *autocorrelation function* of $x(t)$ is $R_{xx}(t)$, i.e., the crosscorrelation of $x(t)$ with itself,

$$R_{xx}(t) = \int_{-\infty}^{\infty} x(t+u)x^*(u)\,du.$$

An interesting property of the autocorrelation function is that $R_{xx}(0) = E_x$, where $E_x$ is the energy of $x(t)$,

$$E_x = \int_{-\infty}^{\infty} |x(u)|^2\,du.$$

The concept of signal energy plays a fundamental role in communications.

Another very useful property is that $|R_{xx}(t)|$ attains its maximal value $E_x$ when $t = 0$. To prove this, we note that, from the Schwarz inequality[1],

$$|R_{xx}(t)|^2 = \left| \int_{-\infty}^{\infty} x(t+u)x^*(u)\,du \right|^2 \le \int_{-\infty}^{\infty} |x(t+u)|^2\,du \int_{-\infty}^{\infty} |x^*(u)|^2\,du = E_x^2,$$

where the inequality hold with equality if and only if $x(t+u) = \alpha x(u)$ for some complex constant $\alpha$. Hence, $|R_{xx}(t)| \le E_x$ for all values of $t$, but since $R_{xx}(0) = E_x$, this proves that $|R_{xx}(t)|$ attains its maximal value ($E_x$) when $t = 0$.

This fact is very useful for understanding how digital communication receivers can be synchronized.

## 5.5 Frequency Domain Representations of Signals and Systems

We use Fourier transforms and Fourier series to represent signals as linear combinations of complex exponentials. A continuous-time complex exponential with amplitude $A$, frequency $f_0$, and phase $\phi$ (the quantities $A$, $f_0$, and $\phi$ are real numbers) can be written as

$$z(t) = Ae^{j(2\pi f_0 t+\phi)} = A\cos(2\pi f_0 t + \phi) + jA\sin(2\pi f_0 t + \phi).$$

Obviously, the complex exponential is a complex-valued function which is periodic with fundamental period $1/f_0$. The signal $z(t)$ represents a single-frequency component at the frequency $f_0$.

If $z(t)$ is the input to an LTI system with impulse response $h(t)$, then the output is

$$\begin{aligned} y(t) &= h(t) * z(t) \\ &= \int_{-\infty}^{\infty} h(u)z(t-u)\,du \\ &= \int_{-\infty}^{\infty} h(u)e^{j(2\pi f_0(t-u)+\phi)}\,du \\ &= \underbrace{e^{j(2\pi f_0 t+\phi)}}_{=z(t)} \underbrace{\int_{-\infty}^{\infty} h(u)e^{-j2\pi f_0 u}\,du}_{=H(f_0)} \\ &= z(t)H(f_0) \end{aligned}$$

[1]The Schwarz inequality (also known as the Cauchy-Schwarz inequality) says that if $g(t)$ and $h(t)$ have finite energies, then

$$\left| \int_{-\infty}^{\infty} g(t)h(t)\,dt \right|^2 \le \int_{-\infty}^{\infty} |g(t)|^2\,dt \int_{-\infty}^{\infty} |h(t)|^2\,dt$$

with equality if and only if $g(t) = \alpha h^*(t)$, where $\alpha$ is a complex constant.

where $H(f)$ is the Fourier transform of the impulse response, also known as the *frequency response* of the system,

$$H(f) = \int_{-\infty}^{\infty} h(t)e^{-j2\pi ft}\, dt.$$

We conclude that the output to a complex exponential is the same exponential scaled with the frequency response of the system.

By invoking the inverse Fourier transform, we can write $h(t)$ as

$$h(t) = \int_{-\infty}^{\infty} H(f)e^{j2\pi ft}\, df.$$

Hence, $h(t)$ can be seen as a sum of many complex exponentials (frequency components), and the frequency component at frequency $f$ has complex amplitude $H(f)$. Hence, $H(f)$ represents the signal's distribution in frequency.

Now, for an input signal $x(t)$ with Fourier transform $X(f)$, the output to an LTI system with impulse response $h(t)$ and frequency response $H(f)$ is $y(t) = h(t) * x(t)$. The Fourier transform of $y(t)$ can be shown to be

$$Y(f) = H(f)X(f).$$

## 5.6 Simulation of Continuous-Time Systems

It is common to simulate systems to verify designs and theoretical calculations. In a tool like MATLAB, it is very easy to perform discrete-time convolutions between time-limited signals. Suppose that we have the signals $x[n]$ and $h[n]$ stored in the MATLAB vectors `x` and `h`, respectively. Then the convolution $z[n] = h[n] * x[n]$ can be computed as

```
>> z = conv(h, x);
```

However, for continuous-time systems, we must in general use numeric integration to compute convolutions. If we want to convolve $x(t)$ with $h(t)$, then the integral that needs to be computed is

$$y(t) = \int_{-\infty}^{\infty} h(u)x(t-u)\, du.$$

The trapezoidal rule for numerical integration says that

$$\int_{-\infty}^{\infty} h(u)x(t-u)\, du \approx \sum_{k=-\infty}^{\infty} h(kT_s)x(t-kT_s)T_s,$$

where $T_s$ is the step size or sample interval. If we compute samples of $y(t)$ with the same sample interval, we arrive at the equation

$$\begin{aligned} y(nT_s) &= \int_{-\infty}^{\infty} h(u)x(nT_s - u)\, du \\ &\approx \sum_{k=-\infty}^{\infty} h(kT_s)x(nT_s - kT_s)T_s \\ &= T_s \sum_{k=-\infty}^{\infty} h(kT_s)x([n-k]T_s). \end{aligned} \tag{2}$$

The summation in the last line is nothing else but the convolution between the discrete-time signals $h_d[n] = h(nT_s)$ and $x_d[n] = x(nT_s)$.

The quality of the approximation becomes better as $T_s$ decreases. As a matter of fact, if $x(t)$ and $h(t)$ are *bandlimited,* e.g., if $X(f) = H(f) = 0$ for $|f| > W$, then the approximation error disappear if $T_s < 1/(2W)$. (This is a consequence of the *sampling theorem*, which is proved and discussed at length in any decent signals and system book.)

To compute (2) in practice, the signals must also be truncated to make the vectors have finite length. This obviously leads to approximation errors if the signals are not time-limited. However, if the main features of the signals remain after truncation, the approximation may still be of use.

To summarize, we can compute samples of $y(t) = h(t) * x(t)$ as

1. Choose a sample interval $T_s$ that is sufficiently small.
2. If $x(t)$ and $h(t)$ are not time-limited, then truncate them such that the main part of each signal still remains.
3. Sample the signals and store them in MATLAB vectors `x` and `h`.
4. Compute samples of $y(t)$ as `y = T_s * conv(x, h);`.

## 6 References

[1] Alan V. Oppenheim and Alan S. Willsky. *Signals and Systems.* Prentice-Hall, Upper Saddle River, New Jersey 07458, USA, second edition, 1997.

[2] John G. Proakis and Masoud Salehi. *Communication Systems Engineering.* Prentice-Hall, second edition, 2002.