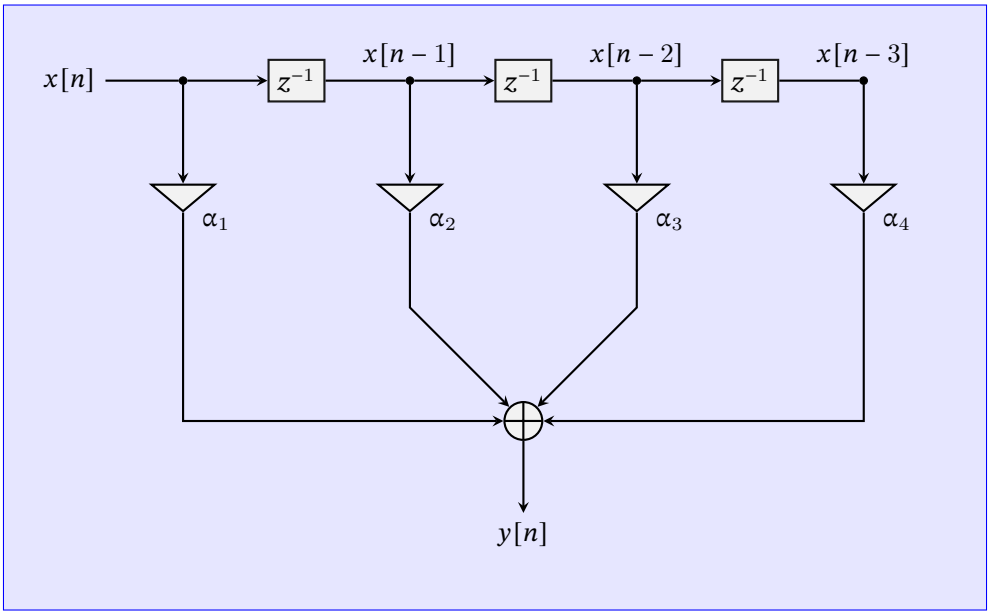
Università degli Studi di Trieste

Marco Sgobino

Notes from the course of

Digital Signal Processing



October 31, 2022

Academic Year 2021-2022

Contents

Part I

Discrete Time Systems

Chapter 1

Time-Domain Representation of Discrete-time Signals

Time domain refers to the analysis of mathematical functions, physical signals or time series of economic or environmental data, with respect to *time*. In the time domain, the signal or function's value is known for all real numbers, for the case of continuous time, or at various separate instants in the case of discrete time. An oscilloscope is a tool commonly used to visualize real-world signals in the time domain. A time-domain graph shows how a signal changes with time, whereas a frequency-domain graph shows how much of the signal lies within each given frequency band over a range of frequencies.

Though most precisely referring to time in physics, the term time domain may occasionally informally refer to position in space when dealing with spatial frequencies, as a substitute for the more precise term spatial domain [1].

1.1 Signal definition

In signal processing, a **signal** is a function that conveys information about a phenomenon. Signals are functions of independent variables, such as *time*, *distance*, *position*, and so on. They are represented as sequences of numbers, called *samples*. Signals are composed of samples—the single values of the signal at a certain, discrete time quantum $n \in \mathbb{Z}$. Signals will be denoted with $\{x[n]\}$ while samples will be denoted as $x[n]$. The variable n may or may not be an integer, since the quantum of time may not be constant in some applications.

Typically a discrete signal can be either a *sampled-data signal* or a *digital* signal. Both types of signals only exist in specific time quanta and they are likely the result of a sampling, but only digital signals have *quantized* value—common sampled-data signals will possess continuous values. Values of digital signals are usually obtained from sample values either by *rounding* or *truncation* techniques—in both cases an error occurs.

Since signals can have multiple time domains, one has to clarify when equivalences between amplitudes may apply. In particular, the following relationship

$$x[n] = x_a(t)|_{t=nT} = x_a(nT), n \in \mathbb{Z}$$

yields only as long as we do not take in account any *quantization error*, since—by definition—digital signals are the result of both a sampling process and a quantization process. The spacing between two samples is said to be *sampling interval* or *sampling period* and measures in seconds, and the reciprocal of such quantity is called *sampling frequency* F_T or F_S and measures in Hertz.

1.2 Basic signal properties

Nature of signals

Signals can be either *deterministic* or *random*; the first case denotes a signal that can be inferred by some rule or formula, while a random signal cannot be generally predicted ahead of time. Examples of random signals are the *stock exchange*, or the *coin flip*. Random signals are usually considered as they are generated in a random fashion—even though there might exist some actual underlying rule we don't actually know.

Filters

A *filter* is a system which *is able to control the frequency content of the output with respect to the input signal*. When a filter is

linear, the output of the filtering operation sums up to the *convolution* operation, which is the following signal,

$$y(t) = \int_0^T x(t-\tau)h(\tau)d\tau.$$

The role of τ in the convolution is to shift-reverse one of the two signals, while the multiplication with the other signal occurs. This ensures that the overall result will be a new signal which is the sum, sample-to-sample, between one of the two signals in its original form and a "moving" shift-reversed form of the other one.

Visual representation of a signal

A signal can be also represented as a sequence of values inside brackets. To denote where the time instant 0 is, an arrow is placed below the sample of the sequence corresponding to $n = 0$. As an example, the following notation can express a signal:

$$\{x[n]\} = \{\dots, -0.2, 1, \underset{\uparrow}{3}, -6.7, -5.4, 0.1, 3.6, 1.2, \dots\}$$

and, with this notation, $x[0] = 3$.

To represent signals, a *stem plot* is usually adopted. There are two different forms of stem plot: a first one in which only samples are represented—dots along with their straight vertical line denoting the *amplitude* of the signal in such quantum of time—and another one in which superposed to the samples an analog-like line is represented. In the latter case, there may be two different interpretations; either the digital signal is sampled from an original analog signal, or the continuous line denotes the amplitude of the digital signal *as if* it were analog.

Two kinds of digital signal representation exist. The first type is the common digital signal where each sample is represented with a dot and a straight line, while a *boxedcar* signal will see each sample represented as a straight line, in a similar fashion to the interpolation.

Complex signals

A signal can be either real or complex valued. A *real sequence* comprises only real samples such that $x[i] \in \mathbb{R}, \forall i \in \mathbb{Z}$ while a *complex sequence* has at least one complex sample $x[k] \in \mathbb{C}$. Complex sequences may be seen as the result of

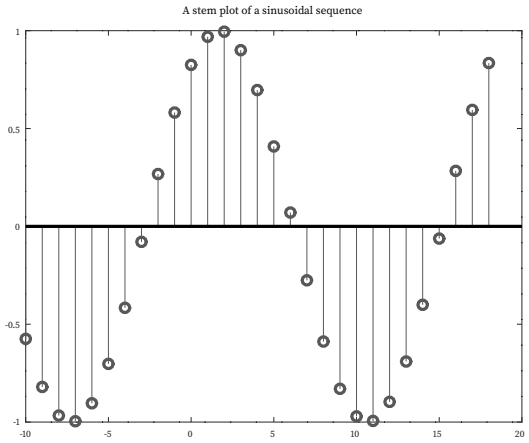


Figure 1.1: *Stem plot* of a sinusoidal function. In particular, a stem plot of `cos(.35*x - .6)` has been performed with the help of the `octave` CLI. Pre-built binaries of the program are available at their website `https://octave.org/`.

the sum of two, distinct, real sequences. In particular, by writing

$$\{x[n]\} = \{x_{re}[n]\} + j\{x_{im}[n]\}$$

two different sequences, one for real part and the other one for the imaginary part, are summed to form a single complex signal. It's easy to understand that a real valued signal can still be written as a complex signal, provided that $x_{im}[n] \equiv 0, \forall n$.

A complex signal's short, alternative notation is

$$\{x[n]\} = \{A[n]e^{-j \cdot f_i n}\},$$

where the imaginary and real part are intrinsic in the exponential representation and can thus be obtained by decomposing it into a sine and a cosine addenda.

Length of a signal

A signal can either be of *finite* or *infinite*-length. In the first case defined values for the sequence lay only in an interval $N_1 \leq n \leq N_2$, with $N_1 \leq N_2$ and both $-\infty < |N_1| \leq |N_2| < \infty$. The overall *length* or *duration* of the sequence will be $N = N_2 - N_1 + 1$. In the second case a sequence is defined for every value of $n \in \mathbb{Z}$, after a certain n_0. For instance, $\{\cos f_0 n\}$ is an infinite-length sequence, because the signal "never ends", as there is no n_1 after which or before which

the signal is null. Other infinite-length signals are any signal defined for all values after an $n_0 \in \mathbb{Z}$.

In practice, no signal really has infinite length, but it could be useful to still think of signals as such to make them more mathematically treatable.

Zero-padding

A signal may see its length varied by arbitrary operations. A very common operation in that regard, is to increase the length of a signal by appending or prepending 0s to the signal, with a process called **zero-padding**. This usually involves a finite-length signal becoming an infinite-length signal by applying an infinitely long list of 0s on both sides of the original signal; it is not uncommon, however, to apply the zero-padding technique to only one side of the signal. Another way to handle zero-padding is to simply add a finite—yet necessary—number of 0s depending on the duty to perform.

Causality

A sequence is said to be *right-sided* if it has zero-valued samples for each $n < N_1$. A special case of right-sided sequence is when $N_1 \geq 0$, the case of a **causal** signal.

Vice-versa, a sequence is said to be *left-sided* if it has zero-valued samples for each $n > N_2$. The same way, when a left-sided sequence has $N_2 \leq 0$, we are in the case of an **anti-causal** signal.

Basically, causal signals are all signals that "are relevant only *after* a certain time N_1", and anti-causal signals are all signals that "are relevant *before* a given time N_2".

Strength of a signal and signal comparison

An important concept is the **strength** of a signal, also called *size* of a signal. The strength of a signal is the $\mathcal{L}_p$-norm of the sequence,

$$||x||_p = \left(\sum_{i=N_1}^{N_2} |x[n]|^p \right)^{\frac{1}{p}},$$

where $p \in \mathbb{N} \setminus \{0\}$ is a positive integer, which is different from zero. Typically, one has $p = 1, 2, \infty$, with $p = 2$ being the *root mean squares* apart from a scaling factor—RMS is the square root of the **mean** of the squares, there's an additional $\frac{1}{\sqrt{n}}$ factor. The other two cases are $p = 1$ corresponding to the *mean absolute value* of the signal—except from a constant factor—and $p = \infty$ where the strength of the signal would be the *absolute peak value* of the signal.

From the concept of signal strength the concept of *relative error* immediately follows. In particular, let $\{y[n]\}, 0 \leq n \leq N-1$ be an approximation of the signal $\{x[n]\}, 0 \leq n \leq N-1$. In this case, one calls relative error the quantity

$$E_{rel} = \left(\frac{\sum_{n=0}^{N-1} |y[n] - x[n]|^2}{\sum_{n=0}^{N-1} |x[n]|^2} \right)^{\frac{1}{2}}, \tag{1.1}$$

which is the ratio of the $\mathcal{L}_2$-norm of the difference signal and the $\mathcal{L}_2$-norm of the signal $\{x[n]\}$. Everything is square-rooted.

The numerator of such quantity can be considered as the error between sequences x and y, while the denominator is the 2-strength of a non-approximated sequence.

1.2.1 Operations on sequences

sequences may be subjected to *operations*. In particular, operations can be described by directed graphs in which there is an input sequence and an output sequence resulting from the operation. Of course, there may be more than one inputs or outputs.

Figure 1.2: The general scheme for elementary operations.

Basic signal operations are called **elementary operations**. The first elementary operation is the *modulation*, an operation in which a signal x is multiplied by another signal, say w, to produce an output signal y such that

$$y[n] = x[n] \cdot w[n].$$

A common application of the modulation is the *windowing* process, in which an infinite-length signal x is "windowed" by a finite-length signal w, whose non-zero values are limited to a finite interval and the non-zero values are all equal to 1. In that case, it would be like "selecting" the values of x that lay in the w window. Modulation is denoted with a cross inside a circle, as shown in Diagram 1.3.

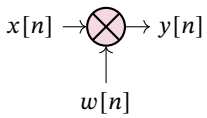


Figure 1.3: Notation for the modulation (product) operation.

Other elementary operations are the *multiplication* operation and the *addition* operation. The multiplier is denoted by a triangle—an opamp-like symbol is adopted—while the addition is a plus symbol inside a circle. Diagrams 1.4 and 1.5 summarize the overall notation. This time, multiplication is not performed between entire signals, but instead a constant A multiplies a signal x, such that

$$y[n] = A \cdot x[n].$$

The addition is performed between two distinct signals, hence

$$z[n] = x[n] + y[n].$$

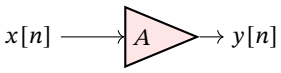


Figure 1.4: Notation for the multiplication (product for a constant) operation.

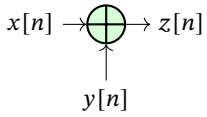


Figure 1.5: Notation for the addition (sum) operation.

Much relevant elementary operations are, indeed, the *time delay* and *time advance*. A signal that is time-delayed is a signal which is a copy of the original signal, but translated in time (delayed in this case). Vice-versa, in the case of time advance, the signal copy of the original will be anticipated. In the first case, the adopted symbol is z^{-1}, while for the anticipation it is z. Hence, $y[n] = x[n-1]$ and $y[n] = x[x+1]$.

An interesting operation is the *time reversal* or *folding*. The folding operation inverts the time axis of a signal, that means $y[n] = x[-n]$. Time reversal does not have any explicit symbol, since it requires a complete reflection of a signal from its origin point.

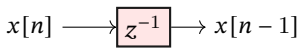


Figure 1.6: Notation for the time delaying operation.

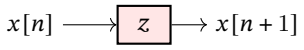


Figure 1.7: Notation for the time advance operation.

Yet another common 'operation' is the *branching* operation, in which a signal w is simply the copy of the input signal, as well as the y signal (a single input is copied into two outputs). Branching is shown in Diagram 1.8.

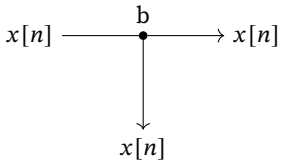


Figure 1.8: Notation for the branching (signal splitting) operation. The b node allows splitting of the signal into two—equal—branches where x is routed.

In order to apply most of the above operations, two signals should possess the same length—in those cases where this does not occur, a *zero-padding* is required. For instance, to perform addition on two signals, the two signals must share the same length, therefore it is mandatory to at least perform a zero-padding on the shortest of the signals. Infinitely long signals can always be summed together.

Multiple elementary operations can be combined as well, by building a directed graph in which the signals are subjected to multiple eventual operations, in the order determined by the particular graph. Usually, multiple operations should be necessary and are therefore applied in special manners to obtain the desired result.

1.3 Ensemble Averaging

A very powerful operation to apply to a signal is the **ensemble averaging**. Ensemble averaging is the operation of performing an average of a signal when multiple, independent

measures are available. The goal of the ensemble averaging is to enhance the quality of a signal with respect to a noise which affects it. In other words, the result is expected to suffer less from noise than the original signal.

Suppose i-th measured signal x_i has two components $x_i = s + d_i$, the first one relative to an original signal s and the noise component d_i, which varies according to the measurement. The ensamble average of the K measurements is the signal x_{avg} such that

$$x_{avg} = \frac{1}{K}\sum_{i=1}^{K} s + d_i = s + \frac{1}{K}\sum_{i=1}^{K} d_i, \tag{1.2}$$

which tends to the original signal s as the number of signals $K \to \infty$ approaches infinity. The benefit of the ensemble averaging lies in the fact that the variance of the white noise d_i is reduced by a factor of K, leading to a reasonable replica of the original signal s. An example of application of ensemble averaging is shown in Figure 1.9.

1.4 Sample Rate Alteration

The operation of alterating the sample rate F_S is much important. Generally, it is adopted to obtain a new signal x' from a signal x, but with a different sample rate F'_S. The new sample rate can either be higher or lower than the original signal, so that a *sampling rate alteration ratio* can be defined,

$$R_S = \frac{F'_S}{F_S}. \tag{1.3}$$

The ratio in equation 1.3 can either be greater or lower than 1, and determines the nature of the sample rate alteration. In particolar, in the case of $R_S > 1$ one performs the **interpolation** operation, while in the opposite case of $R_S < 1$ one gets the **decimation** operation.

Two operations are imperative for, respectively, interpolation and decimation: the *up-sampling* and the *down-sampling* operations. To be specific, in order to perform interpolation a *further* operation is needed other than up-sampling.

Up-Sampling

The up-sampling operation by an integer factor $L > 1$ involves the addition of $L - 1$ zero-valued samples inserted between two consecutive samples of the original signal:

$$x_u[n] = \begin{cases} x[n/L] & n = 0, \pm L, \pm 2L, \dots \\ 0 & otherwise \end{cases} \tag{1.4}$$

Up-sampling operation is denoted by a $\uparrow L$ block, as in Diagram 1.10. Upsampling is usually followed by *interpolation*.

Down-Sampling

The down-sampling operation by an integer factor $M > 1$ means that from the original signal only one sample every M is preserved. This way, $M - 1$ consecutive in-between samples are removed from the signal, while a single one every M samples is preserved,

$$x_d[n] = x[nM] \tag{1.5}$$

Down-sampling operation is denoted by a $\downarrow M$ block, as in Figure 1.11. The result of downsampling is a sequence in which some values are missing, those that are removed by the downsampling operation. A crucial risk of downsampling that we will tackle is the **aliasing** phenomenon.

1.5 Classification of sequences

There are several ways to classify a sequence.

The first one is by **symmetry**. A *conjugate-symmetric* sequence is a sequence that satisfies the property

$$x[n] = x^*[-n];$$

If x is real, then it is said to be an *even sequence*. The *conjugate antisymetric sequence*, on contrary, is such that

$$x[n] = -x^*[-n],$$

and if x is real, then it is said to be an *odd sequence*. Every sequence can be defined with the sum of its conjugate symmetric and antisimetric parts[1], that are two distinct signals

$$x[n] = x_{cs}[n] + x_{ca}[n];$$

[1] In fact, $x_{cs} = \frac{1}{2}(x[n] + x^*[-n])$ and $x_{as} = \frac{1}{2}(x[n] - x^*[-n])$.

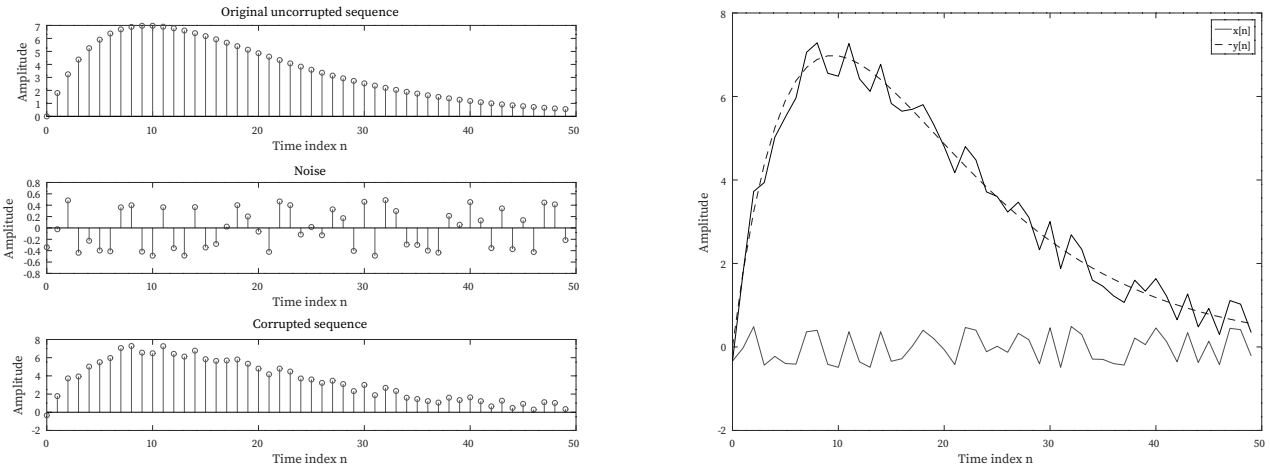


Figure 1.9: The ensemble averaging. On the left, the original uncorrupted sequence $s[n]$, the noise $d[n]$ and the corrupted sequence $x[n]$. On the right, the dashed line represents the sequence $y[n]$, a "recovered" version of the original sequence $s[n]$. Recovering has been performed by means of the ensemble averaging process.

$$x[n] \longrightarrow \boxed{\uparrow L} \longrightarrow x_u[n]$$

Figure 1.10: Notation for the up-sampling operation by a factor L.

$$x[n] \longrightarrow \boxed{\downarrow M} \longrightarrow x_u[n]$$

Figure 1.11: Notation for the down-sampling operation by a factor M.

Another way is by **periodicity**: a sequence $\tilde{x}[n]$ is *periodic with period N* if it satisfies the property

$$\tilde{x}[n] = \tilde{x}[n + kN]$$

for some $N \in \mathbb{Z}$. Smallest N value that satisfies such property is called *fundamental period*. A sequence that does not satisfy this property is said to be *aperiodic*. The sum of two periodic signals x_a and x_b of fundamental period—respectively—N_a and N_b is still a periodic sequence, having fundamental period N_f of

$$N_f = \frac{N_a N_b}{\mathrm{GCD}(N_a, N_b)} \geq \max(N_a, N_b),$$

where GCD is the "greatest common divisor" operation;

One can also compute the **total energy** of a signal x, defined as the quantity

$$\mathcal{E}_x = \sum_{n=-\infty}^{\infty} |x[n]|^2. \tag{1.6}$$

A finite-length sequence will possess finite energy, but an infinite-length sequence might have finite or infinite energy depending on the nature of the signal. Among with the definition of energy, it comes the definition of **average power** of a signal, which is, specifically,

$$P_x = \lim_{K\to\infty} \frac{1}{2K+1} \sum_{n=-K}^{K} |x[n]|^2. \tag{1.7}$$

The average power formula may depend on the nature of the signal. In particular,

- regarding *aperiodic* signals, the average power is

  $$P_x = \lim_{K\to\infty} \frac{1}{2K+1} \sum_{n=-K}^{K} |x[n]|^2,$$

  and—recalling the definition of total energy—can thus be rewritten as

  $$P_x = \lim_{K\to\infty} \frac{1}{2K+1} \mathcal{E}_{x,K};$$

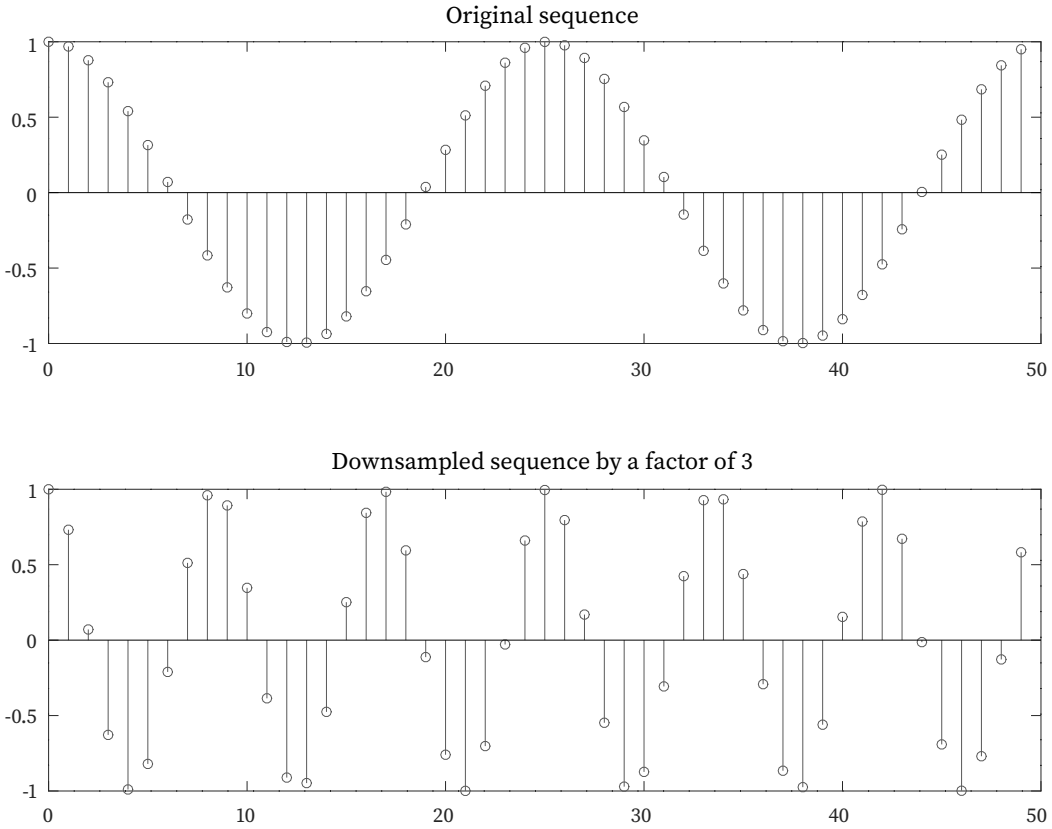


Figure 1.12: Example of down-sampling operation applied to a sinusoidal input sequence.

- regarding *periodic* signals with period N, the average power is

$$P_x = \frac{1}{N}\sum_{n=0}^{N-1} |x[n]|^2,$$

a quantity that is computed through a period and might be finite or infinite[2]. A periodic sequence that possesses an infinite energy—but finite average power—is said to be a *power signal*. Instead, a finite energy signal with infinite length possessing zero average power is said to be an *energy signal*. Energy signals and power signals are peculiar sequences that satisfy some specific properties.

A sequence $x[n]$ is said to be **bounded** if

$$|x[n]| \leq B_x < \infty,$$

when the greatest value—in absolute value—of the sequence is smaller than a certain *bound* B_x of the sequence x. The sequence $x[n] = \cos 0.3\pi n$ is a bounded sequence since

$$|x[n]| = |\cos 0.3\pi n| \leq 1.$$

[2]To introduce some examples, consider a causal sequence defined by

$$x[n] = \begin{cases} 3(-1)^n, & n \geq 0 \\ 0, & n < 0 \end{cases}.$$

sequence x has infinite energy. The average power is given by

$$P_x = \lim_{K\to\infty} \frac{1}{2K+1}\left(9\sum_{n=0}^{K} 1\right) = \lim_{K\to\infty} \frac{9(K+1)}{2K+1} = 4.5,$$

a finite quantity, of course.

A sequence is **absolutely summable** if the sum of all its absolute values for all of its samples samples is a limited quantity[3] An absolutely summable sequence is a sequence such that

$$\sum_{n=-\infty}^{\infty} |x[n]| < \infty. \tag{1.8}$$

As we will see later, absolute summability implies that the Fourier transform converges uniformly, so this means that the unit circle is contained in the region of convergence (ROC).

At last, a sequence $x[n]$ is **square-summable** if

$$\sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty,$$

in a similar fashion to the absolutely summable case. sequences can be absolutely summable but not square-summable, and the contrary can occur as well. For instance, if one considers the sequence $k[n] = \frac{\sin 0.4n}{\pi n}$ will find out that it is square-summable but **not** absolutely summable.

1.6 Basic sequences

Basic sequences are the fundamental sequences that will be encountered throughout the study of signals. There are several kinds of notable basic sequences. All of them possess peculiar properties which are paramount to the discipline of signal theory.

The first one to be mentioned is the **unit sample sequence**, also called the **impulse sequence**. It is a special signal whose only sample different from zero is the sample in the origin:

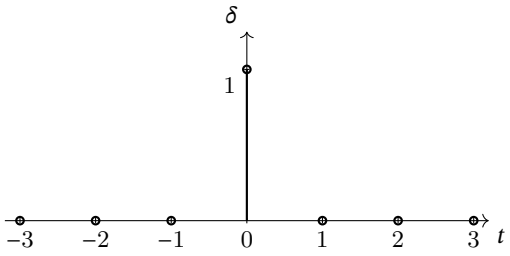
$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}.$$

[3] For example, sequence

$$y[n] = \begin{cases} 0.3^n, & n \geq 0 \\ 0, & n < 0 \end{cases}$$

is absolutely summable since

$$\sum_{n=0}^{\infty} |0.3^n| = \frac{1}{1-0.3} < \infty.$$



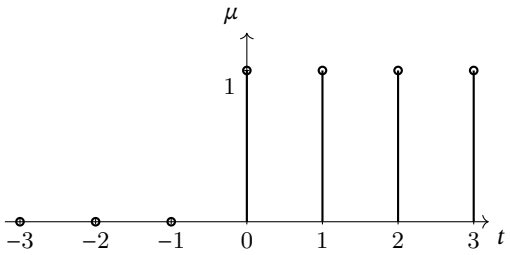
In practice, the "delta impulse" can be used to show a lot of interesting properties, and it is indeed the kernel of the convolution filter.

Let's continue with the second basic sequence.

Defined as

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

the **unit step sequence** is the integral of the impulse sequence. Indeed, when summing up all the values of the unit impulse sequence one has only 0s until the only non-zero value occurs and the value of the integral collapses to the unit 1.



A **real sinusoidal sequence** is a sequence defined as

$$x[n] = A\cos(\omega_0 n + \phi),$$

where A is the *amplitude* of the sinusoid, ω_0 is the *angular frequency* and ϕ is the *phase* of $x[n]$. A real sinusoid sequence therefore possesses amplitude, (angular) frequency and phase as intrinsic properties. Sinusoids fulfill an important role as they are signals whose amplitude and frequency stays the same throughout the time and are describable with sine and cosine "elementary" functions.

An indeed important family of sequences are the **exponential sequences**, all of them having form

$$x[n] = A\alpha^n, -\infty < n < \infty,$$

where $A, \alpha \in \mathbb{C}$ can be either real or complex numbers. The various kinds of exponentials arise from peculiar choices of coefficient and basis, as the behavior will dramatically change depending on the nature of both.

In general, an exponential sequence can be expressed expanding modulus and phase of both numbers, obtaining

$$x[n] = |A|e^{j\phi}e^{(\sigma_0+j\omega_0)n} = x_{re}[n] + jx_{im}[n]$$

a partitioning into real and imaginary part of the sequence. In particular, one has

$$x_{re}[n] = |A|e^{\sigma_0 n}\cos(\omega_0 n + \phi), \tag{1.9}$$
$$x_{im}[n] = |A|e^{\sigma_0 n}\sin(\omega_0 n + \phi). \tag{1.10}$$

Following this idea, the analysis of both terms indicate two different behaviors depending on the nature of A and α. Suppose $A \in \mathbb{C}$ and $\alpha \in \mathbb{C}$. The first and second terms—related to the real and imaginary parts of the exponential sequence—are two real, sinusoidal sequences with constant ($\sigma_0 = 0$), growing ($\sigma_0 > 0$) or decaying ($\sigma_0 < 0$) amplitudes for samples $n > 0$. In practice, they are sinusoids whose angular frequency and phase are determined, respectively, by terms ω_0 and ϕ, and whose amplitude varies according to the term $e^{\sigma_0 n}$ with the exception of a multiplication constant $|A|$, the modulus of the complex number $A \in \mathbb{C}$. Still, their value at zero will be different as they are described by either sine or cosine, but the overall behavior will be the same: either decaying, or growing, or they are sinusoids of constant amplitude.

Let now $A, \alpha \in \mathbb{R}$. Real exponential sequences are the "classic" exponential sequences, described by

$$x[n] = A\alpha^n$$

and possibly constant within the special case $\alpha = 1$. In practice,

- for values $\alpha < 1$, the exponential will decay;
- for values $\alpha > 1$, the exponential will grow over time.

Both sinusoidal sequences and complex exponential sequences are periodic sequences of period N if

$$\omega_0 N = 2\pi r \tag{1.11}$$

holds, with the very same parameters as in the above models and with $r, N \in \mathbb{N}$. As it goes for general periodic signals, the smallest value of N that satisfies the 1.11 is the *fundamental period* of the sequence.

Proof. In order to verify this, two very similar sequences must be picked—the same sequence but original and shifted of N samples:

$$x_1[n] = \cos(\omega_0 n + \phi), \tag{1.12}$$
$$x_2[n] = \cos(\omega_0 (n+N) + \phi). \tag{1.13}$$

Now, the second sequence can be soon rewritten as

$$x_2[n] = \cos(\omega_0 (n+N) + \phi), \tag{1.14}$$
$$= \cos(\omega_0 n + \phi)\cos\omega_0 N - \sin(\omega_0 n + \phi)\sin\omega_0 N. \tag{1.15}$$

The 1.15 will be equal to $x_1[n]$ if and only if

$$\sin\omega_0 N = 0 \wedge \cos\omega_0 N = 1, \tag{1.16}$$

two conditions that are met if and only if the 1.11 is satisfied. Alternatively, the very same equation—but rearranged—must be satisfied,

$$\frac{2\pi}{\omega_0} = \frac{N}{r}. \tag{1.17}$$

■

Along with this proof, one might say that N is the *true* period, while the quantity $\frac{N}{r}$ is the *apparent* period—the one of the implied continuous function. In fact, if $\frac{2\pi}{\omega_0}$ is a non-integer rational number, then the period will be a multiple of $\frac{2\pi}{\omega_0}$, which basically means $r > 1$. Otherwise, in the case of an irrational $\frac{2\pi}{\omega_0}$ the discrete sequence will be aperiodic.

For instance, if one considers

$$x[n] = \sin(\sqrt{3}n + \phi),$$

this (discrete) sequence will be aperiodic regardless of the fact that the continuous function would still be a periodic signal. In fact, substituting in 1.17 yields

$$\frac{2\pi}{\sqrt{3}} = \frac{N}{r},$$

resulting in a quantity on the left that can never be rational as the right member of the equation. This is because the 1.17 strictly requires that the angular frequency is an irrational number—a multiple of π, to be accurate—in order to fulfill the condition of rationality.

1.6.1 Frequency properties of basic sinusoidal sequences

Let's now focus on the peculiar properties of sinusoidal and exponential basis sequences.

To begin with, consider sequences $x[n] = \exp j\omega_1 n$ and $y[n] = \exp j\omega_2 n$, with $0 \leq \omega_1 < \pi$ and $2\pi k \leq \omega_2 < \pi(2k+1)$, with k positive integer. If $\omega_2 = \omega_1 + 2\pi k$, then $x[n] \equiv y[n]$: sequences x and y are *indistinguishable* from each other. This means that for (complex) exponential sequences a shift of $2\pi k$ leads to *the very same* sequence, as the resulting shifted sequence will be exactly the same as the original one.

Another paramount property revolves around the oscillation frequency of a sinusoid $x[n] = \cos(\omega_0 n)$. In practice, the frequency of oscillation increases as ω_0 goes from 0 to π, while the contrary happens when moving from π to 2π. What does this mean?

That is a property of *discrete* sinusoids: the frequency of oscillation is the greatest around $\omega = \pi$, while it is the lowest possible for frequencies close to 0 or close to 2π. The first frequencies are said to be **high frequencies**, while the latter ones are the **low frequencies**.

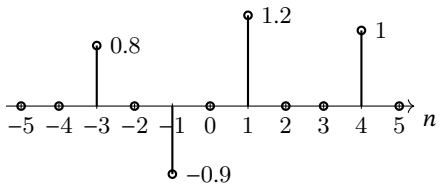
For the first considered property, a sinusoidal sequence having frequency ω_0 in the neighborhood of $\omega = 2\pi k$ is indistinguishable from another sinusoidal sequence whose frequency is $\omega_0 - 2\pi k$, in the neighborhood of $\omega = 0$. Along with the same reasoning, any frequency ω_0 in the neighborhood of $\omega = \pi(2k+1)$ is indistinguishable from a frequency $\omega_0 - \pi(2k+1)$ in the neighborhood of $\omega = \pi$.

Actually, what happens is that the oscillation shows a *periodic* behavior across the frequency—hence, it is possible to describe the entire frequency space with the sole range of values $[0, \pi]$, as the other half $[\pi, 2\pi]$ is completely symmetric to the first range.

As previously said, frequencies in neighborhood of $\omega = 2\pi k$ are the low frequencies, in contrast to those in the neighborhood of $\omega = \pi(2k+1)$ that are the high frequencies[4].

1.6.2 Representation of a sequence with basic sequences

Any arbitrary sequence can be represented in the time domain as a *weighted sum* of some basic sequences and its delayed–advanced versions.



$$x[n] = 0.8\delta[n+3] - 0.9\delta[n+1] + 1.2\delta[n-1] + \delta[n-4]$$

In the above example, the sequence $x[n]$ is represented by a weighted sum of δ impulses $A\delta[n-\tau]$ with amplitude A having the same value of the original sequence at the sample with position $-\tau$ from the origin.

1.7 The sampling process

Usually, when dealing with discrete-time sequences, one encounters examples of signals that are obtained by uniformly sampling a countinuous-time signal. In such scenario, one has a sequence $x[n]$ extracted from a signal $x_a(t)$ by means of a sampling process.

The relationship between the original continuous-time signal and the discrete-time sequence is the following one,

$$x[n] = x_a(t)\Big|_{t=nT} = x_a(nT), n = \ldots, -2, -1, 0, 1, 2, \ldots$$

The above relationship between the original signal and the sampled sequence states that the sampling process is equivalent to simply pick the value of the countinous-time signal x_a at *some* samples nT. By performing such operation, one will obtain a sampled sequence with sampling period TT.

[4] The sequence

$$v_1[n] = \cos(0.1\pi n) \equiv \cos(1.9\pi n)$$

is a low frequency signal, while sequence

$$v_2[n] = \cos(0.8\pi n) \equiv \cos(1.2\pi n)$$

is a high frequency one.

The time variable t of the continuous-time signal $x_a(t)$ is related to the time variable n of $x[n]$ only in some discrete-time instants t_n that must abide to the following rule,

$$t_n = nT = \frac{n}{F_T} = \frac{2\pi n}{\Omega_T}, \tag{1.18}$$

with $F_T = \frac{1}{T}$ denoting the sampling frequency, as expected. Indeed, $\Omega_T = 2\pi F_T$ is the *sampling angular frequency*, in a constant multipliation-factor relationship between the sampling frequency.

Let's now consider the continuous-time sinusoidal signal

$$x_a(t) = A\cos(2\pi f_0 t + \phi) = A\cos(\Omega_0 t + \phi)$$

and perform a sampling process with sampling period T. The resulting sequence will be

$$x[n] = A\cos(\Omega_0 nT + \phi) = A\cos(\underbrace{\frac{2\pi\Omega_0}{\Omega_T}}_{\omega_0} n + \phi), \tag{1.19}$$

$$= A\cos(\omega_0 n + \phi), \tag{1.20}$$

where the angular frequency $\omega_0 := 2\pi\frac{\Omega_0}{\Omega_T} = \Omega_0 T$ is the *normalized digital angular frequency* of the sequence $x[n]$. Notice that an analog function generates *different* sampled sequences according to the choice of the sampling period T. Indeed, *different* sine functions can generate *the same* sampled sequence if a peculiar T^* is selected.

Measurement units

Measurements units are the following ones,

- T measures in seconds;
- ω_0 in radians/sample;
- Ω_0 in radians/second;
- analog frequency f_0 measures in hertz (Hz).

1.7.1 Aliasing

Let's pick three examples of continuous-time signals,

$$g_1(t) = \cos(6\pi t),$$
$$g_2(t) = \cos(14\pi t),$$
$$g_3(t) = \cos(26\pi t),$$

of frequencies—respectively—of 3Hz, 7Hz and 13Hz. Sampling the three signals at a sampling rate of 10Hz, with $T = 0.1s$, three sequences are soon generated,

$$g_1[n] = \cos(0.6\pi n),$$
$$g_2[n] = \cos(1.4\pi n),$$
$$g_3[n] = \cos(2.6\pi n),$$

as $f_0 \cdot T = \frac{1}{10} f_0$.

Now, since

$$g_1[n] = \cos(0.6\pi n) =$$
$$= \cos((2\pi - 0.6\pi)n) = \cos(1.4\pi n) = g_2[n],$$
$$g_1[n] = \cos(0.6\pi n) =$$
$$= \cos((2\pi + 0.6\pi)n) = \cos(2.6\pi n) = g_3[n],$$

all three sequences result to be identical. Therefore, it is rather difficult—let alone possible—to associate a unique continuous-time function with each of these sequences.

The phenomenon for which a continuous-time signal of higher frequency acquires the identity of a sinusoidal sequence of lower frequency after sampling is called **aliasing**. Since there are an infinite number of continuous-time signals that can lead to the same sequence when sampled periodically, additional conditions are needed so that the sequence $\{x[n]\} = \{x_a(nT)\}$ can uniquely represent the parent continuous-time signal $x_a(t)$. Indeed, in the above case $x_a(t)$ can still be fully recovered from the sequence $\{x[n]\}$. The process of bringing the angular frequency to a value between 0 and π is the *normalization of the angular frequency*, a step that should always be performed as it is in the best interests to uniquely represent a specific class of sinusoids with a normalized representation.

Of course, there are many examples of sequences from which signals might be fully recovered. Suppose one wants to obtain the following discrete-time signal $v[n]$ from a continuous-time signal of shape

$$v_a(t) = 6\cos 60\pi t + 3\sin 300\pi t + 2\cos 340\pi t$$
$$+ 4\cos 500\pi t + 10\sin 660\pi t$$

with sampling period $T = \frac{1}{200} = 0.005s$. To do so, one should perform the following algebraic transformations,

$$\begin{aligned}
v[n] &= 6\cos 0.3\pi n + 3\sin 1.5\pi n + 2\cos 1.7\pi n \\
&\quad + 4\cos 2.5\pi n + 10\sin 3.3\pi n, \\
&= 6\cos 0.3\pi n + 3\sin(2\pi - 0.5\pi)n + 2\cos(2\pi - 0.3\pi)n \\
&\quad + 4\cos(2\pi + 0.5\pi)n + 10\sin(4\pi - 0.7\pi)n, \\
&= 6\cos 0.3\pi n - 3\sin 0.5\pi n + 2\cos 0.3\pi n \\
&\quad + 4\cos 0.5\pi n - 10\sin 0.7\pi n, \\
v[n] &= 8\cos 0.3\pi n + 5\cos(0.5\pi n + 0.6435) - 10\sin 0.7\pi n.
\end{aligned}$$

In the end, $v[n]$ is composed of 3 discrete-time sinusoidal signals of normalized angular frequencies 0.3π, 0.5π and 0.7π.

However, that's not the only signal possible that is capable of generating that very same sequence. Veritably, also the two following signals $w_a(t)$ and $g_a(t)$

$$\begin{aligned}
w_a(t) &= 8\cos 60\pi t + 5\cos(100\pi t + 0.6435) - 10\sin 140\pi t \\
g_a(t) &= 2\cos 60\pi t + 4\cos 100\pi t + 10\sin 260\pi t \\
&\quad + 6\cos 460\pi t + 3\sin 700\pi t
\end{aligned}$$

will produce the very same sequence $v[n]$.

To avoid aliasing, an *anti-aliasing* strategy should be adopted. As **Nyquist Sampling Theorem** dictates, since $\omega_0 = \frac{2\pi\Omega_0}{\Omega_T}$, if $\Omega_T > 2\Omega_0$ the corresponding normalized digital angular freqyency ω_0 of the discrete-time signal obtained by sampling the parent continuous-time sinusoidal signal will be in the range $-\pi < \omega < \pi$, and no aliasing phenomena would occur.

In practice, it is sufficient to select a sampling angular frequency Ω_T that is *at least the double* of the Ω_0 angular frequency of the sinusoid. On contrary, if $\Omega_T < 2\Omega_0$ the normalized digital angular frequency will fall in the range $-\pi < \omega < \pi$, then it will fold over a lower digital frequency $\omega_0 = \langle\frac{2\pi\Omega_0}{\Omega_T}\rangle_{2\pi}$ because of the aliasing phenomenon. This will inevitably lead to artifacts, as the sequence can be interpreted as different sinusoids other than the original one.

Generalization

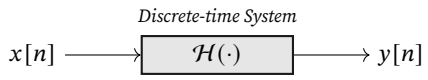
Let's consider an arbitrary continuous-time signal $x_a(t)$, composed of a weighted sum of an arbitrary, finite number of sinusoidal signals. The signal might be uniquely represented—completely avoiding the aliasing effect—by a sequence $\{x[n]\}$, a sampled version of the original signal $x_a(t)$ with sampling frequency Ω_T that should be *greater than the double of* the highest frequency in the signal $x_a(t)$. For instance, suppose x_a contains a sinusoid whose frequency f_h is the highest of all sinusoids in x_a; in order to avoid the aliasing effect, a correct choice for the angular sampling frequency would be $\Omega_T > 2\pi f_h$.

Chapter 2

Discrete-time Systems

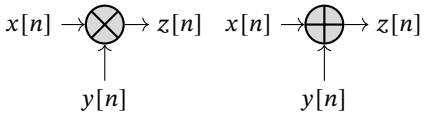
Discrete time views values of variables as occurring at distinct, separate "points in time", or equivalently as being unchanged throughout each non-zero region of time ("time period")—that is, *time is viewed as a discrete variable* rather than as a continuous one. Thus a non-time variable jumps from one value to another as time moves from one time period to the next. This view of time corresponds to a digital clock that gives a fixed reading of 10:37 for a while, and then jumps to a new fixed reading of 10:38, etc. In this framework, each variable of interest is measured once at each time period. The number of measurements between any two time periods is finite. Measurements are typically made at sequential integer values of the variable "time" [2].

A **discrete-time system** is a system that processes a given *input sequence* $x[n]$ to generate an *output sequence* $y[n]$ with certain characteristics derived from both the input sequence and the system. Typically, one has a single-input, single-output discrete-time system, whose general diagram is of the following kind,



Sure thing, the discrete-time system is characterized by an operator $\mathcal{H}(\cdot)$ that transforms the input sequence x into another sequence y, the output. Resulting output will possess some characteristics that should be desirable and depend on the requirements.

We have already seen some examples of discrete-time systems. For instance, two of these were the *modulator* and the *adder* discrete-time systems, both 2-input, 1-output systems.



Other 1-input, 1-output discrete-time systems were the *multiplier*, the *unit delay* and the *unit advance* we previously encountered in Section 1.2.1.

In addition, multiple elementary operators may be combined together to form a much-more-complex discrete-time system. As an example, consider the one-input one-output discrete-time system in Figure 2.1.

2.1 Fundamental Discrete-time Systems

2.1.1 Accumulator

The first fundamental discrete-time system we will meet is the **accumulator** system. An accumulator is led by the following relationship

$$y[n] = \sum_{i=-\infty}^{n} x[i] \tag{2.1}$$

$$= \sum_{i=-\infty}^{n-1} x[i] + x[n] \tag{2.2}$$

$$= y[n-1] + x[n]. \tag{2.3}$$

which establishes that the output $y[n]$ at the time instant n is the sum of two terms, the first one being the input at the time instant n, $x[n]$, and the second one being the previous output $y[n-1]$ at the previous time instant $n-1$. The latter addendum is the sum of all previous input sample values, from the beginning of the signal to the previous time instant—that is, from sample $-\infty$ to sample $n-1$.

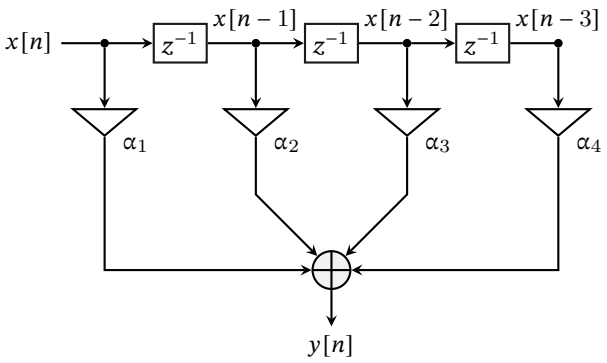


Figure 2.1: A one-input, one-output discrete-time system that shows concatenation of multiple elementary operators to form a more complex graph and a more convoluted input–output relationship.

An accumulator, as its name suggests, *cumulatively adds* all input sample values from the beginning of the signal history up to the current time instant n. This behavior is indeed evident by looking at the terms in the sides of Equation 2.3.

Any accumulator could be split into two terms, a first being a constant, and the second one capable of describing a *causal* input sequence. In truth,

$$y[n] = \sum_{i=-\infty}^{n} x[i] \tag{2.4}$$

$$= \sum_{i=-\infty}^{-1} x[i] + \sum_{i=0}^{n} x[i] \tag{2.5}$$

$$= y[-1] + \sum_{i=0}^{n} x[n]. \tag{2.6}$$

where the first term $y[-1]$ is said to be *initial condition*, while the second term is fit to describe a *causal* input sequence, as it is a sum starting from time instant 0. Basically, this means that *any accumulator can be described with a causal input sequence and with its initial conditions,* as 2.6 suggests. This property reveals to be of paramount importance when describing accumulators in general, especially if one wants to build a discrete-time system that only takes in account causal input signals, starting from time interval 0 and going on.

2.1.2 M-point Moving-Average System

The M-**point Moving Average System** is a discrete-time system described by the input–output relationship

$$y[n] = \frac{1}{M} \sum_{k=0}^{M-1} x[n-k]. \tag{2.7}$$

The M-point attribute is related to the fact that—starting from time instant 0—exactly M points are averaged together, while "moving" stands for the fact that the value of the system can be interpreted as a constantly-moving average between points, since the output value will vary as new input samples are added to the sum.

In most applications, the input data $x[n]$ is a bounded sequence; therefore, the M-point average $y[n]$ is a bounded sequence as well. Doubtlessly, if there is no bias in the measurements, an improved estimate of the noisy data is obtained by simply increasing the value of M.

A direct implementation of the moving average discrete-time system requires $M-1$ additions, a single division and the storage of $M-1$ past input data samples. As we will see later on, this is far from the most efficient solution.

The M-point moving average system can be rewritten as follows,

$$\begin{aligned}
y[n] &= \frac{1}{M}\left(\sum_{l=0}^{M-1} x[n-l]\right) \\
&= \frac{1}{M}\left(\sum_{l=1}^{M-1} x[n-l] + x[n]\right) \\
&= \frac{1}{M}\left(\sum_{l=1}^{M-1} x[n-l] + x[n] + x[n-M] - x[n-M]\right) \\
&= \frac{1}{M}\left(\sum_{l=1}^{M} x[n-l] + x[n] - x[n-M]\right) \\
&= \frac{1}{M}\left(\sum_{l=0}^{M-1} x[n-1-l] + x[n] - x[n-M]\right) \\
&= \frac{1}{M}\left(\sum_{l=0}^{M-1} x[n-l-1]\right) + \frac{1}{M}\left(x[n] - x[n-M]\right) \\
y[n] &= y[n-1] + \frac{1}{M}\left(x[n] - x[n-M]\right).
\end{aligned}$$

We have just witnessed how the computation of the modified M-point moving average system—rearranging the *recursive equation*—requires *only* 2 *additions and* 1 *division,* in contrast with the previous case where a number of operations dependent on the number of samples M was necessary. An apparently unvalicable computation limit had been removed just by cleverly rearranging the equation of the system involved. Later on, we will see other cases in which this is true and very desirable.

An application for the M-point moving average system is the following one: let a sequence $x[n]$ be the sum of two sequences, the first one being a sequence $s[n]$ obtained from a signal which is inevitably corrupted by a noise $d[n]$, hence

$$x[n] = s[n] + d[n].$$

Usually, the noise is a small part of the signal; still, the original signal is somehow corrupted by the presence of the noise in such a way that sometimes noise reduction techniques are auspicable. A common §FIXME

2.1.3 Exponentially Weighted Running Average Filter

The **Exponentially Weighted Running Average Filter** is a kind of filter expressed by the relationship

$$y[n] = \alpha y[n-1] + x[n], 0 < \alpha < 1. \tag{2.8}$$

Since $0 < \alpha < 1$, the exponentially weighted average filter places a lot more emphasis on *current* data samples than *past* data samples, as all the previous preocessed data is multiplied by a constant factor which is smaller than 1. If $x[n] \equiv 0$ at some time interval and for all future samples, the output $y[n]$ is condemned to decay and collapse to 0 at a speed that substantially depends on the value of α.

In fact, this is soon shown by just substituting recursively the definition with itself,

$$\begin{aligned}
y[n] &= \alpha y[n-1] + x[n] \\
&= \alpha(\alpha y[n-2] + x[n-1]) + x[n] \\
&= \alpha^2(\alpha y[n-3] + x[n-2]) + \alpha x[n-1] + x[n] \\
&= \alpha^3 y[n-3] + \alpha^2 x[n-2] + \alpha x[n-1] + x[n]
\end{aligned}$$

where it is immediately clear that the terms with α of higher order tend to be smaller as the order increases.

Computation of the running average filter only requires a single addition and a single multiplication; still, it requires storage of the previously-run average value. As a bonus, it does not require the storage of the full-chain of past data samples.

2.1.4 Median Filter

The **median filter** of a set of $2K + 1$ numbers is the number such that K numbers from the set possess values that are greater than this number, while the other K numbers have lower values. The median can be determined by performing a rank-ordering of the numbers in the set by their values, and ultimately choosing the number at the middle of the just produced sequence. In short, the median filter computes *the median in the window*.

A median filter is usually implemented by a sliding window of *odd* length, over the entire input sequence $x[n]$, moving the window from left to right one sample at a time. The

output $y[n]$ at instant n is the median value of the samples inside the sliding window that is currently centered at n.

Median filters are very good at removing both *shot noise* (impulse-like noise) and *random noise*, either of additive or substitutive kind—especially in the case of sudden, large errors in the corrupted signal. For this reason, median filters are applied to smooth a shot-noise-corrupted signal.

2.2 Linear interpolation

The **linear interpolation** is a technique that is adopted to estimate sample values between pairs of adjacent sample values of a discrete-time sequence. To perform that, additional samples are added in-between two adjacent values (upsampling), and then a further interpolation technique is adopted by means of *linearly connecting the values*.

The linear interpolation can be expressed as in Figure 2.2

Linear interpolation can never produce a correct output sequence. This is because—if the real signal were the dashed line in 2.2—a signal whose derivative is infinite at some points would require *infinite* band, therefore an infinite sampling rate would be necessary.

Mathematically, interpolation having factors of 2 and 3 might be expressed, respectively for sequences $y_2[n]$ and $y_3[n]$, as follows,

$$\begin{aligned} y_2[n] &= x_u[n] + \frac{1}{2}(x_u[n-1] + x_u[n+1]); \\ y_3[n] &= x_u[n] + \frac{1}{3}(x_u[n-2] + x_u[n+2]) \\ &\quad + \frac{2}{3}(x_u[n-1] + x_u[n+1]), \end{aligned}$$

with the *low-frequency gain* equal to DCGain $= L$. Notice that a formally correct solution would require much more complex operators.

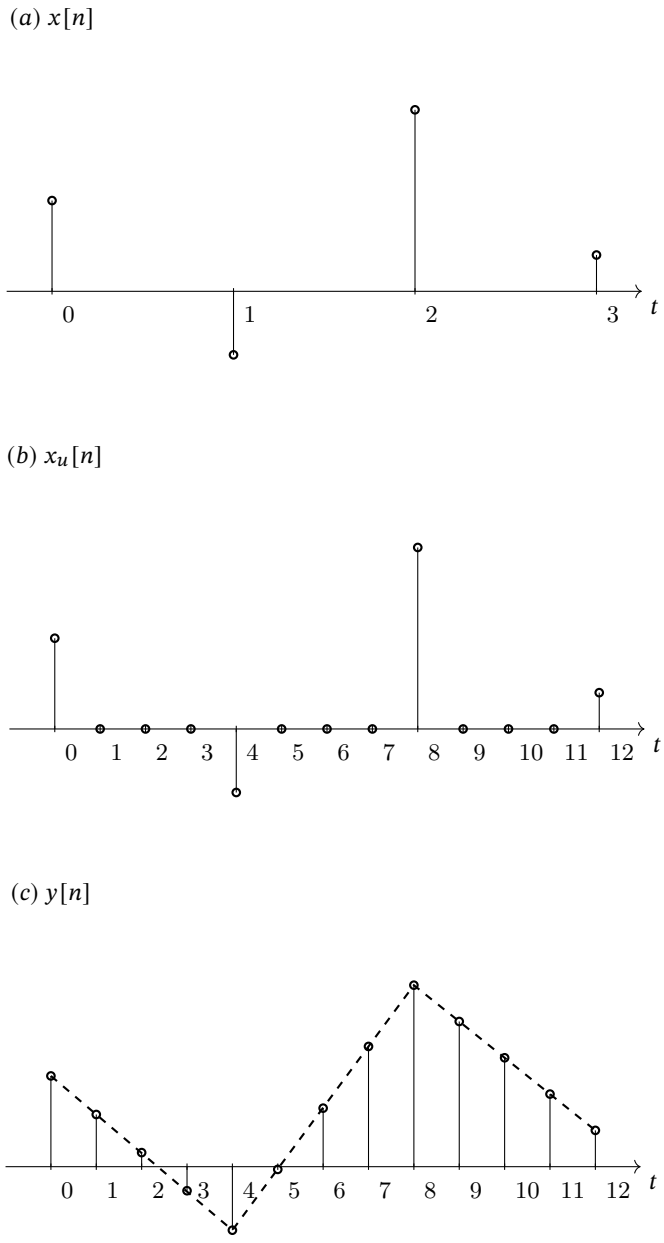


Figure 2.2: Interpolation of a signal $x[n]$ with factor of 4. From top to bottom: (a) original signal $x[n]$, (b) upsampled version $x_u[n]$ and (c) interpolated output sequence $y[n]$.

2.3 Classification of Discrete-time Systems

2.3.1 Linear Discrete-time Systems

A **Linear Discrete-time System** is any special kind of discrete-time system such that, if y_1 is the output of the input x_1 and y_2 is the output from the input x_2, then f or an input that is the combination of the two inputs x_1 and x_2

$$x[n] = \alpha x_1[n] + \beta x_2[n],$$

the output is given by the *superposition* of outputs

$$y[n] = \alpha y_1[n] + \beta y_2[n]. \tag{2.9}$$

The above property is fundamental for any linear discrete-time system, and must hold for any arbitrary constant α and β, for all possible combination of input sequences $x_1[n]$ and $x_2[n]$, and it is called the **superposition principle**.

Discrete-time Systems that follow the superposition principle and are thus linear are all the accumulators. Indeed, given an accumulator output to two inputs,

$$y_1[n] = \sum_{l=-\infty}^{n} x_1[l],$$
$$y_2[n] = \sum_{l=-\infty}^{n} x_2[l],$$

for an input supercomposed of $x[n] = \alpha x_1[n] + \beta x_2[n]$ the corresponding output of the accumulator is

$$\begin{aligned} y[n] &= \sum_{l=-\infty}^{n} (\alpha x_1[l] + \beta x_2[l]) \\ &= \alpha \sum_{l=-\infty}^{n} x_1[l] + \beta \sum_{l=-\infty}^{n} x_2[l] \end{aligned}$$

which indeed shows that any accumulator is a *linear* discrete-time system.

Moreover, since they follow the superposition principle, a property analogous of Equation 2.6 holds—that is, given

$$y_1[n] = y_1[-1] + \sum_{l=0}^{n} x_1[l],$$
$$y_2[n] = y_2[-1] + \sum_{l=0}^{n} x_2[l],$$

by the superposition principle one gets

$$\begin{aligned} \alpha y_1[n] + \beta y_2[n] &= \alpha(y_1[-1] + \sum_{l=0}^{n} x_1[l]) + \beta(y_2[-1] + \sum_{l=0}^{n} x_2[l]) \\ &= (\alpha y_1[-1] + \beta y_2[-1]) + (\alpha \sum_{l=0}^{n} x_1[l] + \beta \sum_{l=0}^{n} x_2[l]), \end{aligned}$$

and for this reason $y[n] = \alpha y_1[n] + \beta y_2[n]$ if and only if

$$y[-1] = \alpha y_1[-1] + \beta y_2[-1], \tag{2.10}$$

a severe equation that establishes a precise condition under which the initial condition must undergo. For any accumulator with a causal input to be *linear* the condition expressed in Equation 2.10 must hold for all initial conditions $y[-1], y_1[-1], y_2[-1]$ and for all constants α and β. Still, the condition can never be satisfied—unless the accumulator is *initially at rest*, with zero initial condition. That practically means having

$$y[-1] = \alpha y_1[-1] + \beta y_2[-1] \equiv 0.$$

2.3.2 Nonlinear Discrete-time Systems

A **Nonlinear Discrete-time System** is any system that does not comply with the superposition principle. To name one, the median filter is a nonlinear system. To prove that, consider outputs for the following inputs (median filter with window size of 3),

$$\{x_1[n]\} = \{3,4,5\}, 0 \le n \le 2 \longrightarrow \{y_1[n]\} = \{3,4,4\}, 0 \le n \le 2,$$
$$\{x_2[n]\} = \{2,-1,-1\}, 0 \le n \le 2 \longrightarrow \{y_2[n]\} = \{0,-1,-1\}, 0 \le n \le 2,$$
$$\{x[n]\} = \{x_1[n] + x_2[n]\} \longrightarrow \{y[n]\} = \{3,4,3\}$$

but

$$\{y_1[n] + y_2[n]\} = \{3,3,3\}$$

which is different from $\{y[n]\}$; hence, the median filter is a nonlinear discrete-time system, for which the superposition principle doesn't apply.

Another example of nonlinear discrete-time system is the second form of the accumulator with non-zero initial condition, for which the term $y[-1] \neq 0$.

2.3.3 Shift-Invariant Systems

Any **Shift-Invariant System** is a system for which if the response to an input $x_1[n]$ is $y_1[n]$, then the response to an input

$$x[n] = x_1[n - n_0]$$

where $n_0 \in \mathbb{Z}$ is the translation of the output as well, that is

$$y[n] = y_1[n - n_0].$$

The above property is called the *time-invariance property*.

The above relationship must hold for any arbitrary input and its corresponding input. This means that by simply shifting the input sequence one obtains an output sequence which is shifted as well, by the same quantity as the shift of the input sequence.

Time-invariance property ensures that—for a specified input—the output is independent of the time the input is being applied.

As an example of a **non**-complying system, consider an upsampler with an input-output relationship given by

$$x_u[n] = \begin{cases} x[n/L] & n = 0, \pm L, \pm 2L, \dots \\ 0 & otherwise \end{cases}$$

For a new input $\hat{x}[n] = x[n - n_0], n_0 \in \mathbb{Z}$, one obtains the output

$$\begin{aligned} \hat{x}_u[n] &= \begin{cases} \hat{x}[n/L] & n = 0, \pm L, \pm 2L, \dots \\ 0 & otherwise \end{cases} \\ &= \begin{cases} x[\frac{n - Ln_0}{L}] & n = 0, \pm L, \pm 2L, \dots \\ 0 & otherwise \end{cases} \end{aligned}$$

which is indeed *different* from what one obtains by applying the definition straightforwardly, that is

$$\begin{aligned} \hat{x}_u[n - n_0] &= \begin{cases} \hat{x}[\frac{n - n_0}{L}] & n = n_0, \pm L, n_0 \pm 2L, \dots \\ 0 & otherwise \end{cases} \\ &\neq \hat{x}_u[n] \end{aligned}$$

as time instant on the numerator is shifted by n_0 and not by Ln_0. When a discrete-time system is not shift-invariant, it is said to be a *time-varying system*.

2.3.4 Linear Time-Invariant Systems

A much important class of discrete-time systems is the class of the **Linear Time-Invariant (LTI) Systems**. An LTI system is any system that it is both linear and shift-invariant—that is a system satisfying both the superposition principle and the time-invariance property. LTI systems are mathematically easy to deal with, to analyze and to characterize. As a consequence, they are very easy to design compared to other "harder" signals. Indeed, highly useful signal processing algorithms have been developed by means of this class of systems over the last several decades.

2.3.5 Causal Systems

In a **Causal System**, the n_0-th output sample $y[n_0]$ depends only on input samples $x[n]$ for $n \leq n_0$; this means that the output depends only on the input from time instants *before* the current one (n_0), and does not depend on future input samples $n \geq n_0$.

Let $y_1[n]$ and $y_2[n]$ be the responses of a causal, discrete-time system to the inputs $x_1[n]$ and $x_2[n]$, respectively. Then,

$$x_1[n] = x_2[n], n < N \Longrightarrow y_1[n] = y_2[n], n < N. \qquad (2.11)$$

Equation 2.11 implies that for a causal system all changes in output samples never precede changes in the input samples. Examples of causal systems are the signals $y_{c,1}$, $y_{c,2}$, and $y_{c,3}$,

$$\begin{aligned} y_{c,1}[n] &= \alpha_1 x[n] + \alpha_2 x[n-1] \\ &\quad + \alpha_3 x[n-2] + \alpha_4 x[n-3] \\ y_{c,2}[n] &= \beta_0 x[n] + \beta_1 x[n-1] + \beta_2 x[n-2] \\ &\quad + \alpha_1 y[n-1] + \alpha_2 y[n-2] \\ y_{c,3}[n] &= y[n-1] + x[n], \end{aligned}$$

while a noncausal system is, for example, the sequence of a linear interpolation $y_{nc}[n]$,

$$y_{nc}[n] = x_u[n] + \frac{1}{3}(x_u[n-2] + x_u[n+2]) + \frac{2}{3}(x_u[n-1] + x_u[n+1]).$$

Any noncausal system can still be implemented as a causal system; this is done by delaying the output by an appropriate number of samples. For instance, a causal implementation of the factor-of-two interpolator—an otherwise noncausal system—is given by

$$y[n] = x_u[n-1] + \frac{1}{2}(x_u[n-2] + x_u[n]). \tag{2.12}$$

The trick here is to delay of a single unit the entire filter, so that the time instant $n+1$ is simply shifted to n in order to produce a causal system. Since a noncausal system has a finite N such that $n + N > n$, it will always be possible to time-delay of N the entire noncausal system so that it fits the property of causality so that $n + N$ becomes n.

2.3.6 Stable Systems

Since there are various definition of stability, one has to first choose an appropriate one before giving the definition of stable systems. We consider the **Bounded-Input, Bounded-Output (BIBO) stability**.

If $y[n]$ is the response to an input $x[n]$, and if

$$|x[n]| \leq B_x, \forall n,$$

then the system is *BIBO stable* if

$$|y[n]| \leq B_y, \forall n.$$

An example of a BIBO stable system is soon provided. Consider an M-point moving average filter as in Equation 2.7,

$$y[n] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-k].$$

Against a bounded input $|x[n]| \leq B_x < \infty$ one obtains

$$\begin{aligned} |y[n]| &= \left|\frac{1}{M}\sum_{k=0}^{M-1} x[n-k]\right| \\ &= \frac{1}{M}\sum_{k=0}^{M-1} |x[n-k]| \\ &= \frac{1}{M}(MB_x) = B_x. \end{aligned}$$

BIBO stable systems are indeed important, as they are all systems for which—provided the input is known to be bounded—the output will be necessarily bounded.

2.3.7 Passive and Lossless Systems

Discrete-time systems are defined to be **passive** if, for every finite-energy input $x[n]$ the output $y[n]$ has—at most—the very same energy, that is

$$\mathcal{E}_y = \sum_{n=-\infty}^{\infty} |y[n]|^2 \leq \mathcal{E}_x = \sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty \tag{2.13}$$

Any passive system *can never increase the energy* of the input sequence.

A particular case of discrete-time systems are the **lossless** systems, for which the output $y[n]$ possesses the very same energy—there is no energy loss through the system. This means that for a lossless system it goes that

$$\mathcal{E}_y = \sum_{n=-\infty}^{\infty} |y[n]|^2 = \mathcal{E}_x = \sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty \tag{2.14}$$

Suppose to have the following discrete-time system defined by the relationship

$$y[n] = \alpha x[n-N], N \in \mathbb{N}.$$

The output energy for such a system is given by

$$\sum_{n=-\infty}^{\infty} |y[n]|^2 = |\alpha|^2 \sum_{n=-\infty}^{\infty} |x[n]|^2,$$

and it depends on the choice of the coefficient α, as it is either a passive system if $|\alpha| < 1$, or a lossless system in the case of $|\alpha| = 1$. Of course, it is neither passive nor lossless if $|\alpha| > 1$.

2.3.8 Impulse and Step Response of a Discrete-time System

The **unit sample response** or the **impulse response** of a discrete-time system is the response of such system to a unit sample sequence $\{\delta[n]\}$ and it is usually denoted as $\{h[n]\}$.

Correspondingly, the **unit step response** or the **step response** of a discrete-time system is the response of such system to a unit step sequence $\{\mu[n]\}$ and it is commonly denoted by $\{s[n]\}$.

Both responses play a crucial role when studying the properties of discrete-time systems.

Suppose one has the following system

$$\begin{aligned} y[n] &= \alpha_1 x[n] + \alpha_2 x[n-1] \\ &\quad + \alpha_3 x[n-2] + \alpha_4 x[n-3]; \end{aligned}$$

the impulse response is soon obtained by setting $x[n] = \delta[n]$ as input sequence and evaluating the result:

$$\begin{aligned} h[n] &= \alpha_1 \delta[n] + \alpha_2 \delta[n-1] \\ &\quad + \alpha_3 \delta[n-2] + \alpha_4 \delta[n-3]. \end{aligned}$$

Therefore, the impulse response is a finite-length *sequence* of length 4, which is given by

$$\{h[n]\} = \left\{ \underset{\uparrow}{\alpha_1} \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \right\}$$

The example shows that—for a specific class of sequences—the impulse response happens to be similar to the sequence of the coefficients of the relationship, a rule that is not true in general.

Let's now study the impulse response of the discrete-time accumulator. One has

$$y[n] = \sum_{i=-\infty}^{n} x[i];$$

by substituting with the impulse sequence one gets

$$h[n] = \sum_{i=-\infty}^{n} \delta[l] = \mu[n].$$

Since what an accumulator does, roughly speaking, is to "collect and sum up all the values of a sequence up to time instant n", its impulse response will be 0 up to the origin, after which the value of 1 will be collected. Undoubtedly, this sounds like performing an "integration" operation, although there are some substantial differences involved.§FIXME

The impulse response $h[n]$ of the factor-of-two interpolator

$$y[n] = x_u[n] + \frac{1}{2}(x_u[n-1] + x_u[n+1])$$

is obtained, in the same manner, by setting $x_u[n] = \delta[n]$ and is ultimately

$$h[n] = \delta[n] + \frac{1}{2}(\delta[n-1] + \delta[n+1])$$

which yields the finite-length sequence having length 3

$$\{h[n]\} = \{0.5, \underset{\uparrow}{1}, 0.5\}.$$

2.4 Time-Domain Characterization of LTI Discrete-time Systems

2.4.1 The Convolution Sum

LTI discrete-time system possess peculiar characteristic for their input–output relationship. A consequence of the linear time-invariance property combination is that any LTI discrete-time system is completely characterized by its own impulse response $h[n]$. Since the impulse response is subjected to both superposition principle and it is invariant to shifts of any entity, it suffices to know the impulse response to compute the output of the system for any arbitrary input sequence $x[n]$.

In fact, let $h[n]$ denote the impulse response of an LTI discrete-time system. For instance, let an input be

$$x[n] = 0.5\delta[n+2] + 1.5\delta[n-1] - \delta[n-2] + 0.75\delta[n-5];$$

in order to compute the output $y[n]$ one has to compute its outputs for each addendum separately, and finally adding the individual outputs (superposition principle). As the system is also time-invariant, the rule that governs the input–output relation is the following one,

$$\overset{input}{\delta[n-k]} \longrightarrow \overset{output}{h[n-k]}. \tag{2.15}$$

Under that rule, it's easy to determine the output relative to each and every addendum, as they are

$$\begin{aligned} 0.5\delta[n+2] &\overset{input}{\longrightarrow} 0.5h[n+2] \overset{output}{},\\ 1.5\delta[n-1] &\longrightarrow 1.5h[n-1],\\ -\delta[n-2] &\longrightarrow -h[n-2],\\ 0.75\delta[n-5] &\longrightarrow 0.75h[n-5]; \end{aligned}$$

in the end—thanks to the superposition principle—one gets

$$y[n] = 0.5h[n+2] + 1.5h[n-1] - h[n-2] + 0.75h[n-5].$$

This example highlights the fact that, both superposition principle and shift-invariance prove useful to immediately compute the impulse response of any LTI system. Indeed, since an arbitrary input $x[n]$ can be rewritten as a sum of (many) unit impulses

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k], \tag{2.16}$$

the response of an LTI system to an input of such form will be

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k], \tag{2.17}$$

because of input–output relationship expressed by 2.15. And after rewriting it, one obtains a completely equivalent formulation

$$y[n] = \sum_{k=-\infty}^{\infty} x[n-k]h[k]. \tag{2.18}$$

Equations 2.17 and 2.18 both describe the same operation known as the **convolution sum** of the sequences $x[n]$ and $h[n]$. The operation is so important that a specific operator has been adopted,

$$y[n] = x[n] \circledast h[n]. \tag{2.19}$$

The properties of the convolution sum are the following ones[1]:

- the convolution is *commutative*,

$$g[n] \circledast h[n] = h[n] \circledast g[n];$$

- the convolution is *associative*,

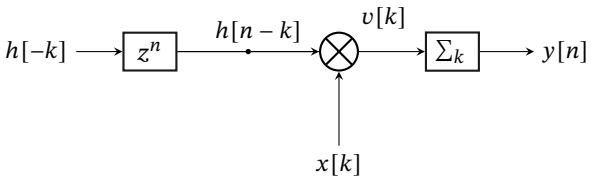
$$(g[n] \circledast h[n]) \circledast k[n] = g[n] \circledast (h[n] \circledast k[n]);$$

- the convolution is *distributive*,

$$g[n] \circledast (h[n] + k[n]) = g[n] \circledast h[n] + g[n] \circledast k[n].$$

Any convolution sum can be interpreted as follows. Suppose the convolution occurs between two sequences $x[n]$ and $h[n]$; the first step to perform is a *time-reverse* of sequence $h[k]$ to form the sequence $h[-k]$. The step that follows is the *time-shift* of $h[-k]$ of n samples if $n > 0$, or on the left by n if $n < 0$ to ultimately form $h[n-k]$. Third step is to form the product $v[k] = x[k]h[n-k]$; at last, one should sum all samples of $v[k]$ to develop the n-th sample of the convolution sum $y[n]$.

A schematic representation of the convolution sum is the following one,



Computation of an output sample using the convolution sum is simply a sum of products between two sequences; it involves fairly simple operations, such as additions, multiplications and delays. Conceptually it is like time-reversing signal $h[k]$ and shifting it through the entire time-domain. For each and every time shift n, product between $x[k]$ and $h[n-k]$ is performed, all samples are summed together and $y[n]$ is determined. The final sequence $\{y[n]\}$ will be the union of all those computed outputs.

Regarding LTI discrete-time systems, the convolution sum is performed between the input sequence $x[n]$ and the impulse response $h[n]$ of the LTI system. As it can be seen, since an impulse response h is always causal, a shifted time-reversed version of the impulse will be—at most—anti-causal, and in general left-sided. For this reason, if $x[n]$ is a causal signal as well, the convolution sum will yield 0 for all values up to n, because the x sequence and the shift-inverted h need to meet at the origin before some non-zero values could be produced from the product and then sum of the

[1] An important remark on this is that the following properties hold only for *infinite-precision* and *infinite-range* representations, as truncated ones might suffer from §FIXME

two sequences. This means that $y[n] = 0$ for $n < 0$ for causal $x[n]$ input sequences[2]

If the two sequences are finite-length, there will be a time instant, let's say, n_1, after which the output convolution sum will collapse to zero. The length of the convolution will be a signal of length $N_x + N_h - 1$, where N_x is the length of the input sequence $x[n]$ and N_h is the length of the impulse response $h[n]$. Of course, the rule still holds for two generic finite-length signals.

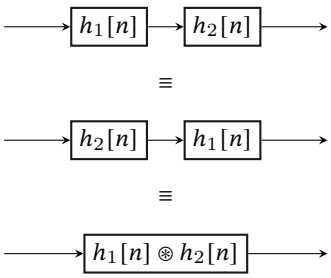
The convolution sum will be able to produce a fine output (in finite time) in the case of *finite length* signals, because the total length of the output will indeed be a finite length signal, although longer than the two input operands. Vice-versa, for any system characterized by an *infinite* impulse response sequence, the convolution sum cannot be adopted to compute the output, since an infinite length output will result from a convolution having at least one of the two operands of infinite length. In those fringe cases a different operator will be considered in place of the convolution sum.

2.5 Simple Interconnection Sche-mes

The simplest interconnection schemas are the **cascade connection** and the **parallel connection**. The first ones are related to the concept of "series", while the latter are linked to the concept of "parallel".

2.5.1 Cascade connections

Cascade connections are "series" connection between two discrete-time systems. If those systems are also LTI, they can be described with their impulse response, leading to the following representations,



[2]The sum of the indices involved in products and sums is equal to the index of the sample being generated by the convolution sum operation. For instance, the computation of the output sample $y[3]$ will involve products of $x[0]h[3]$, $x[1]h[2]$, $x[2]h[1]$ and in the end $x[3]h[0]$. To grasp this, try to substitute inside the convolution sum, step-by-step.

which illustrate that the impulse response $h[n]$ of the cascade of two LTI discrete-time systems—having impulse responses, respectively, of $h_1[n]$ and $h_2[n]$—is given by the formula

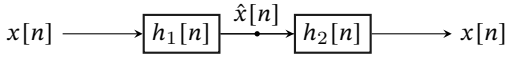
$$h[n] = h_1[n] \circledast h_2[n]. \tag{2.20}$$

Hence, the ordering of two LTI systems "in series" is not important[3], and they can be converted into a single LTI system in which the convolution sum of the two impulse responses is the new impulse response of the unique system. Moreover, a cascade connection of two BIBO stable systems is stable as well, and the same goes for a cascade of two passive—or lossless—systems, which is passive—or lossless—as well.

Cascade connections are important in the development of the so-called *inverse systems*. Let a cascade connection satisfy the relationship

$$h[n] \circledast \hat{h}[n] = \delta[n],$$

then the LTI system corresponding to $\hat{h}[n]$ is said to be the *inverse* of $h[n]$, and vice-versa. Applications of the inverse system are paramount in the recovery of a sequence $x[n]$ from a distorted version $\hat{x}[n]$ of the same one, appearing at the output of a trasmission channel. If the impulse response of the channel is already known, then $x[n]$ can soon be obtained by designing an inverse system of the channel itself, that is



$$h_1[n] \circledast h_2[n] = \delta[n]$$

Some inverse systems are easy to obtain. Consider the discrete-time accumulator, having the impulse response $h[n] = \mu[n]$. The inverse system $\hat{h}[n]$ must satisfy the following relationship

$$\mu[n] \circledast \hat{h}[n] = \delta[n].$$

It immediately follows that $\hat{h}[n] \equiv 0, \forall n < 0$, $\hat{h}[0] = 1$, and $\sum_{l=0}^{n} \hat{h}[l] = 0, \forall n \geq 1$. The only sequence that satisfies the previous relations is the following one,

$$\hat{h}[n] = \delta[n] - \delta[n-1], \tag{2.21}$$

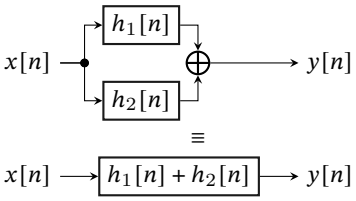
[3]This happens because the ordering of the systems in the cascade has no effect on the overall impulse response, as the convolution sum enjoys the commutative property.

which goes under the name of *backward difference system*. Related input–output relationship is

$$y[n] = x[n] - x[n-1].$$

2.5.2 Parallel connections

Parallel connections are a "parallel" of two connections, that will eventually join together with a *sum* of the two sequences, as in the following diagram



which shows that the output of a parallel connection of two LTI discrete-time systems is simply the sum of the impulse responses of the two systems, given by

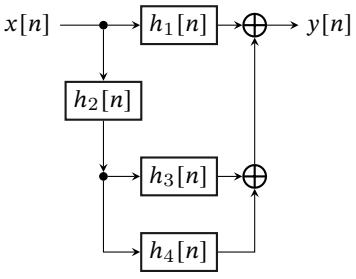
$$h[n] = h_1[n] + h_2[n], \tag{2.22}$$

with $h_1[n]$ and $h_2[n]$ impulse responses of, respectively, the first and the second LTI system.

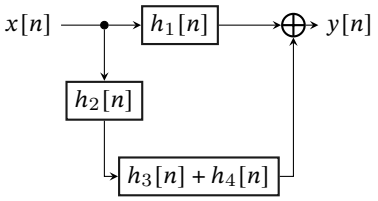
2.5.3 Streamlining the Interconnection Sche-mes

Rules for cascade and parallel configurations can be chained together to simplify an interconnection scheme composed of multiple cascade and parallel systems. Indeed, consider as an example the discrete-time system

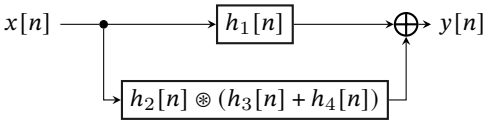
$$\begin{aligned} h_1[n] &= \delta[n] + 0.5\delta[n-1] \\ h_2[n] &= 0.5\delta[n] - 0.25\delta[n-1] \\ h_3[n] &= 2\delta[n] \\ h_4[n] &= -2(0.5)^n \mu[n] \end{aligned}$$



By simplifying the block-diagram with rules 2.20 and 2.22 one obtains



and then



The overall impulse response $h[n]$ is given by the following formula,

$$\begin{aligned} h[n] &= h_1[n] + h_2[n] \circledast (h_3[n] + h_4[n]), \\ &= h_1[n] + h_2[n] \circledast h_3[n] + h_2[n] \circledast h_4[n], \end{aligned}$$

By substituting, one obtains

$$\begin{aligned} h_2[n] \circledast h_3[n] &= \left(\frac{1}{2}\delta[n] - \frac{1}{4}\delta[n-1]\right) \circledast 2\delta[n] \\ &= \delta[n] - \frac{1}{2}\delta[n-1] \\ h_2[n] \circledast h_4[n] &= \left(\frac{1}{2}\delta[n] - \frac{1}{4}\delta[n-1]\right) \circledast \left(-2\left(\frac{1}{2}\right)^n \mu[n]\right) \\ &= -\left(\frac{1}{2}\right)^n \mu[n] + \frac{1}{2}\left(\frac{1}{2}\right)^{n-1} \mu[n-1] \\ &= -\left(\frac{1}{2}\right)^n \mu[n] + \frac{1}{2}\left(\frac{1}{2}\right)^n \mu[n-1] \\ &= -\left(\frac{1}{2}\right)^n \delta[n] \\ &= -\delta[n], \end{aligned}$$

one eventually obtains the final relationship

$$h[n] = \delta[n] + \frac{1}{2}\delta[n-1] + \delta[n] - \frac{1}{2}\delta[n-1] - \delta[n] = \delta[n],$$

which is simply the unit impulse $h[n] = \delta[n]$.

Chapter 3

Linear-Time-Invariant Discrete-time System

As we have already seen in Section 2.3.4, in system analysis—among other fields of study—a **Linear Time-Invariant System (LTI System)** is a system that produces an output signal from any input signal subject to the constraints of linearity and time-invariance; these terms are briefly defined below. These properties apply (exactly or approximately) to many important physical systems, in which case the response $y(t)$ of the system to an arbitrary input $x(t)$ can be found directly using convolution: $y(t) = x(t) \circledast h(t)$ where $h(t)$ is called the *system's impulse response* and symbol '$\circledast$' represents convolution (not to be confused with multiplication, as is frequently employed by the symbol in computer languages). What's more, there are systematic methods for solving any such system (determining $h(t)$), whereas systems not meeting both properties are generally more difficult (or impossible) to solve analytically. A good example of an LTI system is any electrical circuit consisting of resistors, capacitors, inductors and linear amplifiers [3].

In this chapter we will understand in a deeper sense the properties of an LTI system, the conditions under which it is subject, and in the end study the correlation of sequences relative to various kinds of sequences, be them energy signals or power signals.

3.1 Conditions on LTI systems

3.1.1 Stability Condition of an LTI system

In Section 2.3.6 we learnt that *BIBO stable systems* are those under which any bounded input yields a bounded output as well. This is to say that as long as we input a signal with absolute value less than some constant, we are guaranteed to have an output with absolute value less than some other constant.

In this section we will understand the conditions under which any LTI discrete-time system is a BIBO stable system, that is the **stability condition**. The following theorem yields a fundamental property that BIBO stable LTI systems must follow.

Theorem 3.1.1 (BIBO Stability of an LTI discrete-time system)
An LTI discrete-time system is BIBO stable if and only if *its impulse response sequence* $\{h[n]\}$ *is* absolutely summable*, that is when the quantity* $\mathcal{S}$ *is such that*

$$\mathcal{S} = \sum_{n=-\infty}^{\infty} |h[n]| < \infty. \tag{3.1}$$

Proof. Let the impulse response $h[n]$ being a real sequence. Since the input sequence $x[n]$ is bounded, one has that $|x[n]| \leq B_x < \infty$. Therefore, magnitude of the output will be

$$|y[n]| = \left| \sum_{k=-\infty}^{\infty} h[k]x[n-k] \right| \tag{3.2}$$

$$\leq \sum_{k=-\infty}^{\infty} |h[k]||x[n-k]| \tag{3.3}$$

$$\leq B_x \sum_{k=-\infty}^{\infty} |h[k]| \tag{3.4}$$

$$= B_x \mathcal{S} \tag{3.5}$$

two quantities that are indeed bounded.

To prove the inverse relation, the most straightforward way is to simply consider a bounded output $y[n]$, so that

$$|y[n]| \leq B_y.$$

Let the input signal be

$$x[n] = \begin{cases} \text{sgn}(h[-n]), & \text{if } h[-n] \neq 0 \\ K, & \text{if } h[-n] = 0 \end{cases},$$

where the function $\text{sgn}(c) = +1$ if $c > 0$ and $\text{sgn}(c) = -1$ in the opposite case of $c < 0$, and $|K| \leq 1$. The sequence $x[n]$ is bounded, since for each and every of its samples its value is limited.

For this peculiar input, the output $y[n]$ at $n = 0$ will be

$$y[0] = \sum_{k=-\infty}^{\infty} \text{sgn}(h[k])h[k] = \mathcal{S} \leq B_y < \infty,$$

but since $|y[n]| \leq B_y$ this implies that $\mathcal{S} < \infty$. ■

Equation 3.5 shows that the output for an LTI discrete-time system will be bounded, and in particular it will be the product of two bounds—the first related to the input signal $x[n]$, and the second associated to the absolute sum of the impulse response.

Some special LTI discrete-time systems are BIBO stable and are worthy of spending some time on them. Consider, for example, the system with such impulse response

$$h[n] = \alpha^n \mu[n].$$

For this system it holds that, to be BIBO stable,

$$\mathcal{S} = \sum_{n=-\infty}^{\infty} |\alpha|^n \mu[n] = \underbrace{\sum_{n=0}^{\infty} |\alpha|^n}_{|\alpha|<1} = \frac{1}{1-|\alpha|}.$$

For the above reason, if $|\alpha| < 1$ then the system is BIBO stable, as $\mathcal{S} < \infty$. For $|\alpha| \geq 1$ the system is *not* BIBO stable.

3.1.2 Causality Condition of an LTI system

The **causality condition** for an LTI system is a condition that severely limites the design choices of the impulse response for any causal LTI discrete-time system. Let's first exhamine the behavior of two input sequences which, despite being different sequences, are identical before a certain time instant n_0. Indeed, let x_1 and x_2 be two input sequences that follow the subsequent rules,

$$x_1[n] = x_2[n] \text{ for } n \leq n_0,$$
$$x_1[n] \neq x_2[n] \text{ for } n > n_0,$$

then the corresponding outputs at $n = n_0$ of an LTI system with an impulse response $h[n]$ are given by either one of the following formulas,

$$\begin{aligned} y_1[n_0] &= \sum_{k=-\infty}^{\infty} h[k]x_1[n_0-k] \\ &= \sum_{k=0}^{\infty} h[k]x_1[n_0-k] + \sum_{k=-\infty}^{-1} h[k]x_1[n_0-k] \\ y_2[n_0] &= \sum_{k=-\infty}^{\infty} h[k]x_2[n_0-k] \\ &= \sum_{k=0}^{\infty} h[k]x_2[n_0-k] + \sum_{k=-\infty}^{-1} h[k]x_2[n_0-k]. \end{aligned}$$

If the LTI system is also a causal system, then it must be true that

$$y_1[n_0] = y_2[n_0]$$

as the behavior of the two input sequences is identical up to the sample n_0. But this means that

$$\sum_{k=0}^{\infty} h[k]x_1[n_0-k] = \sum_{k=0}^{\infty} h[k]x_2[n_0-k],$$

and also it must be true that

$$\sum_{k=-\infty}^{-1} h[k]x_1[n_0-k] = \sum_{k=-\infty}^{-1} h[k]x_2[n_0-k].$$

Now, since $x_1[n] \neq x_2[n], n > n_0$ the only way the condition

$$\sum_{k=-\infty}^{-1} h[k]x_1[n_0-k] = \sum_{k=-\infty}^{-1} h[k]x_2[n_0-k]$$

is satisfied is when both sums are equal to zero—a condition that is indeed satisfied if and only if

$$h[k] \equiv 0, \forall k < 0. \tag{3.6}$$

Equation 3.6 poses a strict condition on the impulse response $h[n]$ of a causal LTI system: in order for an LTI discrete-time system to be *causal*, its impulse response $\{h[n]\}$ must be a *causal* sequence as well.

As an example, consider the discrete-time system

$$y[n] = \alpha_1 x[n] + \alpha_2 x[n-1] + \alpha_3 x[n-2] + \alpha_4 x[n-3],$$

which will possess an impulse response of the form

$$\{h[n]\} = \left\{ \underset{\uparrow}{\alpha_1} \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \right\},$$

a doubtlessly causal sequence.

Another example is certainly the discrete-time accumulator, defined by 2.3,

$$y[n] = \sum_{l=-\infty}^{n} x[l].$$

The accumulator is a *causal* system as it possesses a causal impulse response, provided by the formula

$$h[n] = \sum_{l=-\infty}^{n} \delta[i] = \mu[n].$$

A good counter–example is the factor-of-2 interpolator, having equation

$$y[n] = x_u[n] + \frac{1}{2}(x_u[n-1] + x_u[n+1]).$$

The interpolator is a *noncausal* system, because it has a noncausal impulse response that's given by

$$\{h[n]\} = \left\{ 0.5 \quad \underset{\uparrow}{1} \quad 0.5 \right\}.$$

As we have previously seen, any noncausal LTI discrete-time system with a finite-length impulse response can be realized as causal systems by inserting appropriate amounts of delay, to push the "future" samples back into the past so that there is no term $x[n+k]$ with $k > 0$. Adopting this method, the simplest causal version of the above system is the one delayed of a single time unit,

$$y[n] = x_u[n-1] + \frac{1}{2}(x_u[n-2] + x_u[n]).$$

3.2 Finite-Dimensional LTI Discrete-time Systems

A very important subclass of LTI systems is characterized by a linear constant coefficients difference equation. Those systems must abide to the following form

$$\sum_{k=0}^{N} d_k y[n-k] = \sum_{k=0}^{M} p_k x[n-k], \tag{3.7}$$

where $x[n]$ and $y[n]$ are, respectively, the input and the output of the LTI system. Sequences $\{d_k\}$ and $\{p_k\}$ are constants characterizing the finite-dimensional system.

Finite-dimensional LTI systems possess an *order*; the order of the system is given by the quantity $\max(N, M)$, which is indeed the order of the difference equation. It is possible to implement any LTI systems characterized by a constant coefficient difference equation with two finite sums of products. In fact, if one assumes that the system is a causal LTI finite-dimensional system and that the coefficient $d_0 \neq 0$, then the output $y[n]$ can be obtained by *recursively* computing the output

$$y[n] = -\sum_{k=1}^{N} \frac{d_k}{d_0} y[n-k] + \sum_{k=0}^{M} \frac{p_k}{d_0} x[n-k], \tag{3.8}$$

obtained by rearranging Equation 3.7. The output can be obtained this way for all $n \geq n_0$, by knowing both $x[n]$ and all the N initial conditions $y[n_0-1], y[n_0-2], \ldots, y[n_0-N]$.

3.3 Classification of LTI Discrete-time Systems

Linear-Time-Invariant discrete-time systems can be classified by some of their properties, such as

- by **length of impulse response**—usually, one wants to deal with finite-length signals, so it's very important to divide impulse responses that might generate infinite-length sequences from those who do not;
- by **output calculation process**—the disctinction is based on the process from which one can compute the output, that is if it is possible to compute the output sequence recursively or not;
- by **coefficients nature**—indeed, coefficients might be real or complex.

Classification based on impulse response length

Impulse response length is a crucial aspect to know when dealing with LTI systems. Like we have already seen, if the impulse response $h[n]$ is of finite length then the convolution with a finite-length input sequence $x[n]$ *will* yield a finite-length output as well. All systems that possess a finite-length impulse response are said to be **finite impulse response (FIR)** discrete-time systems. FIR systems are those for which

$$h[n] = 0 \text{ for } n < N_1, \text{ and } n > N_2,$$

with $N_1 < N_2$.

The convolution sum of a FIR system can simply be written as

$$y[n] = \sum_{k=N_1}^{N_2} h[k]x[n-k].$$

The output $y[n]$ of a FIR LTI discrete-time system can be computed directly from the convolution sum, as it is a finite sum of products (and thus we can use the convolution sum). Examples of FIR LTI systems are the M-moving average system and the linear interpolators, both of which have a finite-length impulse response.

If the impulse response is of infinite length, the corresponding system is known as an **infinite impulse response (IIR)**, a concept that it is opposed to the FIR one. The class of the IIR systems are characterized by a linear constant coefficient difference equations, and for this reason it cannot be computed by means of a convolution sum. Examples of IIR systems are the accumulator, which as we already know, is defined by

$$y[n] = y[n-1] + x[n]$$

which is clearly a linear constant coefficient difference equation yielding a recursion system.

Classification based on output calculation process

Calculation process plays a determinant role over classification of LTI systems. There are two categories of LTI systems concerning calculation process:

- *nonrecursive systems*, that are those whose output can be calculated sequentially, knowing only the present and past input samples;
- *recursive systems* are instead all those systems whose output computation in addition to the present and past input samples has to take into account past (some) output samples.

Classification based on coefficients nature

Even coefficients nature plays a role into sequences classification. As coefficients can be either real or complex, there are two categories:

- *real discrete-time systems*—whose impulse response samples are real valued;
- *complex discrete-time systems*—whose impulse response samples are complex valued, that is when *at least one* coefficient is complex valued.

3.4 Correlation of signals

In general, the **correlation of two signals** or waveforms is defined as the measure of *similarity* between those signals. There are many applications in which it might be necessary to *compare* one reference signal with one or more signals to determine the similarity between the pair, and ultimately to determine additional information based off the similarity.

To name one, in digital communications a set of data symbols in a predetermined alphabet are represented by a set of unique discrete-time sequences that codify them. If one of these sequences has been transmitted, the receiver has to determined *which particular sequence has been received,* and it does so by *comparing the received signal with every member of possible sequences from the set*. In such an application, the correlation of two sequences comes in handy, as it is a measure of similarity—the more the similar two sequences are, the more probable a received sequence codifies a certain symbol in the alphabet.

Another application in which correlation proves useful is in the radar and sonar applications; the received signal refected from the target is a delayed version of the transmitted signal, and by *measuring the delay* one can soon determine the location of the target. Correlation helps on this as measuring the delay might be brought back as the problem of finding the greatest similarity (correlation) between two sequences, the received sequence and a delayed step sequence $A\mu[n-\theta]$, with A amplitude and θ delay.§FIXME[prove

this claim] The detection problem gets much more complicated in practice—as often the received signal is corrupted by additive, random noise.

The correlation of signals idea leads to the following definition of **cross-correlation**,

Definition 3.4.1 (Cross-correlation)**.** Let $x[n]$ and $y[n]$ be two energy signals. The *cross-correlation sequence* $r_{xy}[l]$ is a sequence such that

$$r_{xy}[l] = \sum_{n=-\infty}^{\infty} x[n]y[n-l], l = 0, \pm 1, \pm 2, \dots \tag{3.9}$$

where l is a parameter called *lag*, that indicates the relative time-shift between the pair of sequences. Signal $x[n]$ is said to be the *reference sequence*.

The cross-correlation between signals resembles the convolution sum, except the fact one of the arguments in the sequences is opposite—in this case, one has $n-l$ in place of $l-n$ like in the case of the convolution sum. Cross-correlation is a measure of similarity between a pair of *energy signals*. Indeed, this similarity is much clearer if one rewrites the expression for the cross-correlation as follows,

$$\begin{aligned} r_{xy}[l] &= \sum_{n=-\infty}^{\infty} x[n]y[n-l] \\ &= \sum_{n=-\infty}^{\infty} x[n]y[-(l-n)] \\ &= x[l] \circledast y[-l]. \end{aligned}$$

a rewriting that remarks that the cross-correlation is like performing a convolution with one signal whose arguments are inverted, with $y[-l]$ instead of $y[l]$.

The signal $y[n]$ is said to be shifted by l samples to the *right* with respect to the reference sequence $x[n]$ for positive values of l, and on contrary it is said to be shifted by l samples to the *right* for negative values of l. This is as expected, as the l is simply a delay–advance parameter very similar to the case of the convolution sum. The ordering of the subscripts xy in the definition of the cross-correlation $r_{xy}[l]$ is important specifies that $x[n]$ is the *reference sequence* around which the sequence $y[n]$ is shifted with respect to.

Of course, signals can be interchanged. The cross-correlation equation would soon become

$$\begin{aligned} r_{yx}[l] &= \sum_{n=-\infty}^{\infty} y[n]x[n-l] \\ &= \sum_{m=-\infty}^{\infty} x[m+l]y[m] \\ &= r_{xy}[-l], \end{aligned}$$

which highlights that $r_{yx}[l]$ can be obtained by *time-reversing* $r_{xy}[l]$.

What happens now if one performs a cross-correlation between two signals that are ultimately the same? The operation is said to be the **autocorrelation** and it is given by the following definition,

Definition 3.4.2 (Autocorrelation)**.** Let $x[n]$ be an energy signal. The *autocorrelation sequence* of $x[n]$ is given by

$$r_{xx}[l] = \sum_{n=-\infty}^{\infty} x[n]x[n-l] \tag{3.10}$$

obtained by setting $y[n] = x[n]$ in the definition of the cross-correlation sequence $r_{xy}[l]$.

A peculiar remark of the autocorrelation is that its value at time instant 0 is equal to the total energy of the signal $x[n]$. That is,

Predicate 3.4.1 (Equivalence between autocorrelation and energy)
Let $x[n]$ be an energy signal and $r_{xx}[l]$ be the autocorrelation function of x. Then,

$$r_{xx}[0] = \mathcal{E}_x. \tag{3.11}$$

Proof. It is enough to substitute

$$r_{xx}[0] = \sum_{n=-\infty}^{\infty} x[n]x[n-l] = \sum_{n=-\infty}^{\infty} |x[n]|^2 = \mathcal{E}_x$$

as the sum is equal to the definition of total energy as in Equation 1.6. ∎

The above predicate shows that, when the correlation is computed with no delay involved, the total energy is obtained. From the relation $r_{yx}[l] = r_{xy}[-l]$ it immediately

follows that $r_{xx}[l] = r_{xx}[-l]$, implying that the autocorrelation is an *even* function for real sequences $x[n]$.

3.4.1 Properties of cross-correlation and autocorrelation

Let $x[n]$ and $y[n]$ be two energy sequences. The energy of the combined sequence $\alpha x[n] + y[n-l]$ is also finite and nonnegative[1], that is

$$\sum_{n=-\infty}^{\infty} (\alpha x[n] + y[n-l])^2 = \underbrace{\alpha^2 \sum_{n=-\infty}^{\infty} x^2[n]}_{\mathcal{E}_x} + \underbrace{2\alpha \sum_{n=-\infty}^{\infty} x[n]y[n-l]}_{cross-correlation} + \underbrace{\sum_{n=-\infty}^{\infty} y^2[n-l]}_{\mathcal{E}_y} \geq 0.$$

This means that

$$\alpha^2 r_{xx}[0] + 2\alpha r_{xy}[l] + r_{yy}[0] \geq 0,$$

where $r_{xx}[0] = \mathcal{E}_x > 0$ and $r_{yy}[0] = \mathcal{E}_y > 0$.

One can rewrite the above equations as the following matrix system,

$$\begin{bmatrix} \alpha & 1 \end{bmatrix} \begin{bmatrix} r_{xx}[0] & r_{xy}[l] \\ r_{xy}[l] & r_{yy}[0] \end{bmatrix} \begin{bmatrix} \alpha \\ 1 \end{bmatrix} \geq 0,$$

that is

$$\begin{bmatrix} \alpha & 1 \end{bmatrix} \begin{bmatrix} \mathcal{E}_x & r_{xy}[l] \\ r_{xy}[l] & \mathcal{E}_y \end{bmatrix} \begin{bmatrix} \alpha \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 1 \end{bmatrix} \boldsymbol{R} \begin{bmatrix} \alpha \\ 1 \end{bmatrix} \geq 0,$$

for any finite value of α.

The matrix $\boldsymbol{R}$ is a *positive semidefinite* matrix. As such, $\det(\boldsymbol{R}) \geq 0$, that is

$$\det(\boldsymbol{R}) = r_{xx}[0]r_{yy}[0] - r_{xy}^2[l] \geq 0$$

or equivalently

$$|r_{xy}| \leq \sqrt{r_{xx}[0]r_{yy}[0]} = \sqrt{\mathcal{E}_x \mathcal{E}_y}. \tag{3.12}$$

[1] Energy is nonnegative because it is squared.

The latter specifies that the magnitude of the cross-correlation is always smaller than the geometric average between the energies of signals x and y, and provides an upper bound for the cross-correlation samples. In the case of the autocorrelation it soon becomes

$$|r_{xx}[l]| \leq r_{xx}[0] = \mathcal{E}_x, \tag{3.13}$$

a remarkable property of the autocorrelation. In practice, above inequality establishes that the autocorrelation has (at least) a peak in its time instant 0—or when $x[n]$ is completely superposed on itself—whose value is equal to the total energy ofthe signal $x[n]$. At zero lag $l = 0$ the sample value of the autocorrelation sequence has its maximum value, which is equal to the total energy of the sequence.

Consider now the case of

$$y[n] = \pm b x[n-N],$$

where N is an integer and $b > 0$ is an arbitrary number. That case,

$$\mathcal{E}_y = b^2 \mathcal{E}_x,$$

since the multiplication constant appears squared after the energy evaluation.

But this means that the geometric mean

$$|r_{xy}[l] \leq \sqrt{\mathcal{E}_x \mathcal{E}_y} = \sqrt{b^2 \mathcal{E}_x^2} = b\mathcal{E}_x$$

for a signal scaled by factor b. By means of inequality

$$|r_{xy}[l]| \leq \sqrt{\mathcal{E}_x \mathcal{E}_y}$$

one obtains

$$-b\mathcal{E}_x = -b r_{xx}[0] \leq |r_{xy}[l]| \leq b r_{xx}[0] = b\mathcal{E}_x, \tag{3.14}$$

an important inequality that expresses a precise rule regarding bounds for all signals scaled by a constant factor b. By scaling a signal $x[n]$ by a constant factor b, that is picking $y[n] = bx[n]$ with $b \in \mathbb{R} \setminus \{0\}$ the bound for the cross-correlation expressed by the total energy increases by a factor of b as well.

Autocorrelation and cross-correlation can both be easily computed by means of the MATLAB function `xcorr`.

3.4.2 Normalized forms of correlation

Correlation possesses so-called **normalized forms**. Normalized forms are useful to compare correlations with other signals or to obtain a value that is bounded between -1 and $+1$.

Definition 3.4.3 (Normalized autocorrelation and cross-correlation)**.** Let $x[n]$ and $y[n]$ be energy signals—the *normalized autocorrelation* and *normalized cross-correlation* sequences are given by, respectively,

$$\rho_{xx}[l] = \frac{r_{xx}[l]}{r_{xx}[0]}, \rho_{xy}[l] = \frac{r_{xy}[l]}{\sqrt{r_{xx}[0] r_{yy}[0]}}. \tag{3.15}$$

Normalized forms are often employed for convenience in comparing and displaying correlation values, as both are smaller than 1. Indeed,

$$-1 \leq \rho_{xx}[l] \leq +1, -1 \leq \rho_{xy}[l] \leq +1,$$

independently of the range of values of sequences $x[n]$ and $y[n]$.

3.4.3 Computing the correlation for various kinds of signals

Power signals are all those that, despite being of infinite-length, possess a finite amount of average power as expressed in Equation 1.7. The cross-correlation for them is defined as in the following definition,

Definition 3.4.4 (Cross-correlation for power signals)**.** Let $x[n]$ and $y[n]$ be *power signals*. Then the *cross-correlation* r_{xy} is given by

$$r_{xy}[l] = \lim_{K \to \infty} \frac{1}{2K+1} \sum_{n=-K}^{K} x[n] y[n-l], \tag{3.16}$$

with $\frac{1}{2K+1}$ normalization factor.

Similarly, one can define the autocorrelation for a power signal as in the following definition,

Definition 3.4.5 (Autocorrelation for power signals)**.** Let $x[n]$ be a *power signal*. Then the *autocorrelation* r_{xx} is given by

$$r_{xx}[l] = \lim_{K \to \infty} \frac{1}{2K+1} \sum_{n=-K}^{K} x[n] x[n-l], \tag{3.17}$$

with $\frac{1}{2K+1}$ normalization factor.

The correlation sequences have a distinct formula for periodic signals as well. Following the footsteps of the previous two definitions, one might want to define cross-correlation and autocorrelation for periodic sequences as well.

Definition 3.4.6 (Cross-correlation for periodic signals)**.** Let $x[n]$ and $y[n]$ be periodic signals of period N. Then the *cross-correlation* r_{xy} is defined as

$$r_{xy}[l] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] y[n-l]. \tag{3.18}$$

Definition 3.4.7 (Autocorrelation for periodic signals)**.** Let $x[n]$ be a periodic signal of period N. Then the *autocorrelation* r_{xx} is defined as

$$r_{xx}[l] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] x[n-l]. \tag{3.19}$$

Both quantities, as expected, are evaluated in a single period of the involved sequences. They will produce periodic signals, therefore both cross-correlation $r_{xy}[l]$ and $r_{xx}[l]$ are *periodic signals of period* N. The periodicity property of the autocorrelation sequence can be exploited to determine the period of any periodic signal that might be affected by an additive random noise. Periodic signals affected by random noise will lose their periodicity, as the impredictable and random nature of the noise will inevitably lead to a non-periodic signal.

Indeed, it is sufficient to evaluate the autocorrelation for a disturbed signal $w[n] = x[n] + d[n]$, with x original signal and d additive noise, and evaluate the autocorrelation for signal $w[n]$. If w is composed of $M-1$ samples (that is, we collected M samples of w, from $0 \leq n \leq M-1$) and $M \ll N$, it is possible to capture the period of x by computing the

autocorrelation of $w[n]$. In fact,

$$\begin{aligned} r_{ww}[l] &= \frac{1}{M}\sum_{n=0}^{M-1} w[n]w[n-l] \\ &= \frac{1}{M}\sum_{n=0}^{M-1} (x[n]+d[n])(x[n-l]+d[n-l]) \\ &= \frac{1}{M}\sum_{n=0}^{M-1} x[n]x[n-l] + \frac{1}{M}\sum_{n=0}^{M-1} d[n]d[n-l] \\ &+ \frac{1}{M}\sum_{n=0}^{M-1} x[n]d[n-l] + \frac{1}{M}\sum_{n=0}^{M-1} d[n]x[n-l] \\ &= \underbrace{r_{xx}[l]}_{\text{period } N} + \underbrace{r_{dd}[l]}_{\delta[n]} + \underbrace{r_{xd}[l]}_{\simeq 0} + \underbrace{r_{dx}[l]}_{\simeq 0}. \end{aligned}$$

In the last equation $r_{xx}[l]$ is a periodic sequence having period N, as the autocorrelation of a periodic signal is periodic as well, possessing the same period. The latter term will possess peaks at multiples of the period $l = 0, N, 2N, \ldots$, maxima that can be easily detected. Autocorrelation sequence will have the same amplitudes as l approaches M. The other terms deserve to be tackled separately,

- $r_{dd}[l]$—This term is the unit impulse $\delta[n]$, as the random additive noise has an autocorrelation of 1 located in the origin—purely random noises are never correlated with themselves, except in the origin, in the case of zero delay $l = 0$;
- $r_{xd}[l]$—This term is roughly equal to zero, as $x[n]$ and $d[n]$ are scarcely correlated ($d[n]$ appears to be a squisitely random sequence);
- $r_{dx}[l]$—The same as above goes for the inverted cross-correlation between d and x.

Indeed, evaluating autocorrelation of $w[n]$ is equivalent to performing two autocorrelations and two cross-correlations, with the value of the latter ones almost equal to zero and the autocorrelation of the random additive noise being simply a delta-impulse located in the origin, affecting only evaluation of time instant 0 in the autocorrelation of $w[n]$. Peaks of $r_{ww}[l]$ for $l > 0$ are essentially due to the peaks of $r_{xx}[l]$ and can thus be employed to period detection as their peaks will essentially be the peaks related to the period of the autocorrelation of sequence $x[n]$, and can be soon identified as the period—on contrary, if autocorrelation of sequence $w[n]$ does not manifest any periodicity in the resulting peaks, there will likely be no period at all.

Chapter 4

Discrete-time Signals in the Frequency Domain

In physics, electronics, control systems engineering, and statistics, the **frequency domain** refers to the analysis of mathematical functions or signals with respect to frequency, rather than time. Put simply, a time-domain graph shows how a signal changes over time, whereas a frequency-domain graph shows how much of the signal lies within each given frequency band over a range of frequencies. A frequency-domain representation can also include information on the phase shift that must be applied to each sinusoid in order to be able to recombine the frequency components to recover the original time signal.

A given function or signal can be converted between the time and frequency domains with a pair of mathematical operators called transforms. An example is the Fourier transform, which converts a time function into a complex valued sum or integral of sine waves of different frequencies, with amplitudes and phases, each of which represents a frequency component. The "spectrum" of frequency components is the frequency-domain representation of the signal. The inverse Fourier transform converts the frequency-domain function back to the time-domain function. A spectrum analyzer is a tool commonly used to visualize electronic signals in the frequency domain.

Some specialized signal processing techniques use transforms that result in a joint time–frequency domain, with the instantaneous frequency being a key link between the time domain and the frequency domain.

One of the main reasons for using a frequency-domain representation of a problem is to *simplify the mathematical analysis*. For mathematical systems governed by linear differential equations, a very important class of systems with many real-world applications, converting the description of the system from the time domain to a frequency domain converts the differential equations to algebraic equations, which are much easier to solve.

In addition, looking at a system from the point of view of frequency can often give an intuitive understanding of the qualitative behavior of the system, and a revealing scientific nomenclature has grown up to describe it, characterizing the behavior of physical systems to time varying inputs using terms such as bandwidth, frequency response, gain, phase shift, resonant frequencies, time constant, resonance width, damping factor, Q factor, harmonics, spectrum, power spectral density, eigenvalues, poles, and zeros.

An example of a field in which frequency-domain analysis gives a better understanding than time domain is *music;* the theory of operation of musical instruments and the musical notation used to record and discuss pieces of music is implicitly based on the breaking down of complex sounds into their separate component frequencies (musical notes) [4].

This chapter will be devoted in the introduction to the analysis of discrete-time signals in the frequency domain, by means of the *discrete-time Fourier Transform*.

4.1 Continuous-time Fourier Transform

A **Fourier transform (FT)** is a mathematical transform that decomposes functions depending on space or time into functions depending on spatial frequency or temporal frequency. That process is also called analysis. An example application would be decomposing the waveform of a musical chord into terms of the intensity of its constituent pitches. The term Fourier transform refers to both the frequency domain

representation and the mathematical operation that associates the frequency domain representation to a function of space or time.

The Fourier transform of a function is a complex-valued function representing the complex sinusoids that comprise the original function. For each frequency, the magnitude (absolute value) of the complex value represents the amplitude of a constituent complex sinusoid with that frequency, and the argument of the complex value represents that complex sinusoid's phase offset. If a frequency is not present, the transform has a value of 0 for that frequency. The Fourier transform is not limited to functions of time, but the domain of the original function is commonly referred to as the time domain. The Fourier inversion theorem provides a synthesis process that recreates the original function from its frequency domain representation [5].

Functions that are localized in the time domain have Fourier transforms that are spread out across the frequency domain and vice versa, a phenomenon known as the *uncertainty principle*. The critical case for this principle is the Gaussian function, of substantial importance in probability theory and statistics as well as in the study of physical phenomena exhibiting normal distribution (e.g., diffusion). The Fourier transform of a Gaussian function is another Gaussian function. Joseph Fourier introduced the transform in his study of heat transfer, where Gaussian functions appear as solutions of the heat equation.

Definition 4.1.1 (Continuous-time Fourier Transform)**.** The *Continuous-time Fourier Transform* of a continuous-time signal $x_a(t)$ is given by

$$X_a(j\omega) = \int_{-\infty}^{\infty} x_a(t) e^{-j\omega t} dt. \tag{4.1}$$

The Continuous-time Fourier Transform is often referred to as the *Fourier specturm* or the *spectrum* of a continuous-time signal, with notation

$$x_a(t) \longleftrightarrow X_a(j\omega).$$

Definition 4.1.2 (Inverse Continuous-time Fourier Transform)**.** The *Inverse Continuous-time Fourier Transform* of a Fourier transform $X_a(t)$ is given by

$$x_a(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X_a(j\omega) e^{+j\omega t} d\omega. \tag{4.2}$$

The inverse transform is often referred to as the *Fourier integral*.

Variable ω is real, and denotes the continuous-time angular frequency in *radians/sec* if the unit of the independent variable t is in *seconds*. In general, a Continuous-time Fourier Transform is a complex analog function of ω in the range of $-\infty < \omega < \infty$. As any complex function, it can be rewritten in its polar form, that is

$$X_a(j\omega) = |X_a(j\omega)|\, e^{j\theta_a(\omega)}, \tag{4.3}$$

where the function

$$\theta_a(\omega) = \arg\{X_a(j\omega)\} \tag{4.4}$$

is the argument of the Fourier Transform[1].

The quantity $|X_a(j\omega)|$ is called the *magnitude spectrum* and the quantity $\theta_a(\omega)$ is called the *phase spectrum*. Both quantities are real functions and concur in the definition of the Continuous-time Fourier Transform. In general, the Continuous-time Fourier Transform exists if the original signal in the time domain satisfies the **Dirichlet conditions**,

Rule 4.1.1 (Dirichlet Conditions)
Let $x_a(t)$ be a continuous-time signal in the time-domain. Signal $x_a(t)$ satisfies the Dirichlet conditions *if it satisfies both*

1. *the signal $x_a(t)$ has a finite number of discontinuities and a finite number of maxima and minima in any finite interval;*

[1] The argument of a complex number $x + iy$ is provided by the function

$$\operatorname{Arg}(x + iy) = \operatorname{atan2}(y, x),$$

that is equal to

$$\operatorname{Arg}(x + iy) = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{if } x > 0, \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{if } x > 0 \text{ and } y \geq 0, \\ \arctan\left(\frac{y}{x}\right) - \pi & \text{if } x > 0 \text{ and } y < 0, \\ +\frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0, \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0, \\ \text{undefined} & \text{if } x = 0 \text{ and } y = 0. \end{cases}$$

2. *the signal is absolutely summable, that is when*

$$\int_{-\infty}^{\infty} |x_a(t)|dt < \infty$$

The following theorem, without proposed proof, establishes a consequence of a signal following the Dirichlet conditions.

Theorem 4.1.1 (Convergence of a signal)
Let $x_a(t)$ be a continuous-time signal in the time-domain that satisfies the Dirichlet conditions. Then, the inverse continuous-time Fourier Transform

$$\frac{1}{2\pi}\int_{-\infty}^{\infty} X_a(j\omega)e^{j\omega t}d\omega$$

converges to $x_a(t)$ at all *values of t except at values of t where $x_a(t)$ manifests discontinuities.*

It can be shown that, if $x_a(t)$ is *absolutely integrable*, then its Fourier Transform is bounded, $|X_a(j\omega)| < \infty$, ultimately implying the existence of the Continuous-time Fourier Transform.

4.1.1 Parseval's Theorem and Energy Density Spectrum

The distribution of the energy of a signal in the frequency domain is known as *energy spectral density (ESD)* or *energy density (ED)* or *energy density spectrum*.

The following theorem, known as **Parseval's theorem**, opens the door to the definition of the energy density spectrum, and it is a much important result in signal theory.

Theorem 4.1.2 (Parseval's theorem)
Let $x(t) \in \mathbb{C}$ be a finite energy continuous-time signal in the time-domain and $X(j\omega)$ its Fourier Transform. Then,

$$\int_{-\infty}^{\infty} |x(t)|^2 dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |X(j\omega)|^2 \, d\omega \tag{4.5}$$

Proof. The total energy $\mathcal{E}_x$ of $x(t)$ is given by the following formula,

$$\mathcal{E}_x = \int_{-\infty}^{\infty} |x(t)|^2 dt = \int_{-\infty}^{\infty} x(t)x^*(t)dt;$$

by rewriting the above expression—substituting $x^*(t)$ with its Fourier Transform—one obtains

$$\mathcal{E}_x = \int_{-\infty}^{\infty} x(t)\left(\frac{1}{2\pi}\int_{-\infty}^{\infty} X^*(j\omega)e^{-j\omega t}d\omega\right)dt.$$

Finally, interchanging the order of the integration, one gets

$$\begin{aligned}\mathcal{E}_x &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X^*(j\omega)\left(\int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt\right)d\omega,\\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X^*(j\omega)X(j\omega)d\omega,\\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} |X(j\omega)|^2 \, d\omega,\end{aligned}$$

which leads to the thesis,

$$\int_{-\infty}^{\infty} |x(t)|^2 dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |X(j\omega)|^2 \, d\omega.$$

■

The quantity $|X(j\omega)|^2$ is called **energy density spectrum** of the signal $x(t)$ and it is often denoted as

$$\mathcal{S}_{xx}(\omega) = |X(j\omega)|^2 . \tag{4.6}$$

The energy density spectrum plays a crucial role in determining the arrangement of the energy over the frequency domain. In fact, the energy over a specified range of frequencies $r = [\omega_a, \omega_b]$, that is for frequencies $\omega_a \leq \omega \leq \omega_b$, can be computed by means of the energy density spectrum, by integrating over the latter:

$$\mathcal{E}_{x,r} = \frac{1}{2\pi}\int_{\omega_a}^{\omega_b} S_{xx}(\omega)d\omega. \tag{4.7}$$

In practice, the energy density spectrum allows to select a portion $r = [\omega_a, \omega_b]$ of the frequency spectrum and to determine the arrangement of the frequencies by obtaining the energy that a signal $x(t)$ possesses in the selected frequency range. In plain words, the energy will be arranged over the frequency domain with a distribution that depends

on the signal; since the energy density spectrum describes the frequency arrangement in the frequency domain, integrating over it yields the value of the energy in that selected frequency interval.

4.1.2 Band-limited Continuous-time Signals

Generally speaking, the more concentrated $f(x)$ is, the more spread out its Fourier transform $F(\omega)$ must be. In particular, the scaling property of the Fourier transform may be seen as saying: if we squeeze a function in x, its Fourier transform stretches out in ω, and vice-versa. It is not possible to arbitrarily concentrate both a function and its Fourier transform. This phenomenon is due to the uncertainty principle.

Indeed, depending its the Fourier Transform, a signal could lay either in the entire frequency domain or in a portion of it. Susceptible to the case, a signal could be:

- either a *full-band* signal, when it takes up the *whole* frequency range $-\infty < \omega < \infty$;
- or a *band-limited* signal, when it is limited to *a portion* of the frequency range $-B < \omega < B$, with B *band* of the signal.

A special class of signals deserves to be mentioned: an **ideal band-limited** signal has a spectrum that is zero outside a finite frequency range $[\omega_a, \omega_b]$, that is a signal whose Fourier Transform is defined as

$$X(j\omega) = \begin{cases} 0, & 0 \leq |\omega| \leq \omega_a \\ 0, & \omega_b \leq |\omega| \leq \infty \\ X(j\omega) & \omega_a \leq |\omega| \leq \omega_b \end{cases} . \tag{4.8}$$

An ideal band-limited signal can never be generated in practice, as the uncertainty principle would require an infinite-length sequence for it to be generated.

Band-limited signals can be classified accordingly to their frequency range, that is where the majority of the signal's energy is concentrated,

Lowpass signals are all signals arranged in a frequency range $|\omega| \leq \omega_p < \infty$, where ω_p is said to be the *bandwidth* of the signal;

Highpass signals are all signals arranged in a frequency range $0 < \omega_p \leq |\omega| < \infty$, where the bandwitdh of the signal is in the interval $[\omega_p, \infty[$.

Bandpass signals are any signal whose spectrum is located in the frequency range $[\omega_L, \omega_H]$, that is $0 < \omega_L \leq |\omega| \leq \omega_H < \infty$, where the bandwidth is given by $\mathrm{BW} = \omega_H - \omega_L$.

4.2 Discrete-time Fourier Transform

The frequency-domain representation of a discrete-time sequence is the **discrete-time Fourier Transform (DTFT)**. The Fourier transform maps a time-domain sequence into a continuous function of the frequency variable ω.

Indeed, the DTFT represents a different view on the sequences and systems, that is:

- based on the superposition of sinusoidal basis functions;
- suitable for compressed representations of data;
- suitable for manipulating and designing systems.

Definition 4.2.1 (Discrete-time Fourier Transform)**.** Let $x[n]$ be a sequence. The *discrete-time Fourier Transform (DTFT)*[2] $X(e^{j\omega})$ of $x[n]$ is given by

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}, \tag{4.9}$$

where ω is a continuous variable in the range $-\infty < \omega < \infty$. The infinite series given by 4.9 may or may not converge depending on sequence $x[n]$; if it converges for all values of ω, then the discrete-time Fourier Transform exists.

The above definition provides *two* important factors regarding the Discrete-time Fourier Transform: the first one is the transform formula, declared in Equation 4.9, and the second one the fact that the Discrete-time Fourier Transform cannot exist if the infinite series does not converge. In order for a Discrete-time Fourier Transform to exist, the series must converge.

[2] Not to be confused with the *Discrete Fourier Transform (DFT)* or with the *z-Transform*.

Generally speaking, $X(e^{j\omega})$ is a *complex* function of *real* variable ω and can be therefore written as

$$X(e^{j\omega}) = X_{re}(e^{j\omega}) + jX_{im}(e^{j\omega}),$$

with $X_{re}(e^{j\omega})$ and $jX_{im}(e^{j\omega})$, respectively, being the real and imaginary parts of the Discrete-time Fourier Transform and both are real functions of ω. $X(e^{j\omega})$ can also be expressed in its polar form,

$$X(e^{j\omega}) = \underbrace{\left|X(e^{j\omega})\right|}_{magnitude} \underbrace{e^{j\theta(\omega)}}_{phase},$$

where

$$\theta(\omega) = \arg\{X(e^{j\omega})\}.$$

Function $\left|X(e^{j\omega})\right|$ is called the *magnitude function* or **magnitude spectrum**, while $\theta(\omega)$ is called the *phase function* or **phase spectrum**. Both quantities are real functions of ω, and the Discrete-time Fourier Transform is also called *Fourier spectrum*.

When decomposing the Discrete-time Fourier Transform into its real and imaginary part or in its polar form, interesting patterns suddenly appear. For a real sequence $x[n]$, both magnitude $\left|X(e^{j\omega})\right|$ and real part $X_{re}(e^{j\omega})$ are *even* functions in ω, whereas both phase spectrum $\theta(\omega)$ and imaginary part $X_{im}(e^{j\omega})$ are *odd* functions of ω.

The discrete-time Fourier Transform is a *periodic* function in ω, with a period 2π. Due to exponentiation properties,

$$\begin{aligned} X(e^{j\omega}) &= \left|X(e^{j\omega})\right| e^{j\theta(\omega)} \\ &= \left|X(e^{j\omega})\right| e^{j\theta(\omega)+2\pi k} \end{aligned}$$

for *any* integer k. This means, however, that the phase function $\theta(\omega)$ cannot be *uniquely* specified for any Discrete-time Fourier Transform, and that the range must be appropriately selected for that function, to exclude repetitions. Unless otherwise stated, one shall assume that the phase function $\theta(\omega)$ is restricted to the range of values $[-\pi, \pi[$, that is the *principal value*

$$-\pi \leq \theta(\omega) < \pi. \tag{4.10}$$

The principal value is completely sufficient to characterize all the crucial aspects of the frequency response, as values outside the principal value would regardless yield repetitions in the frequency space, as the period of the exponential is 2π.

Moreover, it is sufficient to know only one *half* of the values in the principal value for both magnitude and phase functions. This is because they are—respectively—even and odd functions whose values can be faithfully reconstructed even knowing only a half of the values.

The discrete-time Fourier Transforms for some sequences exhibit discontinuities of 2π in their phase responses—therefore, an alternate type of phase function that is a continuous function of ω is often used in place of the function that gives discontinuities. The process of removing the discontinuities is said to be *"unwrapping"*, and consists in deriving a new phase function from the original phase one by removing the discontinuities of 2π. The resulting phase function is denoted as $\theta_c(\omega)$. In some cases, discontinuities of π might still be present after performing the unwrapping process.

4.2.1 Some examples

Let $x[n] = \delta[n]$. The discrete-time Fourier Transform of the unit sample sequence is given by

$$\Delta(e^{j\omega}) = \sum_{n=-\infty}^{\infty} \delta[n]e^{-j\omega n} = \delta[0] = 1.$$

Remarkably, the Fourier Transform of the discrete-time unit sample is a *constant* of value 1 across the entire frequency range. Usually, the shortest the sequence is, the largest the frequency range will be.

Let now $x[n] = \alpha^n \mu[n]$ be the impulse response of a causal exponentially weighted sequence with $|\alpha| < 1$. The Fourier Transform of $x[n]$ is given by

$$\begin{aligned} X(e^{j\omega}) &= \sum_{n=-\infty}^{\infty} \alpha^n \mu[n] e^{-j\omega n} = \sum_{n=0}^{\infty} \alpha^n e^{-j\omega n} \\ &= \sum_{n=0}^{\infty} \left(\alpha e^{-j\omega}\right)^n = \frac{1}{1-\alpha e^{-j\omega}} \end{aligned}$$

and it exists since $\left|\alpha e^{-j\omega}\right| = |\alpha| < 1$. In the case of $|\alpha| \geq 1$ the discrete-time Fourier Transform of $x[n]$ would not exist at all. Figure 4.1 illustrates a plot of the above example for a value of $\alpha = 0.5$.

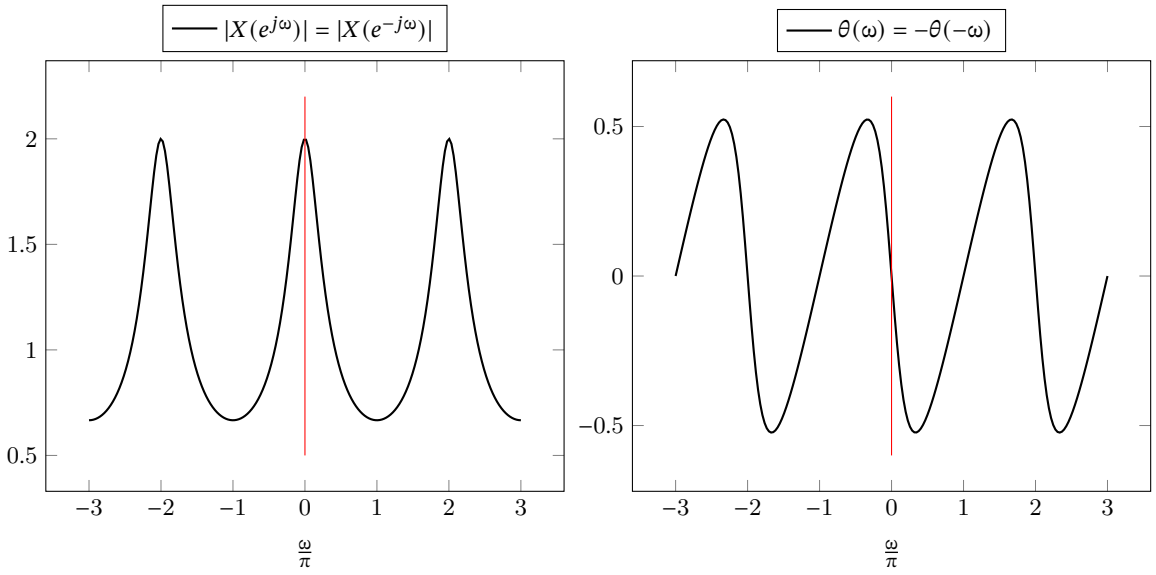


Figure 4.1: Plot of $X(e^{j\omega}) = \frac{1}{1-\alpha e^{-j\omega}}$ with $\alpha = 0.5$. At left, the magnitude spectrum $|X(e^{j\omega})| = \sqrt{1 - 2 \cdot 0.5 \cos\omega + 0.5^2}$, at right, the phase spectrum $\theta(\omega) = \arctan\left(\frac{-0.5 \sin\omega}{1-0.5\cos\omega}\right)$. Notably, the magnitude spectrum is an even function, while the phase spectrum is an odd function. Regarding both functions, it suffices to look what happens in *half* of the principal value of frequency range, that is from 0 to π. This is because the magnitude and phase functions are even–odd functions that can thus be faithfully reconstructed by knowing only half of the values.

A DTFT is a periodic function in ω whose period is 2π. A much interesting way to understand the discrete-time Fourier Transform is to realize that Equation 4.9 represents none other than the *Fourier series representation* of the periodic function X. In fact,

$$\begin{aligned} X(e^{j(\omega_0+2k\pi)}) &= \sum_{n=-\infty}^{\infty} x[n]e^{-j(\omega_0+2k\pi)n} \\ &= \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega_0 n}\underbrace{e^{-j2k\pi n}}_{=1} \\ &= \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega_0 n} \\ &= X(e^{j\omega_0}) \end{aligned}$$

which implies that the discrete-time Fourier Transform manifests the same behavior of a sinusoid among the frequency ω. For this reason, Equation 4.9 represents the Fourier series of the periodic function that is the DTFT. As a result, the Fourier coefficients $x[n]$ can be computed from the transform $X(e^{j\omega})$ by means of the Fourier integral

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}d\omega. \tag{4.11}$$

The Fourier integral not only represents a way to generally obtain the Fourier coefficients of the single waves given a Fourier series representation of a periodic function, but in this specific case it also allows to reconstruct the original sequence $x[n]$ from its discrete-time Fourier Transform, therefore it acts like an *inverse transform*. The following Theorem highlights this property and provides a proof that the Fourier integral of a DTFT is no less than the inverse discrete-time Fourier Transform.

4.2.2 Inverse discrete-time Fourier Transform

Theorem 4.2.1 (Inverse Discrete-time Fourier Transform) *Let $X(e^{j\omega})$ be the discrete-time Fourier Transform of a sequence $x[n]$ so that it exists (the Fourier series related to the DTFT converges uniformly). Then, the* inverse discrete-time Fourier

Transform *formula is*

$$x[n] = \mathcal{F}^{-1}\left[X(e^{j\omega})\right] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}d\omega, \qquad (4.12)$$

the Fourier integral of the discrete-time Fourier Transform.

Proof. Substituting with 4.9 in 4.12, one obtains

$$\mathcal{F}^{-1}\left[X(e^{j\omega})\right] = \frac{1}{2\pi}\int_{-\pi}^{\pi}\left(\sum_{l=-\infty}^{\infty} x[l]e^{-j\omega l}\right)e^{j\omega n}d\omega;$$

provided the summation inside the integral converges uniformly—that is, when the DTFT exists, but it exists by hypothesis—the order of the summation and the integration can be freely exchanged. Hence,

$$\begin{aligned}\frac{1}{2\pi}\int_{-\pi}^{\pi}\left(\sum_{l=-\infty}^{\infty} x[l]e^{-j\omega l}\right)e^{j\omega n}d\omega &= \sum_{l=-\infty}^{\infty} x[l]\left(\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{j\omega(n-l)}d\omega\right)\\ &= \sum_{l=-\infty}^{\infty} x[l]\frac{\sin\pi(n-l)}{\pi(n-l)}\\ &= \sum_{l=-\infty}^{\infty} x[l]\,\mathrm{sinc}\,\pi(n-l).\end{aligned}$$

Now, since

$$\frac{\sin\pi(n-l)}{\pi(n-l)} = \mathrm{sinc}\,\pi(n-l) = \begin{cases} 1 & n = l \\ 0 & n \neq l \end{cases} = \delta[n-l],$$

one soon obtains

$$\sum_{l=-\infty}^{\infty} x[l]\frac{\sin\pi(n-l)}{\pi(n-l)} = \sum_{l=-\infty}^{\infty} x[l]\delta[n-l] = x[n]$$

which corresponds to the thesis. ∎

4.2.3 Convergence conditions for discrete-time Fourier Transforms and Dirac's delta

We have previously said that, in order for a discrete-time Fourier Transform to exist, the Fourier series related to the DTFT must necessarily converge uniformly. Indeed, one might lay emphasis on the *converge condition*, that are the conditions that a discrete-time Fourier Transform has to comply with.

Given a Fourier Transform like in Equation 4.9,

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n},$$

let's build a function X_K that is very similar, but that has a finite sum instead of a series,

$$X_K(e^{j\omega}) = \sum_{n=-K}^{K} x[n]e^{-j\omega n}.$$

From X_K follows the concept of *uniform convergence* of $X(e^{j\omega})$: the discrete-time Fourier Transform $X(e^{j\omega})$ is said to *uniformly converge* if

$$\lim_{K\to\infty}\left|X(e^{j\omega}) - X_K(e^{j\omega})\right| = 0. \qquad (4.13)$$

A sufficient condition for the existence of a discrete-time Fourier Transform is given by the following Theorem,

Theorem 4.2.2 (Existence of discrete-time Fourier Transform)
Let $x[n]$ *be an* absolutely summable *sequence such that*

$$\sum_{n=-\infty}^{\infty}|x[n]| < \infty$$

then the discrete-time Fourier Transform $X(e^{j\omega})$ *of* $x[n]$ *exists for all values of* ω.

Proof. By substitution,

$$\left|X(e^{j\omega})\right| = \left|\sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}\right| \leq \sum_{n=-\infty}^{\infty}|x[n]| < \infty,$$

and it exists as its magnitude is always finite for all values of ω. ∎

The absolute summability of the sequence $x[n]$ is a sufficient condition fo the existence of the discrete-time Fourier Transform $X(e^{j\omega})$. Let's now evaluate the absolute summability of an example sequence.

Let $x[n] = \alpha^n \mu[n]$ for $|\alpha| < 1$. The sequence is absolutely summable, as

$$\sum_{n=-\infty}^{\infty} |\alpha^n| \mu[n] = \sum_{n=-\infty}^{\infty} |\alpha^n| = \frac{1}{1-|\alpha|} < \infty,$$

and its discrete-time Fourier Transform will converge uniformly to $\frac{1}{1-\alpha e^{-j\omega}}$ as a consequence. We have already met this scenario in Section 4.2.1.

Absolutely summable sequences will always possess finite energy. Indeed, this is because

$$\sum_{n=-\infty}^{\infty} |x[n]|^2 \leq \left(\sum_{n=-\infty}^{\infty} |x[n]| \right)^2,$$

an inequality imposing that an absolutely summable sequence always corresponds to a finite energy signal. If a sequence does not have finite energy, it will not be absolutely summable. However, a finite energy sequence might not be necessarily an absolutely summable sequence, as the implication only works in a single sense. As an example of this, consider the sequence defined as

$$x[n] = \begin{cases} \frac{1}{n} & n \geq 1 \\ 0 & n \leq 0 \end{cases};$$

such sequence has finite energy that is equal to

$$\mathcal{E}_x = \sum_{n=1}^{\infty} \left(\frac{1}{n} \right)^2 = \frac{\pi^2}{6}.$$

However, $x[n] = \sum_{n=1}^{\infty} \frac{1}{n}$ is *not* absolutely summable.

To represent a finite energy sequence $x[n]$ that is not absolutely summable by a discrete-time Fourier Transform $X(e^{j\omega})$, it is necessary to consider an integral condition, that is called the **mean-square convergence** of $X(e^{j\omega})$. Such integral condition allows to build a Fourier Transform $X_K(e^{j\omega})$ which despite *not* being the exact discrete-time Fourier Transform of the sequence $x[n]$, it is a good approximation of it. Indeed, the integral condition is expressed in the following equation,

$$\lim_{K\to\infty} \int_{-\pi}^{\pi} \left| X(e^{j\omega}) - X_K(e^{j\omega}) \right|^2 d\omega = 0, \tag{4.14}$$

where

$$X_K(e^{j\omega}) = \sum_{n=-K}^{K} x[n] e^{-j\omega n}. \tag{4.15}$$

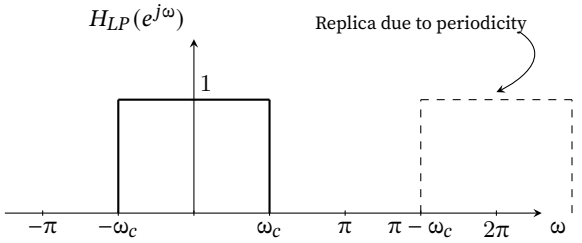
One might already have noticed that the approximate transform has a very similar formula to the 4.9, but now there is a *finite* summation over the time domain. Indeed, this is the reason why the 4.15 is an *approximate* transform of the sequence $x[n]$ and is computable. The underlying reason for all of this is that the sequence $x[n]$ is *not* absolutely summable, therefore it is not possible to represent its transform.

The integral condition 4.14 implies that the total energy of the error[3] quantity $\Xi = X(e^{j\omega}) - X_K(e^{j\omega})$ must approach zero at each value of ω, as $K \to \infty$. In such a case, the absolute value of the error $|\Xi|$ does *not* go to zero as $K \to \infty$, and the discrete-time Fourier Trasnform *may be no longer bounded*. In practice, this means that the resulting Fourier Transform might include a $\delta(\omega)$, a quantity that is indeed not bounded in the continuous domain. To summarise, the mean-square convergence—our integral condition above—basically enforces a rule under which the approximate transform X_K should be formed: the total energy of the error must be close to zero, at each value of ω.

As an example, consider the discrete-time Fourier Transform as follows,

$$H_{LP}(e^{j\omega}) = \begin{cases} 1 & 0 \leq |\omega| \leq \omega_c \\ 0 & \omega_c \leq |\omega| \leq \pi \end{cases}$$

that acts as a *low-pass filter*. Its plot is



[3] Posing a condition over the total energy of the error is a *much less strict* requirement than to pose the same condition over the absolute value of the same quantity.

The inverse discrete-time Fourier Transform is soon obtained by

$$\begin{aligned} h_{LP}[n] &= \int_{-\omega_c}^{\omega_c} e^{j\omega n} d\omega \\ &= \frac{1}{2\pi}\left(\frac{e^{j\omega_c n}}{jn} - \frac{e^{-j\omega_c n}}{jn}\right) \\ &= \frac{\sin \omega_c n}{\pi n} \\ &= \operatorname{sinc} \omega_c n \end{aligned}$$

for all values $-\infty < n < \infty$.

For Parseval's Theorem, the energy of $h_{LP}[n]$ is given by $\frac{\omega_c}{\pi}$. Thus, $h_{LP}[n]$ is a finite-energy sequence, despite being *not absolutely summable*. As a result, the quantity

$$\sum_{n=-K}^{K} h_{LP}[n] e^{-j\omega n} = \sum_{n=-K}^{K} \operatorname{sinc}(\omega_c n) e^{-j\omega n}$$

does not uniformly converge to the Fourier Transform $H_{LP}(e^{j\omega})$ for all ω values; still, it converges to $H_{LP}(e^{j\omega})$ in the *mean-square sense* because its energy is finite.

The mean-square convergence property of the sequence $h_{LP}[n]$ can be further illustrated bu examining the plot of the following function,

$$H_{LP,K}(e^{j\omega}) = \sum_{n=-K}^{K} e^{-j\omega n} \operatorname{sinc} \omega_c n,$$

for many values of K, as in Figure 4.2. A particularly interesting effect the Figure shows is the **Gibbs phenomenon**, a behavior of the DTFT in which multiple ripples—whose energy is smaller as the value of K increases—are present. Verily, Gibbs phenomenon can be described as the oscillatory behavior of the Fourier series of a piecewise continuously differentiable periodic function around a jump discontinuity. Ripples expressed by the Gibbs phenomenon are completely independent of K—the greater the K, the faster the oscillation will be. The number of ripples increases as K increases, with the height of the largest ripple remaining the same for all values of K.

As K goes to infinity, the integral condition

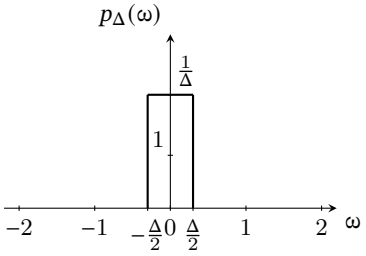
$$\lim_{K\to\infty} \int_{-\pi}^{\pi} \left| X(e^{j\omega}) - X_K(e^{j\omega}) \right|^2 d\omega = 0$$

holds indicating the convergence of $H_{LP,K}(e^{j\omega})$ towards $H_{LP}(e^{j\omega})$. The oscillatory behavior of $H_{LP,K}$ approximating the "real" DTFT in the mean-square sense is due to the Gibbs phenomenon.

The discrete-time Fourier Transform can also be defined for a specific class of sequences, which are neither absolutely summable nor square summable—a class of sequences whose energy is infinite. Examples of such sequences are the *unit step sequence* $\mu[n]$, the sinusoidal sequence $\cos(\omega_0 n + \phi)$ and the exponential sequence $A\alpha^n$. All of these possess infinite total energy and thus are not expressible by means of a "standard" DTFT.

In order to express infinite energy sequences with a discrete-time Fourier Transform, one has to adopt the **Dirac's delta function** $\delta(\omega)$. Dirac delta function is defined such as a function in ω with infinite height, zero-infinitesimal width, and unit area,

$$\lim_{\Delta\to 0} \int_{-\infty}^{\infty} p_\Delta(\omega) d\omega = \int_{-\infty}^{\infty} \delta(\omega) d\omega = 1. \tag{4.16}$$



The Dirac's delta is the limiting form of a unit area pulse function $p_\Delta(\omega)$ as $\Delta \to 0$, satisfying Equation 4.16.

As an exaple, consider the complex exponential sequence

$$x[n] = e^{j\omega_0 n}. \tag{4.17}$$

Its discrete-time Fourier Transform is given by the following

$$X(e^{j\omega}) = \sum_{k=-\infty}^{\infty} 2\pi\delta(\omega - \omega_0 + 2\pi k) \tag{4.18}$$

with $\delta(\omega)$ being the Dirac's delta impulse function and $-\pi \leq \omega_0 \leq \pi$. The function expressed by 4.18 is a periodic function of ω—this can be seen by the $2\pi k$ term in the Dirac's delta argument—which has a peculiar name and is called *periodic impulse train*. To verify that the above DTFT is the

correct one, the best idea would be to simply compute the inverse discrete-time Fourier Transform of $X(e^{j\omega})$. Therefore,

$$\begin{aligned} x[n] &= \frac{1}{2\pi}\int_{-\pi}^{\pi}\sum_{k=-\infty}^{\infty} 2\pi\delta(\omega-\omega_0+2\pi k)e^{j\omega n}d\omega \\ &= \int_{-\pi}^{\pi}\delta(\omega-\omega_0)e^{j\omega n}d\omega = e^{j\omega_0 n}, \end{aligned}$$

where we have employed the sampling property of the impulse function[4] $\delta(\omega)$.

This way, one can define numerous commonly used DTFT pairs, such as the following ones,

$$\begin{aligned} \delta[n] &\longleftrightarrow 1, \\ 1 &\longleftrightarrow \textstyle\sum_{k=-\infty}^{\infty} 2\pi\delta(\omega+2\pi k), \\ 1\cdot e^{j\omega_0 n} &\longleftrightarrow \textstyle\sum_{k=-\infty}^{\infty} 2\pi\delta(\omega-\omega_0+2\pi k), \\ \mu[n] &\longleftrightarrow \frac{1}{1-e^{-j\omega}} + \textstyle\sum_{k=-\infty}^{\infty}\pi\delta(\omega+2\pi k), \\ \alpha^n\mu[n], (|\alpha|<1) &\longleftrightarrow \frac{1}{1-\alpha e^{-j\omega}}. \end{aligned}$$

The last equation, in particular, depends on a parameter α: if $|\alpha| \geq 1$ then a representation like the above cannot be employed, and instead one is forced to employ Dirac's delta impulses.

[4] The sampling property of the impulse function is different regarding of the discrete or continuous domain, and states that

$$\sum_{k=-\infty}^{\infty} e^{j2\pi n}\delta(n-k) = e^{j2\pi k}, \tag{4.19}$$

and

$$\int_{-\infty}^{\infty}\delta(\omega-\omega_0)e^{j\omega} = e^{j\omega_0}. \tag{4.20}$$

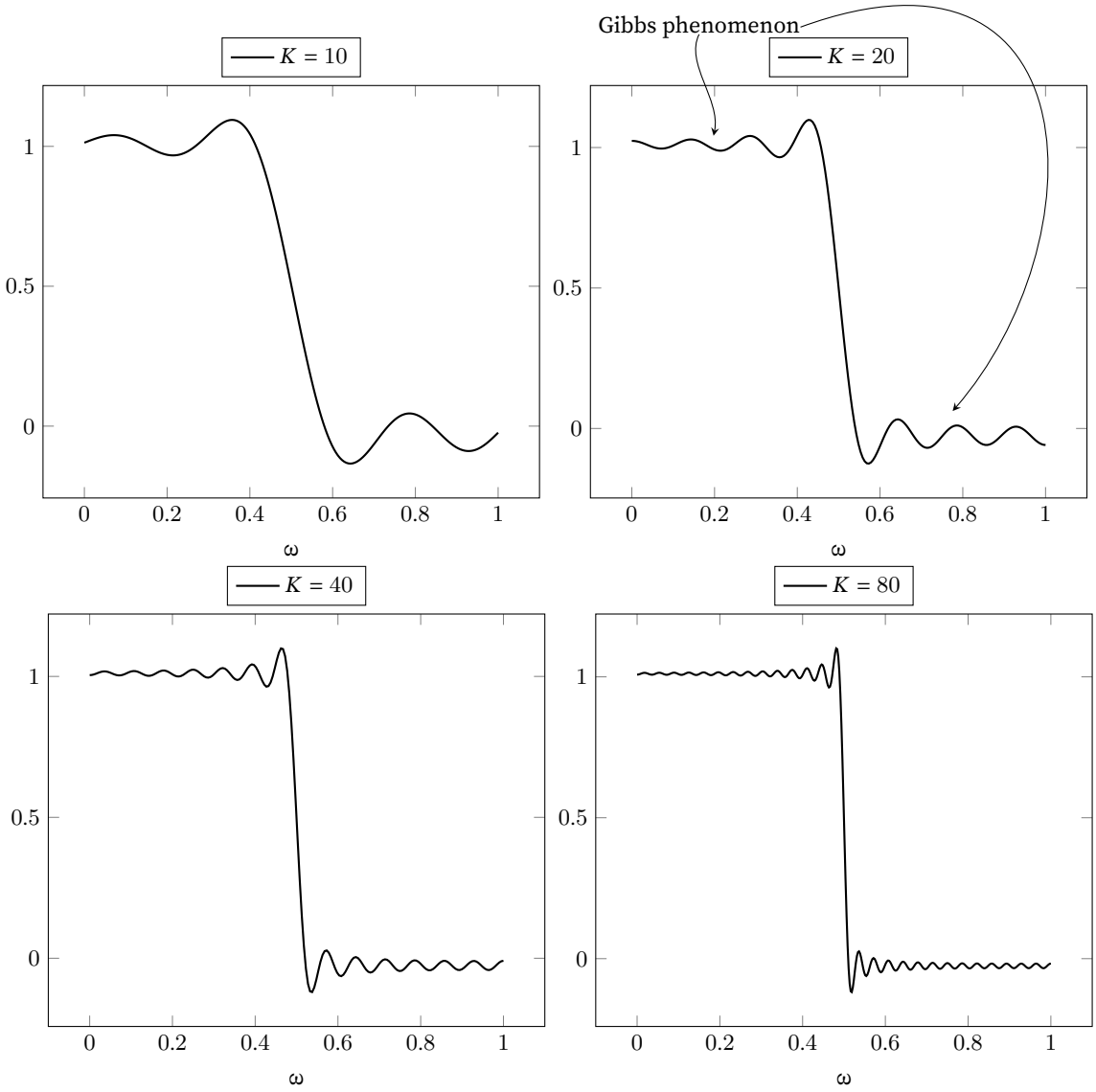


Figure 4.2: Discrete-time Fourier Transform obtained through the mean-square convergence criterion. Noteworthy is the so called *Gibbs phenomenon*, a sinusoidal effect that is always present in the DTFT and is manifest by means of a series of ripples in proximity of the magnitude change. The DTFT approximates an ideal `rect`.

4.2.4 List of noteworthy discrete-time Fourier Transform Theorems and formulas

A number of important properties and theorems, very useful in digital signal processing applications, are listed without proof in Tables 4.1, 4.2 and 4.3.

For instance, suppose to find the DTFT $V(e^{j\omega})$ of a sequence $v[n]$ that follows the relationship

$$d_0 v[n] + d_1 v[n-1] = p_0 \delta[n] + p_1 \delta[n-1].$$

The discrete-time Fourier Transform of the delta impulse is exactly 1—using the time-shifting theorem of the DTFT given by Table 4.3 one finds that the DTFT of $\delta[n-1]$ is $e^{-j\omega}$, and the DTFT of $v[n-1]$ is $e^{-j\omega}V(e^{j\omega})$. Applying the linearity theorem from the same Table, one gets the frequency-domain representation

$$\begin{array}{ccccccc} d_0 v[n] & + & d_1 v[n-1] & = & p_0\delta[n] & + & p_1\delta[n-1] \\ \updownarrow & & \updownarrow & & \updownarrow & & \updownarrow \\ d_0 V(e^{j\omega} & + & d_1 e^{-j\omega} V(e^{j\omega}) & = & p_0 & + & p_1 e^{-j\omega}. \end{array}$$

By solving the above equation, one obtains the final formula

$$V(e^{j\omega}) = \frac{p_0 + p_1 e^{-j\omega}}{d_0 + d_1 e^{-j\omega}}.$$

4.3 Energy Density Spectrum relationshipts in discrete-time Fourier Transforms

Let $g[n]$ be a *finite-energy* sequence. We have already seen that the total energy of $g[n]$ is the quantity

$$\mathcal{E}_x = \sum_{n=-\infty}^{\infty} |g[n]|^2.$$

From the Pareseval's theorem in Table 4.3 one can soon observe that

$$\mathcal{E}_x = \sum_{n=-\infty}^{\infty} |g[n]|^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} \underbrace{\left|G(e^{j\omega})\right|^2}_{S_{gg}(\omega)} d\omega. \tag{4.21}$$

where the quantity $S_{gg}(\omega) = \left|G(e^{j\omega})\right|^2$ is called energy density spectrum, as in Equation 4.6.

The energy density spectrum draws a curve—the area under the curve in the frequency range $-\pi < \omega \leq \pi$ divided by 2π is called **Area Under the Curve**.

Spectrum of signals will differ in relation to their band type.

Since the spectrum of a discrete-time signal is a periodic function of ω having period 2π, a *full-band* signal will possess a spectrum lying in the whole frequency range of $-\pi < \omega \leq \pi$. Vice-versa, a *band-limited* discrete-time signal will possess a spectrum which will be limited to a specific portion of the frequency range $-\pi < \omega \leq \pi$.

Any *ideal band-limited* signal has a spectrum that is zero outside a specific frequency range $[\omega_a, \omega_b]$, that is indeed expressed in Equation 4.8,

$$X(j\omega) = \begin{cases} 0, & 0 \leq |\omega| \leq \omega_a \\ 0, & \omega_b \leq |\omega| \leq \infty \\ X(j\omega) & \omega_a \leq |\omega| \leq \omega_b \end{cases}.$$

In Section 4.1.2 we have already encountered *lowpass*, *highpass* and *bandpass* signals. To recap,

Lowpass signals are all signals arranged in a frequency range $|\omega| \leq \omega_p < \infty$, where ω_p is said to be the *bandwidth* of the signal;

Highpass signals are all signals arranged in a frequency range $0 < \omega_p \leq |\omega| < \infty$, where the bandwitdh of the signal is in the interval $[\omega_p, \infty[$.

Bandpass signals are any signal whose spectrum is located in the frequency range $[\omega_L, \omega_H]$, that is $0 < \omega_L \leq |\omega| \leq \omega_H < \infty$, where the bandwidth is given by $\mathrm{BW} = \omega_H - \omega_L$.

Consider as an example the sequence

$$x[n] = (0.5)^n \mu[n];$$

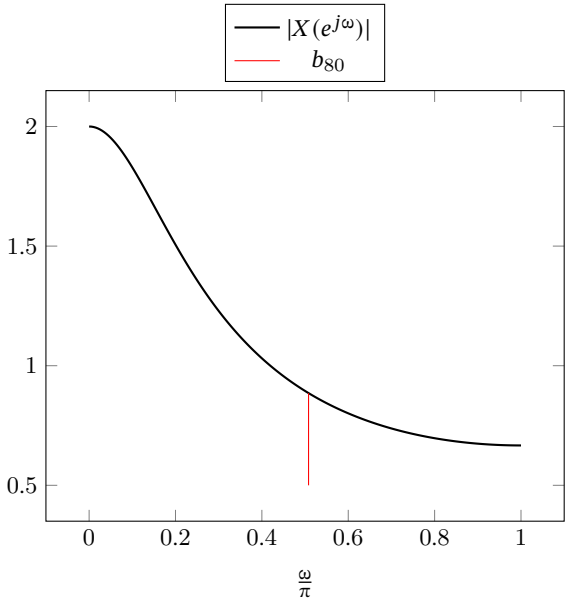
its discrete-time Fourier Transform is given below on the left along with its *magnitude spectrum* shown below.

Sequence	**Discrete-time Fourier Transform**
$x[n]$ (a complex sequence)	$X(e^{j\omega})$
$x[-n]$	$X(e^{-j\omega})$
$x^*[n]$	$X^*(e^{j\omega})$
$\Re\{x[n]\}$	$X_{cs}(e^{j\omega}) = \frac{1}{2}\{X(e^{j\omega}) + X^*(e^{-j\omega})\}$
$j\Im\{x[n]\}$	$X_{ca}(e^{j\omega}) = \frac{1}{2}\{X(e^{j\omega}) - X^*(e^{-j\omega})\}$
$x_{cs}[n]$	$X_{re}(e^{j\omega})$
$x_{ca}[n]$	$jX_{im}(e^{j\omega})$

Table 4.1: Notable discrete-time Fourier Transform properties. $X_{cs}(e^{j\omega})$ and $X_{ca}(e^{j\omega})$ are the conjugate-symmetric and conjugate-antisymmetric parts of $X(e^{j\omega})$, respectively. Likewise, $x_{cs}[n]$ and $x_{ca}[n]$ are, respectively, the conjugate-symmetric and conjugate-antisymmetric parts of $x[n]$.

Sequence	**Discrete-time Fourier Transform**
$x[n]$	$X(e^{j\omega}) = X_{re}(e^{j\omega}) + jX_{im}(e^{j\omega})$
$x_{ev}[n]$	$X_{re}(e^{j\omega})$
$x_{od}[n]$	$jX_{im}(e^{j\omega})$
Symmetry relations	$X(e^{j\omega}) = X^*(e^{-j\omega})$
—	$X_{re}(e^{j\omega}) = X_{re}(e^{-j\omega})$
—	$X_{im}(e^{j\omega}) = -X_{im}(e^{-j\omega})$
—	$\lvert X(e^{j\omega})\rvert = \lvert X(e^{-j\omega})\rvert$
—	$\arg X(e^{j\omega}) = -\arg X(e^{-j\omega})$

Table 4.2: Notable discrete-time Fourier Transform properties. Sequences $x_{ev}[n]$ and $x_{od}[n]$ are, respectively, the even and the odd parts of the sequence $x[n]$.



It can be shown that the 80% of the energy of the lowpass signal in the above example is contained in a very precise frequency range, that is $0 \leq |\omega| \leq 0.5081\pi$. Therefore, we can define the 80% *bandwidth frequency* as the frequency $b_{80} = 0.5081\pi$.

In another example one has the sequence

$$h_{LP}[n] = \frac{\sin \omega_c n}{\pi n}.$$

In order to compute the total energy one uses the Parseval's theorem as follows,

$$h_{LP}[n] = \sum_{n=-\infty}^{\infty} |h_{LP}[n]|^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left|H_{LP}(e^{j\omega})\right|^2 d\omega,$$

Name	Sequence	Discrete-time Fourier Transform
Linearity	$\alpha g[n] + \beta h[n]$	$\alpha G(e^{j\omega}) + \beta H(e^{j\omega})$
Time-shifting	$g[n - n_0]$	$e^{-j\omega n_0} G(e^{j\omega})$
Frequency-shifting	$e^{j\omega_0 n} g[n]$	$G\left((e^{j(\omega-\omega_0)}\right)$
Frequency differentiation	$ng[n]$	$j\frac{dG(e^{j\omega})}{d\omega}$
Convolution	$g[n] \circledast h[n]$	$G(e^{j\omega}) \cdot H(e^{j\omega})$
Modulation	$g[n] \cdot h[n]$	$\frac{1}{2\pi}\int_{-\pi}^{\pi} G(e^{j\theta}) \cdot H(e^{j(\omega-\theta)})d\theta$
Parseval's relation	$\sum_{n=-\infty}^{\infty} g[n]h^*[n]$	$= \frac{1}{2\pi}\int_{-\pi}^{\pi} G(e^{j\omega})H^*(e^{j\omega})d\omega$

Table 4.3: Notable discrete-time Fourier Transform properties.

where the already known discrete-time Fourier Transform $H_{LP}(e^{j\omega})$ is the function

$$H_{LP}(e^{j\omega}) = \begin{cases} 1 & 0 \le |\omega| \le \omega_c \\ 0 & \omega_c \le |\omega| \le \pi \end{cases}.$$

By applying the latter in the Parseval's theorem equation, one finally gets

$$h_{LP}[n] = \sum_{n=-\infty}^{\infty} |h_{LP}[n]|^2 = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} d\omega = \frac{\omega_c}{\pi} < \infty,$$

showing that the sequence $h_{LP}[n]$ is a lowpass sequence possessing finite energy.

In MATLAB, the function `freqz` can be invoked to compute the values of the discrete-time Fourier Transform of a sequence, described as a ragional function having the form

$$X(e^{j\omega}) = \frac{p_0 + p_1 e^{-j\omega} + \cdots + p_m e^{-j\omega m}}{d_0 + d_1 e^{-j\omega} + \cdots + d_n e^{-j\omega n}}, \tag{4.22}$$

at a prescribed set of discrete frequency points $\omega = \omega_l$, dense enough to *look* continuous[5].

The statement

```
H = freqz(num, den, w);
```

instructs MATLAB to return the frequency response values of a discrete-time Fourier Transform defined in terms of the vectors `num` and `den`, containing—respectively—the coefficients $\{p_i\}$ and $\{d_i\}$, at a prescribed set of frequencies between 0 and 2π given by the vector `w`. The result is returned as a vector `H`.

As an example, let's plot the real and imaginary parts along with the magnitude and phase spectra of the following discrete-time Fourier Transform,

$$X(e^{j\omega}) = \frac{0.008 - 0.033e^{-j\omega} + 0.05e^{-j2\omega} - 0.033e^{-j3\omega} + 0.008e^{-j4\omega}}{1 + 2.37e^{-j\omega} + 2.7e^{-j2\omega} + 1.6e^{-j3\omega} + 0.4e^{-j4\omega}} \tag{4.23}$$

First off, let's define the proper variables,

```
pkg load signal;
num = [0.008 -0.033 0.05 -0.033 0.008];
den = [1 2.37 2.7 1.6 0.4];
w = [0:0.01:3.14];
```

and then compute the DTFT as we have seen,

```
H = freqz(num, den, w);
```

A quick plot, for instance, of the real part can be quickly produced with

```
plot(w, real(H));
```

however, plots as in Figure 4.3 have been generated with the `subplot` command,

```
set(groot,'defaultLineLineWidth',2.0)
figure(1);
subplot(2, 2, 1); plot(w, real(H));
title("Real part of H");
subplot(2, 2, 2); plot(w, imag(H));
title("Imaginary part of H");
subplot(2, 2, 3); plot(w, abs(H));
title("Magnitude spectrum of H");
subplot(2, 2, 4); plot(w, angle(H));
```

[5] Indeed, no calculator can possibly compute all the infinite points of the continuous curve, and can only determine the values of the transform for a finite number of them.

```
title("Phase spectrum of H");
```

which were enough to generate the four-plot grid.

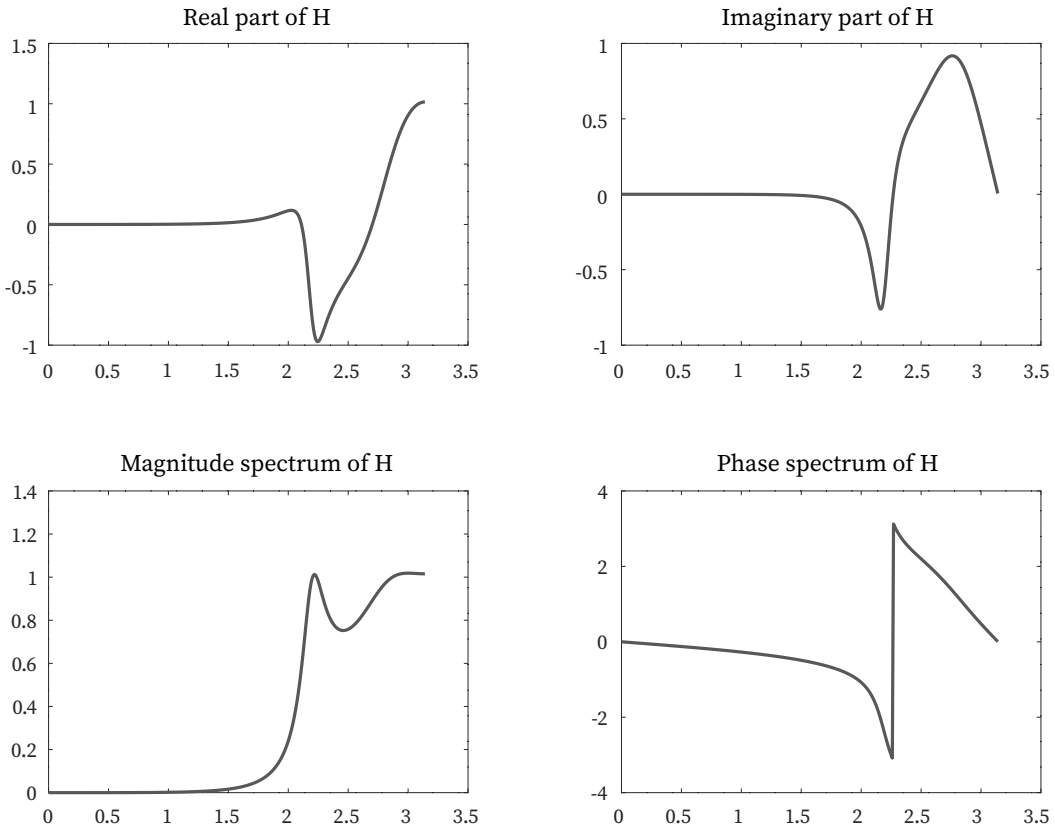


Figure 4.3: *Real part, imaginary part, magnitude spectrum* and *phase spectrum* plots of Example 4.23. Notice the discontinuity in the $\arctan$ function that manifests with a sudden jump in the phase spectrum. Plots have been generated with the help of `octave`.

4.3.1 The Unwrapped Phase Function

In numerical computation when the computed phase function is outside of the possible range of ω—that is $[-\pi, \pi]$—to bring the computed value to the correct range the phase is computed *modulo* 2π. Hence, the phase may exhibit some discontinuities of 2π radians in the phase spectrum plot, as in Figure 4.3. Indeed, one would like to possess some kind of representation which doesn't present any kind of visible discontinuity; to do so, one should adopt the **unwrapped phase function**.

Unwrapped phase functions are special functions whose goal is to represent the phase without any discontinuity—that is, by *unwrapping the phase*. Related function goes under the name of $\theta_c(\omega)$. In the following Figure 4.4 the unwrapped phase spectrum has been plotted with the commands

```
plot(w,unwrap(angle(H)));
```

which are enough to produce a plot of the desired unwrapped phase function of `H`.

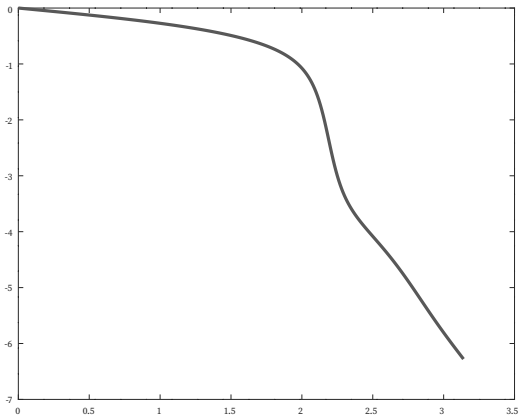


Figure 4.4: *Unwrapped* phase spectrum plot of phase spectrum of Example 4.23. By employing `unwrap` function in `octave`, the discontinuity in the $\arctan$ function had disappeared.

Undeniably, the unwrapped representation remains imperative when plotting phase spectra.

4.3.2 The Convolution Theorem

The discrete-time Fourier transform (DTFT) is a form of Fourier analysis that is applicable to a sequence of values.

The DTFT is often used to analyze samples of a continuous function. The term discrete-time refers to the fact that the transform operates on discrete data, often samples whose interval has units of time. From uniformly spaced samples it produces a function of frequency that is a periodic summation of the continuous Fourier transform of the original continuous function. Under certain theoretical conditions, described by the sampling theorem, the original continuous function can be recovered perfectly from the DTFT and thus from the original discrete samples. The DTFT itself is a continuous function of frequency, but discrete samples of it can be readily calculated via the discrete Fourier transform (DFT), which is by far the most common method of modern Fourier analysis.

Indeed, a paramount property that surely helps when it comes to performing convolutions on the time-domain is the following, remarkable property, that goes under the name of **Convolution Theorem**.

Theorem 4.3.1 (Convolution Theorem)
Consider two signals $g(x)$ and $h(x)$ whose Fourier Transforms are $G(\omega)$ and $H(\omega)$, and let $\mathcal{F}\{\cdot\}$ denote the operator of Fourier Transformation. The convolution of g and h is defined by

$$g \circledast h = \mathcal{F}^{-1}\{G \cdot H\}, \tag{4.24}$$

where a convolution sum in the time-domain corresponds to a product *in the frequency domain.*

The Convolution Theorem states that any convolution sum can be performed as a product in the frequency domain. This also means that no matter the signals' domain, jumping into the related frequency domain and performing a product of the Fourier Transforms of the two signals is a straightforward way to perform a convolution sum. An immediate application of this crucial result is that the linear convolution $y[n]$ of $x[n]$ and $h[n]$ in a linear discrete-time system can be performed *much* more quickly by going for

the frequency-domain route. The steps of this faster process are the following ones,

1. first, one computes the discrete-time Fourier Transforms of x and h, that are $X(e^{j\omega})$ and $H(e^{j\omega})$;
2. second, from the frequency-domain one quickly evaluates $Y(e^{j\omega}) = X(e^{j\omega}) \cdot H(e^{j\omega})$;
3. and at last, to obtain $y[n]$ it is enough to compute the inverse DTFT. The overall process is depicted in Figure 4.5.

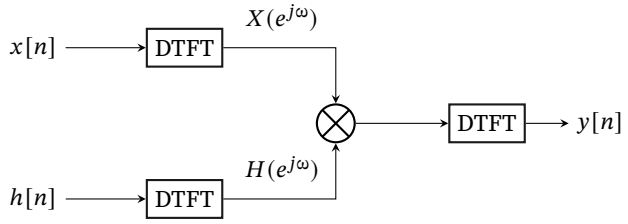


Figure 4.5: Linear convolution as a product in frequency domain. In reality, the process is not performed, as it is far from convenient to handle continuous signals.

The aforementioned procedure is not feasible in reality for two reasons: first of all, it is never convenient—let alone possible—to effectively handle *continuous* signals. Indeed, no machine can possibly compute all the infinite points required for the computation. Secondarily and not less importantly, the inverse discrete-time Fourier Transform is an *integral*. Performing integrations on a continuous is far from efficient for the same reasons as above.

With the Convolution Theorem ends the chapter on Discrete-time Fourier Transforms. Indeed, as we will see, the Convolution Theorem will prove useful even for the next subject of study, the *Discrete Fourier Transforms*.

Chapter 5

Discrete Fourier Transform

In mathematics, the discrete Fourier transform (DFT) converts a finite sequence of equally-spaced samples of a function into a same-length sequence of equally-spaced samples of the discrete-time Fourier transform (DTFT), which is a complex-valued function of frequency. The interval at which the DTFT is sampled is the reciprocal of the duration of the input sequence. An inverse DFT is a Fourier series, using the DTFT samples as coefficients of complex sinusoids at the corresponding DTFT frequencies. It has the same sample-values as the original input sequence. The DFT is therefore said to be a frequency domain representation of the original input sequence. If the original sequence spans all the non-zero values of a function, its DTFT is continuous (and periodic), and the DFT provides discrete samples of one cycle. If the original sequence is one cycle of a periodic function, the DFT provides all the non-zero values of one DTFT cycle.

The DFT is the most important discrete transform, used to perform Fourier analysis in many practical applications. In digital signal processing, the function is any quantity or signal that varies over time, such as the pressure of a sound wave, a radio signal, or daily temperature readings, sampled over a finite time interval (often defined by a window function). In image processing, the samples can be the values of pixels along a row or column of a raster image. The DFT is also used to efficiently solve partial differential equations, and to perform other operations such as convolutions or multiplying large integers.

Since it deals with a **finite** amount of data, it can be implemented in computers by numerical algorithms or even dedicated hardware. These implementations usually employ efficient fast Fourier transform (FFT) algorithms; so much so that the terms "FFT" and "DFT" are often used interchangeably. Prior to its current usage, the "FFT" initialism may have also been used for the ambiguous term "finite Fourier transform" [6].

5.1 Orthogonal Transforms

Let $x[n]$ be a sequence of length N. We define $\mathcal{X}[k], 0 \leq k \leq N-1$ the N-point **orthogonal transform** of $x[n]$. An orthogonal transforms is a *generalized* version of Transforms that takes the form

$$\mathcal{X}[k] = \sum_{n=0}^{N-1} x[n]\psi^*[k,n], \quad 0 \leq k \leq N-1, \tag{5.1}$$

$$x[k] = \sum_{k=0}^{N-1} \mathcal{X}[n]\psi[k,n], \quad 0 \leq n \leq N-1. \tag{5.2}$$

The first Equation 5.2 is called *Analysis equation*, while the second Equation 5.2 is said to be the *Synthesis equation*. Both analysis and synthesis equations concur in the definition of the orthogonal transform. Indeed, the analysis equation allows to perform the orthogonal transformation, while implicitly the synthesis equation serves as a "inverse" orthogonal transform. Still, the ψ are said to be *basis sequences* and are of length N as well as the original sequence and the orthogonal transform sequence. Basis sequences can have complex values, so generally $\psi[k,n] \in \mathbb{C}$. In the class of transforms to be considered in this notes of digital signal processing, all the basis sequences will satisfy the following condition,

$$\frac{1}{N}\sum_{n=0}^{N-1} \psi[k,n]\psi^*[l,n] = \begin{cases} 1, & l = k \\ 0, & l \neq k \end{cases}, \tag{5.3}$$

a condition that substantially poses that the product

$$\psi[k,n]\psi^*[l,n]$$

is equal to 1 if and only if indexes $l = k$, a property that if satisfied then the basis sequences are said to be *orthogonal to each other*.

Really, in order for ψ^* to be the inverse orthogonal transform, ψ and ψ^* have to be orthogonal to each other. This can be verified by substituting the synthesis equation 5.2 into the analysis equation 5.2,

$$\begin{aligned}\sum_{n=0}^{N-1} x[n]\psi^*[k,n] &= \sum_{n=0}^{N-1}\left(\frac{1}{N}\sum_{k=0}^{N-1}\mathcal{X}[k]\psi[n,k]\right)\psi^*[l,n]\\ &= \sum_{k=0}^{N-1}\mathcal{X}[k]\left(\frac{1}{N}\sum_{n=0}^{N-1}\psi[n,k]\psi^*[l,n]\right)\\ &= \mathcal{X}[l],\end{aligned}$$

which yields an identity. Of course, the other way around works exactly the same, hence the inverse orthogonal transform is expressed as in Equation 5.2.

An important consequence of the orthogonality condition for orthogonal transforms is the **Energy Preservation Property**, which states that

$$\sum_{n=0}^{N-1}|x[n]|^2 = \frac{1}{N}\sum_{k=0}^{N-1}|\mathcal{X}[n]|^2 \tag{5.4}$$

Yet, the Energy Preservation Property is more commonly known as the *Parseval's Theorem,* of which it is a more general formulation.

5.2 Definition of Discrete Fourier Transform

It's now the time to introduce the **Discrete Fourier Transform (DFT)**, a tool of a paramount importance in the field of digital signal processing.

Definition 5.2.1 (Discrete Fourier Transform)**.** Let $x[n]$ be a finite-length sequence of length N defined for $0 \le n \le N-1$ and be $X(e^{j\omega})$ its discrete-time Fourier Transform as in Equation 4.9. By *uniformly sampling* $X(e^{j\omega})$ on the ω axis in the interval $[0, 2\pi[$ at samples $\omega_k = \frac{2k\pi}{N}$ for $0 \le k \le N-1$ one obtains a sequence $X[k]$ such that

$$X[k] = X(e^{j\omega})\Big|_{\omega=\frac{2k\pi}{N}} = \sum_{n=0}^{N-1} x[n]e^{-2j\pi\frac{kn}{N}}, 0 \le k \le N-1 \tag{5.5}$$

where $X[k]$ is a sequence of length N in the frequency domain represented by the variable k. The sequence $X[k]$ is called the *Discrete Fourier Transform (DFT)* of the sequence.

The above definition has been obtained by sampling the ω frequency variable in samples $2k\pi/N$ so that the complex continuous exponential $e^{-j\omega}$ has become $e^{-2j\pi\frac{kn}{N}}$. Usually, in order to streamline the notation, the substitution

$$W_N = e^{-j\frac{2\pi}{N}} \tag{5.6}$$

under which the Discrete Fourier Transform now is rewritten into the following,

$$X[k] = \sum_{n=0}^{N-1} x[n]W_N^{kn}, 0 \le k \le N-1. \tag{5.7}$$

The notation expressed in 5.6 renders the orthogonal transform operator W_N in Discrete Fourier Transform sum have a positive sign in its exponents. This behavior is the opposite of what one encounters by looking at the original definition.

Along with the DFT, the **Inverse Discrete Fourier Transform** should be defined as well in order to produce meaningful computations. Indeed,

$$x[k] = \sum_{k=0}^{N-1} X[k]W_N^{-kn}, 0 \le n \le N-1, \tag{5.8}$$

expressed under the new notation[1]. Verifying the above inverse transform is straightforward, as it is enough to substitute the original transform into the inverse, then multiply both sides of the equation by W_N^{ln} and sum the result from

[1]The original Discrete Fourier Transform—without employing the new notation—would be

$$x[k] = \sum_{k=0}^{N-1} X[k]e^{2j\pi\frac{kn}{N}}, 0 \le n \le N-1,$$

showing a positive sign at the exponentiation.

$n = 0$ to $n = N - 1$. Of course,

$$\begin{aligned}\sum_{n=0}^{N-1} x[n]W_N^{ln} &= \sum_{n=0}^{N-1}\left(\frac{1}{N}\sum_{k=0}^{K-1} X[k]W_N^{-kn}\right)W_n^{ln} \\ &= \frac{1}{N}\sum_{n=0}^{N-1}\sum_{k=0}^{K-1} X[k]W_N^{-(k-l)n} \\ &= \frac{1}{N}\sum_{k=0}^{K-1} X[k]\left(\sum_{n=0}^{N-1} W_N^{-(k-l)n}\right).\end{aligned}$$

Now, since

$$\sum_{n=0}^{N-1} W_N^{-(k-l)n} = \begin{cases} N, & k-l = rN, r \in \mathbb{Z} \\ 0, & \text{otherwise}\end{cases} \tag{5.9}$$

due to the exponentiation properties. In order to verify that, just substitute with the definition of W_N, that is

$$\sum_{n=0}^{N-1} e^{+2j\pi\frac{(k-l)n}{N}},$$

a quantity that indeed is equal to N if and only if $k - l$ is a multiple of N and 0 otherwise. The term relative to $n = 0$ is the sum of N terms all equal to 1, while the term for $n \neq 0$ is related to the first N terms of a geometric progression, that are

$$\sum_{n=0}^{N-1} \rho^n = \frac{1-\rho^N}{1-\rho} = 0$$

as the exponential $e^{j2\pi\frac{N}{N}} = 1$. Another way to see it is to consider that a set of W_N vectors is summed together—when $k \neq l$ for each vector of phase ϕ another vector of opposite phase $-\phi$ is summed, and the result of the sum is exactly 0; on contrary, the only way to avoid the cancellation of terms is when $k = l$, that is when all terms have magnitude equal to $|e^{j2\pi\frac{N}{N}}| = 1$ and possess phase $\phi = 0$. For this reason, one obtains that the only non-zero term in the sum is $\sum_{n=0}^{N-1} W_N^{-(k-l)n}, k = l, 0 \leq l, k \leq N-1$. Hence,

$$\begin{aligned}\sum_{n=0}^{N-1} x[n]W_N^{ln} &= \frac{1}{N}\sum_{k=0}^{K-1} X[k]\left(\sum_{n=0}^{N-1} W_N^{-(k-l)n}\right). \\ &= \frac{1}{N}X[l]N = X[l].\end{aligned}$$

The first example in exam is a DFT of the following sequence of length N,

$$x[n] = \begin{cases} 1 & n = 0 \\ 0 & 1 \leq n \leq N-1\end{cases}$$

that is quite similar to the delta impulse. Its N-point Discrete Fourier Transform is given by

$$X[k] = \sum_{n=0}^{N-1} x[n]W_N^{kn} = x[0]W_N^{k0} = 1, 1 \leq k \leq N-1.$$

The second example is of a sequence

$$y[n] = \begin{cases} 1 & n = m \\ 0 & 0 \leq n \leq m-1, m+1 \leq n \leq N-1\end{cases}$$

which is like the previous example, but translated of m. Its Discrete Fourier Transform will be

$$Y[k] = \sum_{n=0}^{N-1} y[n]W_N^{kn} = y[m]W_N^{km} = W_N^{km}, 1 \leq k \leq N-1.$$

Let now $g[n] = \cos 2\pi\frac{rn}{N}$ be a sequence of length N. Since

$$\begin{aligned}g[n] &= \frac{1}{2}\left(e^{j2\pi\frac{rn}{N}} + e^{-j2\pi\frac{rn}{N}}\right) \\ &= \frac{1}{2}\left(W_N^{rn} + W_N^{rn}\right),\end{aligned}$$

and due to linearity the Discrete Fourier Transform will be the transform of the two addends. Therefore,

$$\begin{aligned}G[k] &= \sum_{n=0}^{N-1} g[n]W_N^{kn} \\ &= \frac{1}{2}\left(\sum_{n=0}^{N-1} W_N^{-(r-k)n} + \sum_{n=0}^{N-1} W_N^{-(r+k)n}\right)\end{aligned}$$

and by means of Identity 5.9 one finally obtains that

$$G[k] = \begin{cases} \frac{N}{2} & k = r \\ \frac{N}{2} & k = N-r \\ 0 & \text{otherwise}\end{cases}$$

The DFT of a cosine wave is a pair of impulses of amplitude $\frac{N}{2}$, located at samples r and $N - r$ that represent *the*

same frequency. Conceptually then, a cosine DFT is a pair of impulses just like in the case of the Discrete-time Fourier Transform.

5.2.1 Matrix relations

Discrete Fourier Transform as expressed in 5.7 can be easily expressed into a matrix form. Let $\boldsymbol{x}$ and $\boldsymbol{X}$ be two column vectors as defined,

$$\boldsymbol{x} = \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} X[0] \\ X[1] \\ \vdots \\ X[N-1] \end{bmatrix}$$

the *Matrix form of the Discrete Fourier Transform* is

$$\boldsymbol{X} = \boldsymbol{D}_N \boldsymbol{x} \tag{5.10}$$

with $\boldsymbol{D}_N$ the **DFT Matrix**, a $N \times N$ matrix given by

$$\boldsymbol{D}_N = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & W_N^2 & \cdots & W_N^{(N-1)} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{(N-1)} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)^2} \end{bmatrix} \tag{5.11}$$

Conversely, the *Matrix form of the Inverse Discrete Fourier Transform* is given by a similar relationship,

$$\boldsymbol{x} = \boldsymbol{D}_N^{-1} \boldsymbol{X} \tag{5.12}$$

with $\boldsymbol{D}_N$ the **Inverse DFT Matrix**, a $N \times N$ matrix given by

$$\boldsymbol{D}_N^{-1} = \frac{1}{N} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^{-1} & W_N^{-2} & \cdots & W_N^{-(N-1)} \\ 1 & W_N^{-2} & W_N^{-4} & \cdots & W_N^{-2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{-(N-1)} & W_N^{-2(N-1)} & \cdots & W_N^{-(N-1)^2} \end{bmatrix} \tag{5.13}$$

Remarkably, the following relationship holds between the inverse and the straight DFT Matrix,

$$\boldsymbol{D}_N = \frac{1}{N} \boldsymbol{D}_N^{-1},$$

with a factor of $\frac{1}{N}$ between the one and the other.

Matrix relations are crucial to learn how to manipulate Discrete Fourier Transforms in MATLAB. In fact, the two functions `fft` and `ifft`—which stand for *Fast Fourier Transform* and *Inverse Fast Fourier Transform*—manipulate column vector data structures. These two functions make use of sophisticated techniques and efficient algorithms based on the "dividi et impera" concept to optimize the computation of the DFT.

Let's study the DFT of the cosine $\cos 2\pi r \frac{n}{N}$. The following `octave` code computes and plots the DFT of such function, first by setting $r = 3$, then with $r = 3.3$.

```
N = 21;
M = 21;
r = 3; n = 0:N-1;  u = cos(2*pi*r*n/N);
U = fft(u,M);
figure(1);
subplot(2,3,1);   n = 0:1:N-1;   stem(n,u);
title('Original time-domain sequence');
xlabel('Time index n'); ylabel('Amplitude')
subplot(2,3,2);   k = 0:1:M-1;   stem(k,abs(U));
title('Magnitude, r = 3');
xlabel('Frequency index k'); ylabel('Magnitude')
subplot(2,3,3);   stem(k,angle(U));
title('Phase, r = 3');
xlabel('Frequency index k'); ylabel('Phase')

r = 3.3; n = 0:N-1;  u = cos(2*pi*r*n/N);
U = fft(u,M);
subplot(2,3,4);   n = 0:1:N-1;   stem(n,u);
title('Original time-domain sequence');
xlabel('Time index n'); ylabel('Amplitude')
subplot(2,3,5);   k = 0:1:M-1;   stem(k,abs(U));
title('Magnitude, r = 3.3');
xlabel('Frequency index k'); ylabel('Magnitude')
subplot(2,3,6);   stem(k,angle(U));
title('Phase, r = 3.3');
xlabel('Frequency index k'); ylabel('Phase')
```

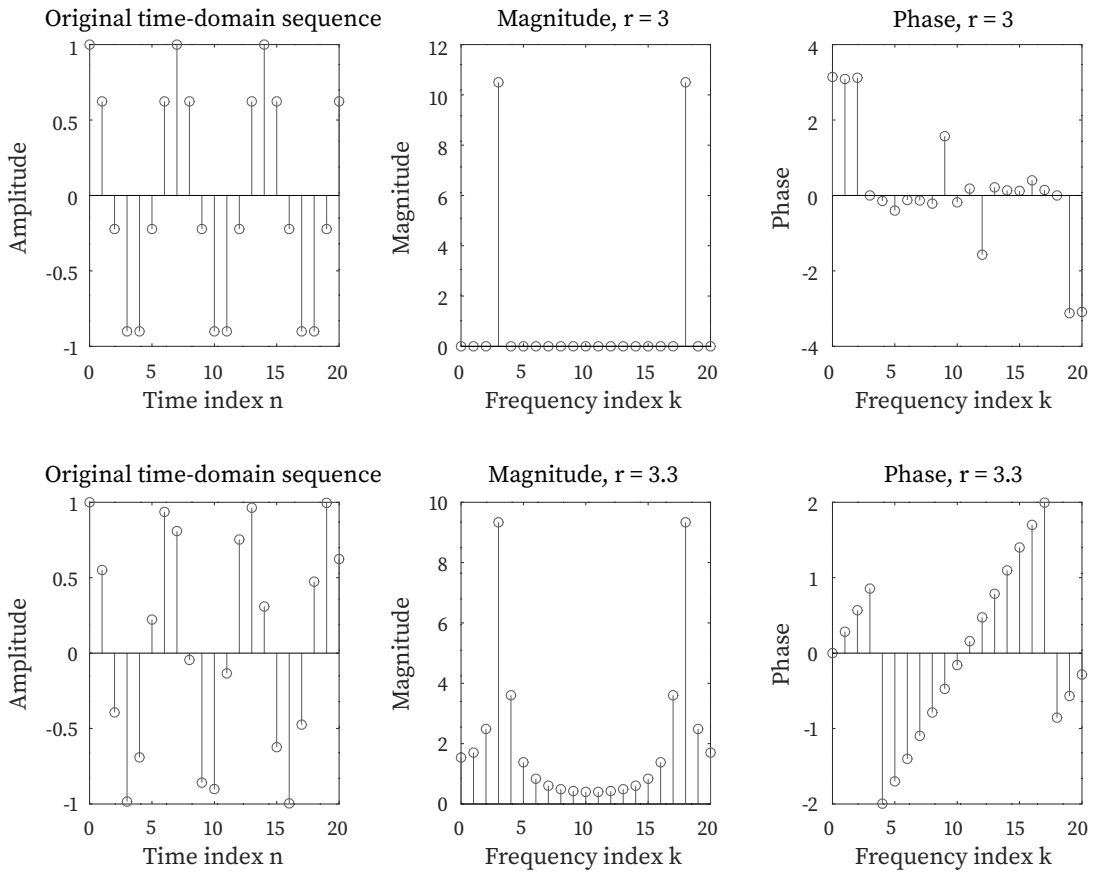


Figure 5.1: Plots of original signal (left), magnitude (center) and phase (right) of the Discrete Fourier Transform of sequence $\cos 2\pi r \frac{n}{N}$. Notice how two perfect impulses are generated if and only if the quantity r is an integer.

The Discrete Fourier Transform of a cosine wave will produce a peculiar *pair of impulses* located at the very same frequency of the sinusoid, provided the multiplying factor of n is a *multiple of* 2π—that is, when $r \in \mathbb{Z}$. In all other cases, imperfect impulses will be generated, a phenomenon that is quite visible in Figure 5.1 for the case of $r = 3.3$.

The DFT of the delta impulse can be produced by running the following `octave` code, that will first generate three pictures like those in Figure 5.1 related to a $\delta[n]$ sequence, then it will do the same with a delayed delta impulse,

```
N = 21;
M = 21;
u = [1 zeros(1,N-1)];
U = fft(u,M);
figure(1);
subplot(2,3,1);   n = 0:1:N-1;   stem(n,u);
title('Impulse in zero');
xlabel('Time index n'); ylabel('Amplitude')
subplot(2,3,2);   k = 0:1:M-1;   stem(k,abs(U));
title('Magnitude');
xlabel('Frequency index k'); ylabel('Magnitude')
subplot(2,3,3);   stem(k,angle(U));
title('Phase');
xlabel('Frequency index k'); ylabel('Phase')

u = [zeros(1,4) 1 zeros(1,N-5)];
U = fft(u,M);
```

```
subplot(2,3,4);   n = 0:1:N-1;   stem(n,u);
title('Delayed impulse');
xlabel('Time index n'); ylabel('Amplitude')
subplot(2,3,5);   k = 0:1:M-1;   stem(k,abs(U));
title('Magnitude');
xlabel('Frequency index k'); ylabel('Magnitude')
subplot(2,3,6);   stem(k,angle(U));
title('Phase');
xlabel('Frequency index k'); ylabel('Phase')
```

5.2.2 Sampling process of a Discrete Fourier Transform

Consider a sequence $x[n]$ whose Discrete-time Fourier Transform is $X(e^{j\omega})$. In order to obtain the Discrete Fourier Transform, one has to sample the DTFT at equally spaced points $\omega_k = \frac{2k\pi}{N}$ with $0 \leq k \leq N-1$, developing the frequency samples $\{X(e^{j\omega_k})\}$—these frequency samples can be considered the N-point DFT $Y[k]$ whose N-point IDFT is a sequence $y[n]$ of length N. This is because sampling a Discrete-time Fourier Transform will result in a Discrete Fourier Transform *provided* the sampling is applied in N points. That way,

$$\{X(e^{j\omega_k})\} = Y[k].$$

Now, since the Discrete-time Fourier Transform is, like in Equation 4.12

$$X(e^{j\omega}) = \sum_{l=-\infty}^{\infty} x[l]e^{-j\omega l};$$

this leads to

$$\begin{aligned} Y[k] = X(e^{j\omega_k}) &= X\left(e^{j2\pi\frac{k}{N}}\right) \\ &= \sum_{l=-\infty}^{\infty} x[l]e^{-j2\pi\frac{kl}{N}} \\ &= \sum_{l=-\infty}^{\infty} x[l]W_N^{kl}. \end{aligned}$$

The trick now is to compute the IDFT of $y[n]$ and substitute the just obtained expression for $Y[k]$. That is,

$$\begin{aligned} y[n] &= \frac{1}{N}\sum_{k=0}^{N-1} Y[k]W_N^{-kn} \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\sum_{l=-\infty}^{\infty} x[l]W_N^{kl}W_N^{-kn} \\ &= \sum_{l=-\infty}^{\infty}\left(\frac{1}{N}\sum_{k=0}^{N-1} x[l]W_N^{-k(n-l)}\right); \end{aligned}$$

by means of the following identity

$$\sum_{n=0}^{N-1} W_N^{-k(n-l)} = \begin{cases} N, & l = n + mN, \\ 0, & \text{otherwise} \end{cases} \tag{5.14}$$

which is analogous of 5.9, one soon obtains the desired relation

$$y[n] = \sum_{m=-\infty}^{\infty} x[n+mN], 0 \leq n \leq N-1. \tag{5.15}$$

This last result, expressed by Equation 5.15 implies that $y[n]$ is obtained from the original sequence $x[n]$ by adding an *infinite number of shifted replicas* of it, with each replica shifted by an integer multiple of N sampling instants and observing the sum only for the interval $0 \leq n \leq N-1$. It has been obtained by picking an original sequence $x[n]$, performing its Discrete-time Fourier Transform $X(e^{j\omega})$, sampling it in N points $\omega_k = \frac{2k\pi}{N}$ with $0 \leq k \leq N-1$ and finally obtained by rearranging and substituting on formulas. The integer number m determines the amount of shift any replica of $x[n]$ is subjected to—value $m = 0$ corresponds to the original signal, value $m = 1$ corresponds to a shift of N samples, $m = 2$ yields a replica shifted by $2N$, and so on.

The concept of infinite replicas is related to the fact that, in order to compute the DFT of a signal, one conceptually has to first make the sequence $x[n]$ *periodic* by adding infinite replicas of the very same signal to create a sequence $y[n] = \sum_{m=-\infty}^{\infty} x[n+mN]$, with $0 \leq n \leq N-1$, and then the DFT can be computed.

When applying Equation 5.15 to finite-length sequences, one assumes that the samples outside the specified range are all zeros (like in zero-padding). Hence, if $x[n]$ is a sequence of length M with $M \leq N$, then the sequence $y[n] = x[n]$ for all

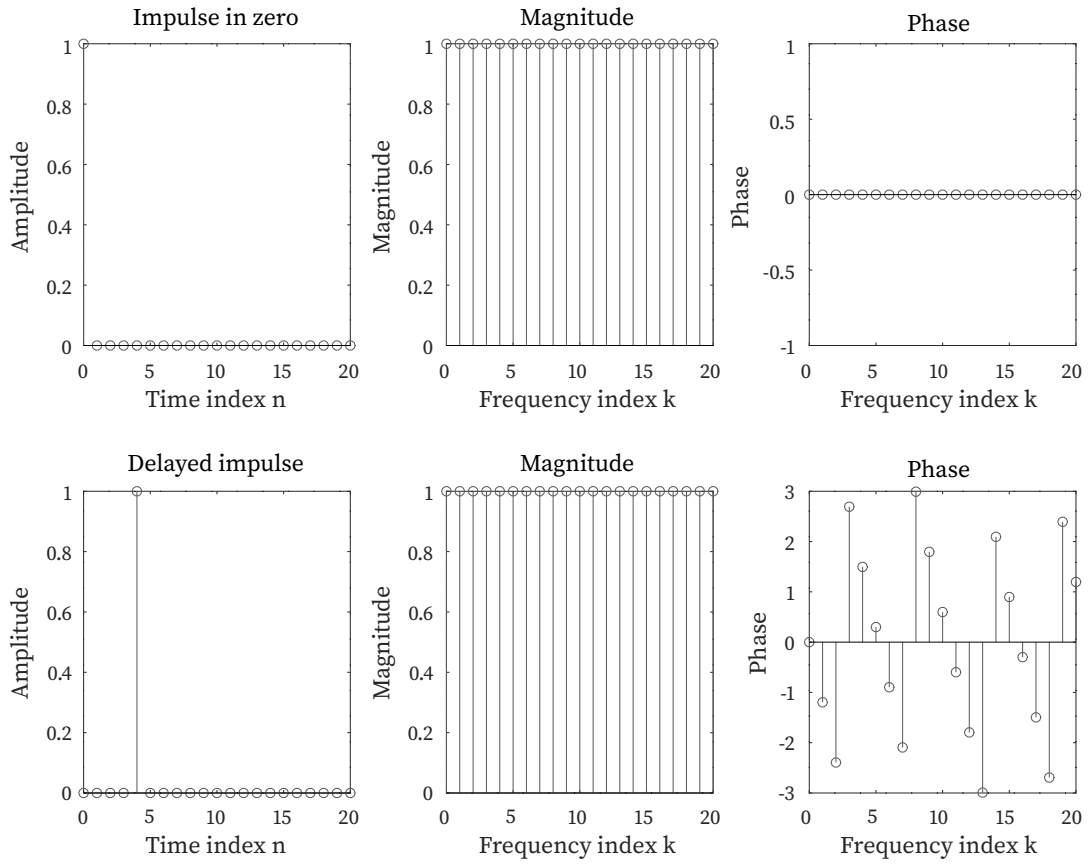


Figure 5.2: Plots of original signal (left), magnitude (center) and phase (right) of the Discrete Fourier Transform of sequences $\delta[n]$ and $\delta[n-4]$. The magnitude of the DFT is a constant as the transform of a delta will be a constant.

values $0 \leq n \leq N-1$. This behavior results from the fact that N samples are more than what is needed (M samples are needed to describe the original sequence $x[n]$). On contrary, in the opposite case of $M > N$ there is a so called *time-domain aliasing* of samples of the sequence $x[n]$ in generating $y[n]$, with the end result that $x[n]$ cannot be recovered anymore from $y[n]$.

Suppose one has the sequence

$$\{x[n]\} = \left\{ \underset{\uparrow}{0} \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \right\}$$

of length $M = 6$. By sampling its Discrete-time Fourier Transform $X(e^{j\omega})$ at samples $\omega_k = \frac{2\pi k}{4}, 0 \leq k \leq 3$ and then applying a 4-point Discrete Fourier Transform to these samples, one gets the sequence $y[n]$

$$y[n] = x[n] + x[n+4] + x[n-4], 0 \leq n \leq 3$$

that is

$$\{y[n]\} = \left\{ \underset{\uparrow}{4} \quad 6 \quad 2 \quad 3 \right\}.$$

It is clear that from this sequence it is impossible to recover the original sequence $x[n]$—this behavior is induced by the choice of $N = 4$, less than what would be required $N \leq M = 6$ in order to recover the original sequence.

Let now

$$\{x[n]\} = \begin{Bmatrix} 3 & 4 & 5 & 6 & 7 \end{Bmatrix},$$

a sequence of length $M = 5$ and compute $y[n]$ for values of $N = 5$, $N = 7$ and $N = 3$. One obtains that,

- for $N = 5$, the resulting sequence

$$\{y[n]\} = \underbrace{\begin{Bmatrix} 3 & 4 & 5 & 6 & 7 \end{Bmatrix}}_{m=0} + \underbrace{\begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \end{Bmatrix}}_{m=1} + \underbrace{\begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \end{Bmatrix}}_{m=2} + \dots,$$

  showing that the original sequence can be fully recovered from $m = 0$;

- for $N = 7$ the behavior is quite similar—resulting sequence will be

$$\{y[n]\} = \underbrace{\begin{Bmatrix} 3 & 4 & 5 & 6 & 7 & 0 & 0 \end{Bmatrix}}_{m=0} + \underbrace{\begin{Bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{Bmatrix}}_{m=1} + \underbrace{\begin{Bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{Bmatrix}}_{m=2} + \dots,$$

  and again, the original sequence can be fully recovered from $m = 0$;

- for $N = 3$ the behavior drastically change, as the $N = 3$ samples are insufficient against the length of $x[n]$ which is M. Indeed, one obtains

$$\{y[n]\} = \underbrace{\begin{Bmatrix} 3 & 4 & 5 \end{Bmatrix}}_{m=0} + \underbrace{\begin{Bmatrix} 6 & 7 & 0 \end{Bmatrix}}_{m=1} + \underbrace{\begin{Bmatrix} 0 & 0 & 0 \end{Bmatrix}}_{m=2} + \dots = \begin{Bmatrix} 9 & 11 & 5 \end{Bmatrix},$$

  which implies aliasing as the end result is different from the original sequence $x[n]$.

The case relative to $N = 7$ suggests that a Discrete Fourier Transform with *higher sample rate* can be obtained by transforming a zero-padded $x[n]$. This is because they are basically equivalent as choosing $N = 7, M = 7$ with a zero-padded signal $x[n]$ in which the length has been increased from $M = 5$ to $M = 7$ by adding zeros to the right.

5.2.3 Discrete-time Fourier Transform by interpolation of a DFT

As a brief recap, an N-point Discrete Fourier Transform of a sequence $x[n]$ of length N is quite simply the frequency samples of its Discrete-time Fourier Transform $X(e^{j\omega})$, evaluated at N uniformly spaced points in frequency that are

$$\omega_k = 2\pi \frac{k}{N}, 0 \le k \le N-1.$$

In the above case one can obtain the DFT from the DTFT by sampling it at some points; however, the contrary can be performed. In fact, an *approximated* version of a Discrete-time Fourier Transform can be obtained by *interpolation* of the Discrete Fourier Transform. This practical approach allows one to get an approximated version of the DTFT of a finite-length sequence by computing the DFT first, choosing a proper interpolation function, and finally interpolating the DFT to obtain the continuous-time Fourier Transform.

Numerical computation

Let $X(e^{j\omega})$ be the DTFT of a sequence $x[n]$ of length N—one wants to evaluate the DTFT at a dense grid of M frequencies[2] $\omega_k = 2\pi \frac{k}{M}, 0 \le k \le M-1$ where the quantity $M \gg N$. In formulas,

$$\begin{aligned} X(e^{j\omega}) &= \sum_{n=0}^{N-1} x[n] e^{-j\omega_k n} \\ &= \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{kn}{M}}. \end{aligned}$$

Now, the original sequence of length N; since $M \gg N$, in order to avoid aliasing phenomena, one has to define a new *extended sequence* $x_e[n]$ whose length is M and it is extended

[2] The dense grid of frequencies is one of the ways in which a computation of a continuous-domain function can occur with calculators. Since it is impossible to compute all the infinite points in a real number interval, it is often enough to compute a great number of them so that the computed function appears almost continuous.

by zero-padding from sample N to sample $M-1$:

$$x_e[n] = \begin{cases} x[n] & 0 \leq n \leq N-1 \\ 0 & N \leq n \leq M-1 \end{cases}$$

Thanks to this extension of the original sequence, the Discrete-time Fourier Transform can be expressed as

$$X(e^{j\omega}) = \sum_{n=0}^{M-1} x_e[n] e^{-j2\pi \frac{kn}{M}}.$$

The DTFT $X(e^{j\omega})$ can essentially be thought as an M-point DFT $X_e[k]$ of the extended sequence $x_e[n]$ of length M. The latter DFT $X_e[k]$ can be computed ahead using the *Fast Fourier Transform* algorithm if M is conveniently chosen as an integer of power 2, such as $M = 2^K, K \in N^+$.

On MATLAB code, the function `freqz` employs the above approach to evaluate the frequency response at a prescribed set of frequencies of a DTFT, expressed as a rational function in $e^{-j\omega}$.

5.2.4 Periodicity of the DFT and the IDFT

Due to the periodic nature of the complex exponential, both the Discrete Fourier Transform and the Inverse Discrete Fourier Transform are *periodic sequences*. Indeed,

$$\begin{aligned} X[k+N] &:= \sum_{n=0}^{N-1} x_n e^{-j2\pi \frac{(k+N)n}{N}} \\ &= \sum_{n=0}^{N-1} x_n e^{-j2\pi \frac{kn}{N}} \underbrace{e^{-j2\pi n}}_{=1} \\ &= \sum_{n=0}^{N-1} x_n e^{-j2\pi \frac{kn}{N}} \\ &= X[k]. \end{aligned}$$

and the inverse holds

$$\begin{aligned} x[n+N] &= \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi \frac{k(n+N)}{N}} \\ &= \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi \frac{kn}{N}} \underbrace{e^{j2\pi k}}_{=1} \\ &= \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi \frac{kn}{N}} \\ &= x[n] \end{aligned}$$

showing off that both DFT and IDFT are periodic sequences of period N.

The above property is just the tip of the iceberg of a deeper rule, which is known as the **circular time-shifting property** of Discrete Fourier Transforms. We will encounter it later on.

5.3 Overview of Fourier Transforms classes

In general, Fourier Transforms can be divided in multiple classes. The following list indexes all possible cases related to various class of signals and their Fourier Transform.

- **The sequence is continuous and aperiodic in the time domain**—then its Fourier Transform will be *continuous and aperiodic* as well. In practice, this means that, given a signal $x_a(t)$, the Fourier Transform $X_a(f)$ will be

  $$X_a(f) = \int_{-\infty}^{\infty} x_a(t) e^{-j2\pi ft} dt, \tag{5.16}$$

  with its inverse transform of the form

  $$x_a(t) = \int_{-\infty}^{\infty} X_a(f) e^{+j2\pi ft} df. \tag{5.17}$$

  The very same can be said of *signals whose period is protracted for a very long time*, that is all signals that can be thought as "almost aperiodic" in the sense that they are remarkably far-reaching.

- **The sequence is continuous and periodic in the time domain**—then, its Fourier Transform will be *discrete and aperiodic*. Its aperiodicity is inherited from the fact that the original signal is *continuous*, just like in the previ-

ous case. Indeed, the formulas will be

$$X[k] = \frac{1}{T_p} \int_{T_p} x_a(t) e^{-j2\pi k f_0 t} dt, \tag{5.18}$$

where the integration occurs in a portion of the signal that is the period T_p; its inverse transform will be of the form

$$x_a(t) = \sum_{k=-\infty}^{\infty} X[k] e^{+j2\pi k f_0 t}. \tag{5.19}$$

The value of f_0 is exactly the inverse of the length of the period T_p, that is $f_0 = \frac{1}{T_p}$. Veritably, in the case of signals whose period is far-reaching—that is, when T_p is a lofty quantity—the frequency $f_0 \to 0$ as T_p approaches infinity. Indeed, such transforms will resemble much and much more the continuous ones we find in the case of continuous aperiodic signals.

- **The sequence is discrete and aperiodic in the time domain**—then, its Fourier Transform is *continuous and periodic*. This is the case of the Discrete-time Fourier Transform as in Equation 4.9, a ccase that is conceptually the dual of the *discrete and aperiodic sequences* that lead to continuous and periodic transforms. In truth, in this case the formulas are actually the reverse, as

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}, \tag{5.20}$$

$$x[n] = \frac{1}{2\pi} \int_{2\pi} X(\omega) e^{+j\omega n} d\omega. \tag{5.21}$$

Noteworthily, the period for the Discrete-time Fourier Transform will be 2π, as seen in 4.10; undeniably, the integration is performed in a single period of 2π.

- **The sequence is discrete and periodic in the time domain**—then, its Fourier Transform is *discrete and periodic* as well. Undoubtedly, we are again facing the Discrete Fourier Transform as in Equation 5.5. The formulas will for sure be

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{kn}{N}}, \tag{5.22}$$

$$x[n] = \sum_{k=0}^{N-1} X[n] e^{+j2\pi \frac{kn}{N}}, \tag{5.23}$$

As already seen, there's a brawny relationship between the Discrete-time Fourier Transform and the Discrete Fourier Transform. As a matter of fact, the DFTF can be obtained by a DFT from an interpolation procedure; whilst the DFT can be obtained by sampling a DTFT.

To summarize concisely,

1. from **continuous** signals one can only retrieve **aperiodic** transforms;
2. **discrete** signals will yield **periodic** transforms;
3. **aperiodic** signals will tally to **continuous** transforms;
4. **periodic** signals will generate **discrete** transforms;

If one pays adequate attention, can notice that there's a pattern comprised in the above rules, that is the following Table 5.1,

signal	$\longleftrightarrow$	**transform**
continuous	$\longleftrightarrow$	aperiodic
discrete	$\longleftrightarrow$	periodic
aperiodic	$\longleftrightarrow$	continuous
periodic	$\longleftrightarrow$	discrete

Table 5.1: Rules for retrieving the nature of the Fourier Transform based off the nature of the original signal–sequence.

Essentially, to determine the kind of Fourier Transform it is enough to look at the nature of the original signal–sequence and apply the above rules expressed in Table 5.1. For instance, if x is a *discrete aperiodic* sequence, its Fourier Transform X will be *periodic continuous*. Analogously, let x be a *periodic discrete* sequence; its Fourier Transform X will be *discrete periodic* as well by applying the very same rules.

5.4 Properties of Discrete Fourier Transforms

Discrete Fourier Transforms share a lot of properties with the Discrete-time Fourier Transforms, its continuous counterpart. Some of these properties are essentially identical to those of the DTFT, while some others are different. A summary of the Discrete Fourier Transforms properties are given by the following Tables 5.2, 5.3 and 5.4. All properties refer to sequences of length N.

Sequence	Discrete Fourier Transform
$x[n]$	$X[k]$
$x^*[\langle -n \rangle_N]$	$X^*[k]$
$\Re\{x[n]\}$	$X_{pcs}[k] = \frac{1}{2}\{X[\langle k \rangle_N] + X^*[\langle -k \rangle_N]\}$
$j\Im\{x[n]\}$	$X_{pca}[k] = \frac{1}{2}\{X[\langle k \rangle_N] - X^*[\langle -k \rangle_N]\}$
$x_{pcs}[n]$	$\Re X[k]$
$x_{pca}[n]$	$j\Im X[k]$

Table 5.2: Notable Discrete Fourier Transform properties. $x_{pcs}[k]$ and $X_{pca}[k]$ are the conjugate-symmetric and conjugate-antisymmetric parts of $x[k]$, respectively. Likewise, $X_{pcs}[k]$ and $X_{pca}[k]$ are, respectively, the conjugate-symmetric and conjugate-antisymmetric parts of $X[k]$. The expression $\langle -k \rangle_N$ denotes the *modulo expression*—the sequence is periodic of period N. X is periodic and of infinite length.

Sequence	Discrete Fourier Transform
$x[n]$	$X[k] = \Re X[k] + j\Im X[k]$
$x_{pe}[n]$	$\Re X[k]$
$x_{po}[n]$	$j\Im X[k]$
Symmetry relations	$X[k] = X^*[\langle -k \rangle_N]$
—	$\Re X[k] = \Re X[\langle -l \rangle_N]$
—	$\Im X[k] = -\Im X[\langle -l \rangle_N]$
—	$\lvert X[k] \rvert = \lvert X[\langle -l \rangle_N] \rvert$
—	$\arg X[k] = -\arg X[\langle -l \rangle_N]$

Table 5.3: Notable Discrete Fourier Transform properties. Sequences $x_{pe}[n]$ and $x_{po}[n]$ are, respectively, the even and the odd parts of the sequence $x[n]$. In the above table, $x[n]$ is a real sequence.

5.5 Circular operations on finite-length sequences

Finite-length sequences have distinctive traits for which a special care is required. In particular, common operations such as time-shifting, time-reversal and convolutions have to be conceived diversely. I'm referring in particular to all operations which *modify* the domain of the original signal—let's say, $0 \leq n \leq N-1$—by producing a result whose domain is *different* from $0 \leq n \leq N-1$. Since the Discrete Fourier Transform is periodic with the *same* period of the original signal, one wants to avoid to modify the domain of the resulting signal with any operation.

Contemplate a length-N sequence $x[n]$, defined for samples $0 \leq n \leq N-1$. Its sample values are equal to zero for all samples before 0 and after $N-1$, that is for $n < 0 \wedge n \leq N$. A *time-reversal* operation on $x[n]$ would result in a sequence $x[-n]$ of length N, that is defined for time instants $-(N-1) \leq n \leq 0$. Furthermore, a *linear time-shift* of $x[n]$ by a quantity $M \in \mathbb{Z}$ would result in a sequence $x[n+M]$ of length N, no longer defined for $0 \leq n \leq N-1$, but for $M \leq n \leq N-1+M$. Correspondingly, a *convolution sum* of two sequences of length N defined for $0 \leq n \leq N-1$ would result in a sequence of length $2N+1$, defined for $0 \leq n \leq 2N-2$—such that it is longer than the original sequences.

Yet, we would like to preserve both the *exact length* of the sequence and *its location in time instants*. For instance, we would like the output of operations to persist in the range $0 \leq n \leq N-1$. By this logic, the need to define new types of operations emerges—in particular, new operations analogous of time-reversal, time-shifting and convolution sum with the resultant sequences in the same range of the input sequences. The key to solve this issue are the **circular operations**.

Name	Sequence	Discrete Fourier Transform
Linearity	$\alpha g[n] + \beta h[n]$	$\alpha G[k] + \beta H[k]$
Circular time-shifting	$g[\langle n - n_0 \rangle_N]$	$W_N^{kn_0} G[k]$
Circular frequency-shifting	$W_N^{-kn_0} g[n]$	$G[\langle k - k_0 \rangle_N]$
Duality	$G[n]$	$Ng[\langle -k \rangle_N]$
Circular N-point convolution	$\sum_{m=0}^{N-1} g[m]h[\langle n - m \rangle_N]$	$G[k] \cdot H[k]$
Modulation	$g[n] \cdot h[n]$	$\frac{1}{N} \sum : m = 0^{N-1} G[k] \cdot H[\langle k - m \rangle_N]$
Parseval's relation	$\sum_{n=-\infty}^{\infty} g[n]h^*[n]$	$= \frac{1}{2\pi} \int_{-\pi}^{\pi} G(e^{j\omega}) H^*(e^{j\omega}) d\omega$

Table 5.4: Notable Discrete Fourier Transform properties. Notice how the properties—set side by side to those of Table 4.3 are now *circular* properties that make use of the modulo operation.

5.5.1 Modulo operation

The time-reversal operation on a finite-length sequence is obtained by means of the **modulo operation**. Let $0, 1, \ldots, N-1$ be a set of N positive integers and let $m \in \mathbb{Z}$ be any integer. The integer r obtained by evaluating

$$r = \langle m \rangle_N = m \mod N \tag{5.24}$$

is called the **residue**. The residue r is an integer whose value is between 0 and $N-1$ (as the modulo operation yields). We will denote the modulo operation with the notation

$$\langle m \rangle_N = m \mod N.$$

Due to modulo properties, if $r = \langle m \rangle_N$ then $r = m + lN$, where $l \in \mathbb{Z}$ such that $0 \leq m + lN \leq N - 1$. As an example, let $N = 7$ and $m = 25$: one has

$$r = 25 + 7l = 25 - 7 \times 3 = 4$$

which means $\langle 25 \rangle_7 = 4$. Another example is $N = 7$ and $m = -15$, for which

$$r = -15 + 7l = -15 + 7 \times 3 = 6$$

and yields $\langle -15 \rangle_7 = 6$.

5.5.2 Circular time-reversal operation

Another operation we are going to find a substitute for is the *time-reversal*—in practice, we are going to build the **circular time-reversal operation**.

Circular time-reversals grab the original sequence, and invert the order of the sequence by preserving the time instant at position 0. The circular time-reversal $\{y[n]\}$ of $\{x[n]\}$ is a new sequence of length N as well, thus defined for $0 \leq n \leq N - 1$, given by the formula

$$\{y[n]\} = \{x[\langle -n \rangle_N]\}. \tag{5.25}$$

As an example, consider the sequence in Figure 5.3, where the original sequence is said to be *circularly reversed*.

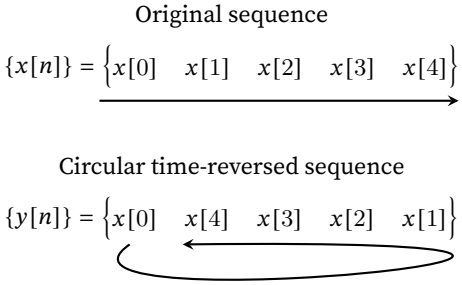


Figure 5.3: The *circular time-reversal* technique. The output sequence is a reversed original copy, except for the sample at the origin which remains in place.

Basically, the circular time-reversal is a special kind of time-reversal in which the "position" from 0 to $N - 1$ of the original sequence is preserved, whilst its samples are almost totally reversed, with the sole exception of the sample at the origin. A sequence like

$$-1 \quad 7 \quad 5 \quad 9 \quad 4 \quad -12$$

the moment it is exposed to the circular time-reversal operation is remodeled into the sequence

$$-1 \quad -12 \quad 4 \quad 9 \quad 5 \quad 7\cdot$$

5.5.3 Circular shift operation

As formerly mentioned, the time shifting operation has to be modified as well to fit our Discrete Fourier Transform signal processing. Sure thing, we might introduce the **circular shift** operation, defined using the modulo operation as in 5.24 as well.

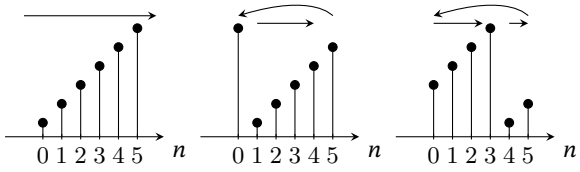
Let $x[n]$ be a sequence possessing length N, defined for $0 \leq n \leq N-1$—the *circularly shifted by n_0 samples* version $x_c[n]$ is given by

$$x_c[n] = x[\langle n - n_0 \rangle_N], \tag{5.26}$$

with $x_c[n]$ also being a length-N sequence defined for the same time instants $0 \leq n \leq N-1$ as the original sequence. For $n_0 > 0$ one performs a *right circular shift*—the above equation implies that the new sequence is defined as such

$$x_c[n] = \begin{cases} x[n - n_0] & n_0 \leq n \leq N-1 \\ x[N - n - n_0] & 0 \leq n < n_0 \end{cases}$$

Figuring out the implications of the above system can be tricky—Figure 5.4 helps with an illustration of what happens when a signal is circularly-shifted



Original	$x[\langle n-1 \rangle_6]$	$x[\langle n-4 \rangle_6]$
$x[n]$	$x[\langle n+5 \rangle_6]$	$x[\langle n+2 \rangle_6]$

Figure 5.4: Circular shift operation on a ramp sequence of length $N = 6$. As the samples are pushed to the right, they "come back" from the left, realizing a circular pattern. Of course, the original sequence $x[n]$ is equal to a sequence $x[\langle n-6 \rangle_N] = x[n]$ that is circularly shifted of a quantity that is the same of its entire length, thanks to the modulo operation.

As can be seen from Figure 5.4, a right circular shift from n_0 is completely equivalent to a left circular shift by $N - n_0$ sample periods. As soon as we are performing a shift by means of the modulo operation, any circular shift by a quantity $n_g > N$ will be completely equivalent to a circular shift by $\langle n_g \rangle_N$, as any number greater than N will be exposed to the modulo operation as in 5.24, causing the result to be still less than N. For instance, suppose one has a length-11 signal and wants to perform a circular shift by $n_g = 67$. What takes effect is that such shift will be completely equivalent to a shift $n_0 = \langle n_g \rangle_N$, that is

$$n_0 = \langle n_g \rangle_N = \langle 67 \rangle_{11} = 67 \mod 11 = 1$$

completely comparable to a simple unit shift.

5.5.4 Circular padding operation

Yet another operation should be modified—the zero-padding operation, which would elseways alter the length of the original signal. In point of fact, the zero-padding operation *actually means* altering the length of the original signal by adding a bunch of zeros to the left or to the right. Such an operation cannot deal proficiently in relation to the Discrete Fourier Transforms, since as we have earlier said the favored approach would be to preserve the characteristics of the original signal.

The implicit periodicity of the sequences manipulated with the Discrete Fourier Transform requires to substitute the *zero-padding* operation with a *circular padding* variant. This circular padding variant will reconstruct and remodel our input signal by periodicizing it, an operation that will pad our signal with *as many replicas of the original sequance as needed* instead of padding with zeros. This means that instead of performing

$$\begin{Bmatrix} 1 & 2 & 3 & 4 \end{Bmatrix} \rightarrow \begin{Bmatrix} \cdots & 0 & 0 & 1 & 2 & 3 & 4 & 0 & 0 & \cdots \end{Bmatrix}$$

one adds multiple replicas of the very same signal as from input. The operation is illustrated in Figure 5.5.

This way, one does not append an infinite number of zeros, but instead appends an infinite number of signal replicas before and after the original sequence. Of course, the resulting signal will virtually be of infinite length—still, it will be a periodic sequence.

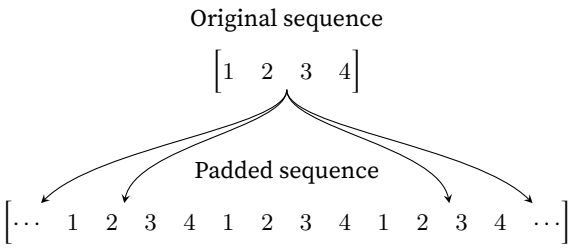


Figure 5.5: The *circular padding* technique. Pay attention at how the pattern $\begin{bmatrix}1 & 2 & 3 & 4\end{bmatrix}$ is repeated indefinitely, to construct an infinite-length signal which repeats indefinitely with a period of the same length of the original signal—in this situation, $N = 4$.

The circular padding acts very well with the Discrete Fourier Transforms. When we have examined the Discrete Fourier Transform we figured out that the N-point Inverse Discrete Fourier Transform $y[n]$ as expressed by Equation 5.15 was akin to the original N-length sequence $x[n]$, but composed of *infinite replicas* of the original sequence. Regardless of the fact that the DFT was attained with its original formula as in 5.5 or from sampling the Discrete-time Fourier Transform, as a matter of fact the inverse DFT is an infinite-length sequence which is assembled from the original sampling by adding infinite replicas of $x[n]$ at a distance N each other, an operation that is fully equivalent to applying infinite times circular padding to the original sequence.

As we just modified three basic operations of signal processing to adapt to the Discrete Fourier Transform, we approach the adjustment of our last operation, the *circular convolution*.

5.5.5 Circular Convolution

Circular convolution is akin to linear convolution, with the paramount difference that in place of convolving two sequences in the old fashion by first time-reversing and then shifting one of the sequences from $-\infty$ to $+\infty$ one *circularly reverses* and then *circularly shifts* a sequence over another—a profound discrepancy that will be soon outlined in extra detail.

Let $g[n]$ and $h[n]$ be two sequences each possessing length N. Their linear convolution—the standard convolution as seen in 2.19—results in a new sequence $y_L[n]$ of length $2N-1$ given by

$$y_L[n] = \sum_{l=0}^{N-1} g[l]h[n-l], 0 \leq n \leq 2N-2.$$

When computing y_L the underlying assumption was that both input sequences were zero-padded to extend their lengths to match the output's length $2N-1$. The longer duration of the output $y_L[n]$ results from the time-reversal of the sequence $h[n]$ and its linear shift from left to right, as the convolution sum requires to do. A new sequence of "double-length less one" is created with the linear approach.

The first nonzero value of $y_L[n]$ is $y_L[0] = g[0]h[0]$—that is when the two signals first "rendezvous" during the shift from left to right of sequence $h[n-l]$—and the last nonzero value is $y_L[2N-2] = g[N-1]h[N-1]$—just before the $h[n-l]$ sequence "departs" from the steady sequence $g[l]$. Further samples will all yield 0, as there is no more superposition between sequences $g[l]$ and $h[n-l]$.

As we have earlier mentioned, this is a poor way to perform the convolution as the output sequence will have a *different* period than input sequences, an undesired behavior. From there arises the need of developing a new convolution-like operation, whose result is still a length-N sequence $y_C[n]$ alike the input sequences. The new operation is called the **Circular convolution** and is cleverly designed as a convolution whose *time-reversal* and *time-shifting* operations are replaced by their *circular* counterparts by means of the modulo operation:

$$y_C[n] = \sum_{l=0}^{N-1} g[l]h[\langle n-l \rangle_N], 0 \leq n \leq N-1, \tag{5.27}$$

an operation whose result is now defined for $0 \leq n \leq N-1$ rather than for $0 \leq n \leq 2N-2$.

Since the just defined operation involves two length-N sequences, it is often referred to as an N-point circular convolution, denoted as

$$y[n] = g[n] \circledN h[n]. \tag{5.28}$$

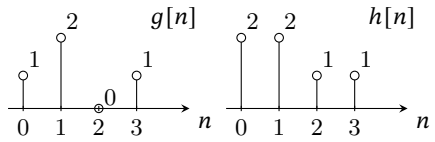
Just like the linear convolution sum, the circular convolution is commutative, which is

$$g[n] \circledN h[n] = h[n] \circledN g[n].$$

As an example, consider the following

$$\{g[n]\} = \left\{\underset{\uparrow}{1} \quad 2 \quad 0 \quad 1\right\}$$
$$\{h[n]\} = \left\{\underset{\uparrow}{2} \quad 2 \quad 1 \quad 1\right\}$$

depicted below,



As they are 4-length signals, the end result will be a length-4 sequence $y_C[n]$ given by

$$y_C[n] = g[n] \textcircled{4} h[n] = \sum_{l=0}^{3} g[l]h[\langle n-l\rangle_4], 0 \leq n \leq 3.$$

Pursuing step-by-step the above formula one obtains the first sample of the circular convolution by

$$\begin{aligned} y_C[0] &= \sum_{l=0}^{3} g[l]h[\langle -l\rangle_4] \\ &= g[0]h[0] + g[1]h[3] + g[2]h[2] + g[3]h[1] \\ &= 1\times 2 + 2\times 1 + 0\times 1 + 1\times 2 = 6. \end{aligned}$$

and as one can see, as there is no shift when $l = 0$ the circularly time-reversed sequence $h[\langle l\rangle_N]$ follows the very same pattern in Equation 5.25.

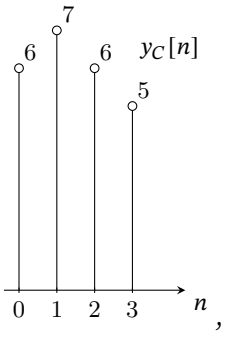
Equivalently,

$$\begin{aligned} y_C[1] &= \sum_{l=0}^{3} g[l]h[\langle 1-l\rangle_4] \\ &= g[0]h[1] + g[1]h[0] + g[2]h[3] + g[3]h[2] \\ &= 1\times 2 + 2\times 2 + 0\times 1 + 1\times 1 = 7; \\ y_C[2] &= \sum_{l=0}^{3} g[l]h[\langle 2-l\rangle_4] \\ &= g[0]h[2] + g[1]h[1] + g[2]h[0] + g[3]h[3] \\ &= 1\times 1 + 2\times 2 + 0\times 2 + 1\times 1 = 6, \end{aligned}$$

and lastly

$$\begin{aligned} y_C[3] &= \sum_{l=0}^{3} g[l]h[\langle 3-l\rangle_4] \\ &= g[0]h[3] + g[1]h[2] + g[2]h[1] + g[3]h[0] \\ &= 1\times 1 + 2\times 1 + 0\times 2 + 1\times 2 = 5. \end{aligned}$$

obtaining the output sequence



,

a length-4 sequence alike the two input sequences $g[n]$ and $h[n]$, as opportune. Notice how—similarly to the case of the linear convolution sum—the sequence $g[n]$ "stays still" while the $h[n]$ sequence "moves" as it is first time-reversed, then shifted along the n axis. The preeminent difference is that both time-reversing and shifting operations are *circular operations* by virtue of the modulo operation. Instead of grabbing the inverted sequence $h[-l]$ and shifting it through the entire domain from left to right, one "rotates it in place" *as if* the sequence was conceptually extended with infinite replicas of itself. Circular convolution in practice is equivalent to shifting an inverted sequence h which was beforehand extended with infinite replicas of it, with the number of sums that is related to the length of both original sequences; the end result will be a circular shift movement that can be described by Figure 5.4.

Conceptually chaining together (a) the circular time-reversal of sequence $h[\langle l\rangle_N]$, (b) the circular shift movement of the circularly reversed sequence $h[\langle n-l\rangle_N]$ and (c) the summation of it with an in-place sequence $g[l]$ grant a deeper understanding of the circular convolution operation that is only possible after having fully grasped the meaning of the three steps (a), (b) and (c).

Of course, performing the circular convolution straightforwardly is not the only available way to gauge the problem—

another method is to jump into the frequency domain and then performing a product between DFTs, as the convenient rule expressed in Table 5.4 declares. Let $g[n]$ and $h[n]$ be the same sequences as in the ealier example. The 4-point DFT $G[k]$ of $g[n]$ is soon obtained by

$$\begin{aligned} G[k] &= g[0] + g[1]e^{-j2\pi\frac{k}{4}} \\ &\quad + g[2]e^{-j4\pi\frac{k}{4}} + g[3]e^{-j6\pi\frac{k}{4}} \\ &= 1 + 2e^{-j\pi\frac{k}{2}} + e^{-j3\pi\frac{k}{2}}, \end{aligned}$$

for values $0 \leq k \leq 3$. Hence,

$$\begin{aligned} G[0] &= 1 + 2 + 1 = 4 \\ G[1] &= 1 - j2 + j = 1 - j \\ G[2] &= 1 - 2 - 1 = -2 \\ G[3] &= 1 + j2 - j = 1 + j \end{aligned}$$

Correspondingly,

$$\begin{aligned} H[k] &= h[0] + h[1]e^{-j2\pi\frac{k}{4}} \\ &\quad + h[2]e^{-j4\pi\frac{k}{4}} + h[3]e^{-j6\pi\frac{k}{4}} \\ &= 2 + 2e^{-j\pi\frac{k}{2}} + e^{-j\pi k} + e^{-j3\pi\frac{k}{2}}, \end{aligned}$$

again for values $0 \leq k \leq 3$. Thus,

$$\begin{aligned} H[0] &= 2 + 2 + 1 + 1 = 6 \\ H[1] &= 2 - j2 - 1 + j = 1 - j \\ H[2] &= 2 - 2 + 1 - 1 = 0 \\ H[3] &= 2 + j2 - 1 - j = 1 + j \end{aligned}$$

Another way to obtain the very outcome would have been by employing the *DFT matrices*, that is

$$\begin{bmatrix} G[0] \\ G[1] \\ G[2] \\ G[3] \end{bmatrix} = \boldsymbol{D}_4 \begin{bmatrix} g[0] \\ g[1] \\ g[2] \\ g[3] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 1-j \\ -2 \\ 1+j \end{bmatrix}$$

and

$$\begin{bmatrix} H[0] \\ H[1] \\ H[2] \\ H[3] \end{bmatrix} = \boldsymbol{D}_4 \begin{bmatrix} h[0] \\ h[1] \\ h[2] \\ h[3] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 1-j \\ -2 \\ 1+j \end{bmatrix}$$

Let $Y_C[k]$ denote the 4-point DFT of sequence $y_C[n]$; from Table 5.4 one has that the circular convolution is simply the product between the two transforms, that is

$$Y_C[k] = G[k]H[k], 0 \leq k \leq 3.$$

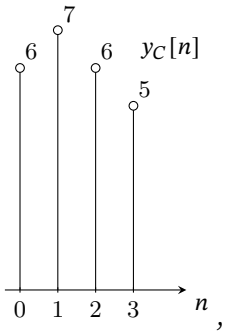
Then, it's quick to perform

$$\begin{bmatrix} Y_C[0] \\ Y_C[1] \\ Y_C[2] \\ Y_C[3] \end{bmatrix} = \begin{bmatrix} G[0]H[0] \\ G[1]H[1] \\ G[2]H[2] \\ G[3]H[3] \end{bmatrix} = \begin{bmatrix} 24 \\ -j2 \\ 0 \\ j2 \end{bmatrix}$$

Now, to finish off the procedure it only remains to invert the Fourier Transform and to obtain the final sequence $y_C[n]$:

$$\begin{bmatrix} y_C[0] \\ y_C[1] \\ y_C[2] \\ y_C[3] \end{bmatrix} = \frac{1}{4}\boldsymbol{D}_4^* \begin{bmatrix} Y_C[0] \\ Y_C[1] \\ Y_C[2] \\ Y_C[3] \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 24 \\ -j2 \\ 0 \\ j2 \end{bmatrix} = \begin{bmatrix} 6 \\ 7 \\ 6 \\ 5, \end{bmatrix}$$

that is the following sequence



which is equal to the one obtained by performing the circular convolution through the time domain.

5.5.6 Mimicking the linear operations

All circular operations we met are very similar to the original linear ones, save they have been designed with the modulo operation. Modulo operation modifies the nature of all operations, inducing a circular behavior that is repeated in the domain.

Although the modulo operation drastically changes the nature of the above operations, they all can be used to "emulate" the linear operations.

To do so, it is enough to apply a zero-padding with many zeros, enough to render all circular effects irrelevant. At heart, one can add zeroes to the left and the right of the original sequence to induce a behavior that is essentially a linear one, because the modulo operation will barely act since the length of the new zero-padded signal is drastically increased. This is in behalf of the fact that when performing modulo-based operation with zero-padded sequences zeros will enter, rather than circular values of the original sequence.

A practical example would be to perform a pseudo-linear convolution out of a circular convolution by extending with zero-padding the two length-4 sequences to length 7:

$$g_e[n] = \begin{cases} g[n] & 0 \le n \le 3 \\ 0 & 4 \le n \le 6 \end{cases},$$

$$h_e[n] = \begin{cases} h[n] & 0 \le n \le 3 \\ 0 & 4 \le n \le 6 \end{cases},$$

When performing the 7-point circular convolution of the two, extended, sequences,

$$y[n] = \sum_{l=0}^{6} g_e[l] h_e[\langle n-l \rangle_7], 0 \le n \le 6,$$

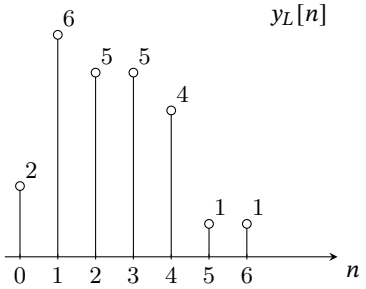
one gets

$$\begin{aligned} y[0] &= g_e[0]h_e[0] + g_e[1]h_e[6] + g_e[2]h_e[5] \\ &+ g_e[3]h_e[4] + g_e[4]h_e[3] + g_e[5]h_e[2] \\ &+ g_e[6]h_e[1] = g[0]h[0] = 1 \times 2 = 2 \end{aligned}$$

If one keeps on doing the process for all samples he gets

$$\begin{aligned} y[1] &= g_e[0]h_e[1] + g_e[1]h_e[0] = 1 \times 2 + 2 \times 2 = 6 \\ y[2] &= g_e[0]h_e[2] + g_e[1]h_e[1] + g_e[2]h_e[0] \\ &= 1 \times 1 + 2 \times 2 + 0 \times 2 = 5 \\ y[3] &= g_e[0]h_e[3] + g_e[1]h_e[2] + g_e[2]h_e[1]g_e[3]h_3[0] \\ &= 1 \times 1 + 2 \times 1 + 0 \times 2 + 1 \times 2 = 5 \\ y[4] &= g_e[1]h_e[3] + g_e[2]h_e[2] + g_e[3]h_e[1] \\ &= 2 \times 1 + 0 \times 2 + 1 \times 2 = 4 \\ y[5] &= g_e[2]h_e[3] + g_e[3]h_e[2] = 0 \times 1 + 1 \times 1 = 1 \\ y[6] &= g_e[3]h_e[3] = 1 \times 1 = 1 \end{aligned}$$

which has the following stem plot



Emulation of linear sequences is beneficial when one explicitly wants to perform linear operations adopting the DFT, especially in the Fast Fourier Transform algorithm form.

5.5.7 Linear convolution using Discrete Fourier Transform

As previously depicted in Section 5.5.6, it is possible to emulate the behavior of linear operators by modifying signals so that they are surrounded by a number of zeros and subsequently using the circular operators. The end result will be roughly equivalent to the linear operators, since the modulo operation becomes irrelevant in the case of long zero-padded signals.

Following this logic, one can find helpful to perform many linear operation by means of their circular counterparts, especially the *linear convolution*. This pragmatic move is truly effective, as the linear convolution can be performed as seen in Section 5.5.6 with the help of the DFT in frequency domain. Since Discrete Fourier Transforms can exploit the *Fast Fourier Transform* algorithm, it is of profound interest to

develop methods for the implementation of the linear convolution using already working and efficient solutions for DFTs.

Let $g[n]$ and $h[n]$ be two finite-length sequences of different length—respectively—of N and M. The length of a linear convolution sum would be $L = N + M - 1$. Considering that the circular convolution outputs a sequence whose length is *the same* of the input sequences, to imitate the behavior of the linear convolution sum it is imperative to extend the original sequences to match the length L. For sure, two extended length-L sequences should be defined,

$$g_e[n] = \begin{cases} g[n] & 0 \leq n \leq N-1 \\ 0 & N \leq n \leq L-1 \end{cases},$$

$$h_e[n] = \begin{cases} h[n] & 0 \leq n \leq N-1 \\ 0 & N \leq n \leq L-1 \end{cases},$$

and the convolution will be given by

$$y_L[n] = g[n] \circledast h[n] = y_C[n] = g_e[n] \textcircled{L} h_e[n].$$

Fundamentally, the linear convolution sum is equal to the circular convolution in the case of the extended sequences so that they are long exactly as the linear convolution sum outcome. The corresponding implementation scheme is illustrated in Figure 5.6.

The procedure as in Figure 5.6 is rather convenient, as previously—when we dealt with Discrete-time Fourier Transforms—the modulation passage required to work with *continuous* DTFTs in place of convenient *Discrete* Fourier Transforms. By all means this represents a paramount improvement over the case as in Figure 4.5 and described in Section 4.3.2.

5.5.8 The Overlap–Add Method

A special attention is required by the linear convolution of a finite-length sequence with an *infinite-length sequence*. Since now one of the two sequences is an infinite-length sequence, we cannot use the previous technique again, since that explicitly demands to have two finite-length inputs so that the outcome will be a finite-length signal as well, with a predetermined length.

The trick to perform said operation is to *partition the infinite-length sequence in finite-length blocks*. Consider the DFT-based implementation of the circular convolution as in 5.27, with input sequence $x[n]$ being an infinite-length sequence, or at least a sequence whose length is much greater than M. Let the circular convolution be implemented by means of the convenient trick of going into the frequency-domain and perform a multiplication of DFTs.

To do so, one first has to segment the dragging sequence $x[n]$ into smaller sequences: without loss of generality, it is legitimate to assume that $x[n]$ is a causal sequence—this is feasible, since the input signal can be always be shifted and adjusted to make it causal. Thereafter, many contiguous N-length sequences $x_m[n]$ shall be produced

$$x[n] = \sum_{m=0}^{\infty} x_m[n - mN]$$

with each sequence called **partition**, detailed as such

$$x_m[n] = \begin{cases} x[n + mN], & 0 \leq n \leq N-1 \\ 0, & \text{otherwise} \end{cases}$$

This sum is constructed so that a new collection of sequences $x_m[n]$ is obtained by segmenting the original sequence $x[n]$, each one carrying a small portion of the far-reaching sequence x, and so that each value of $m \in \mathbb{N}$ will yield a new block of the original signal. The sequence of sequences x_m will ultimately generate the original sequence x. While this looks at least labyrinthine, there are many reasons to do so. Partitioning the original sequence in many smaller ones having the same length grants us to compute the total convolution by means of *partial* convolutions. Thus, the output is the *sum of the results from partial convolutions* (i.e. convolutions made with partitions $x_m[n]$):

$$y[n] = h[n] \circledast x[n] = \sum_{m=0}^{\infty} y_m[n - mN], \tag{5.29}$$

where

$$y_m[n] = h[n] \circledast x_m[n]$$

is the linear convolution—obtained with a DFT-based procedure—of the impulse response $h[n]$ with the partition $x_m[n]$ of the sequence $x[n]$. Since $h[n]$ is of length M and $x_m[n]$

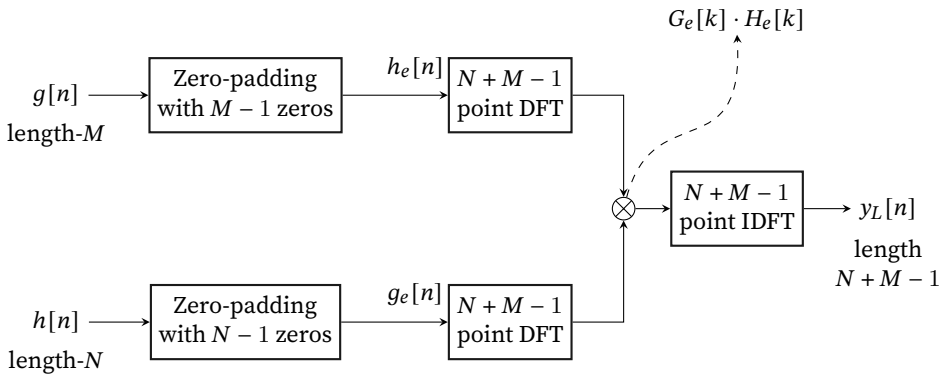


Figure 5.6: Implementation scheme of the linear convolution sum by virtue of Discrete Fourier Transforms. The magic occurs during the signal modulation operation, when the single operation required is a multiplication between signals, in this procedure the multiplication between Discrete Fourier Transforms obtained from extended sequences $G_e[k] \cdot H_e[k]$.

is of length N, the linear convolution will possess length $N+M-1$.

There is an elegant correspondence between the sequences $y[n], x[n]$ and $y_m[n], x_m[n]$, condensing in the fact that the first ones are obtained by the second ones by summing up all m partitioned sequences, each one right-shifted by the quantity mN.

As a result, the desired linear convolution $y[n]$ is broken up into a sum of infinite short-length linear convolutions $y_m[n]$, each of length $N+M-1$. Without exception, each one can be implemented through the DFT technique we encountered before and considering a basis of $N+M-1$ points.

The implementation of 5.29 requires a further attention: the first convolution $y_0[n]$ in the sum is of length $N+M-1$ like the other ones and is defined for values of $0 \leq n \leq N+M-2$; the second one $y_1[n]$ is defined for $N \leq n \leq 2N+M-2$, the third one for $2N \leq n \leq 3N+M-2$. Following the sum strictly results in an *overlap* of $M-1$ samples between any two short linear convolutions $x_{r-1}[n]$ and its successor $x_r[n]$ for $(r-1)N \leq n \leq rN+M-2$. The output sequence $y[n]$ can be gathered by not only the concatenation of all partial convolutions, but by also *summing up all the overlapping parts*. This method is in fact called the **Overlap–Add Method** since the result of the short linear convolutions overlap in some samples and the overlapped portions are then added altogether to achieve the final outcome. Depending on the value of M, $M-1$ values for each partial convolution should always be merged into a sum considering the proper position of them.

As an exemplification, let $N=7$ and $M=5$. Global outcome $y[n]$ can be obtained by performing a number of overlaps and sums, such that

$$y[n] = \begin{cases} y_0[n] & 0 \leq n \leq 6 \\ y_0[n] + y_1[n-7] & 7 \leq n \leq 10 \\ y_1[n-7] & 11 \leq n \leq 13 \\ y_1[n-7] + y_2[n-14] & 14 \leq n \leq 17 \\ y_2[n-14] & 18 \leq n \leq 20 \\ \vdots & \vdots \end{cases}$$

Figures 5.7 and **??** illustrate succinctly this process.

In MATLAB, function `fftfilt` can be invoked to implement the above method, with Fast Fourier Transform and by applying the Convolution Theorem.

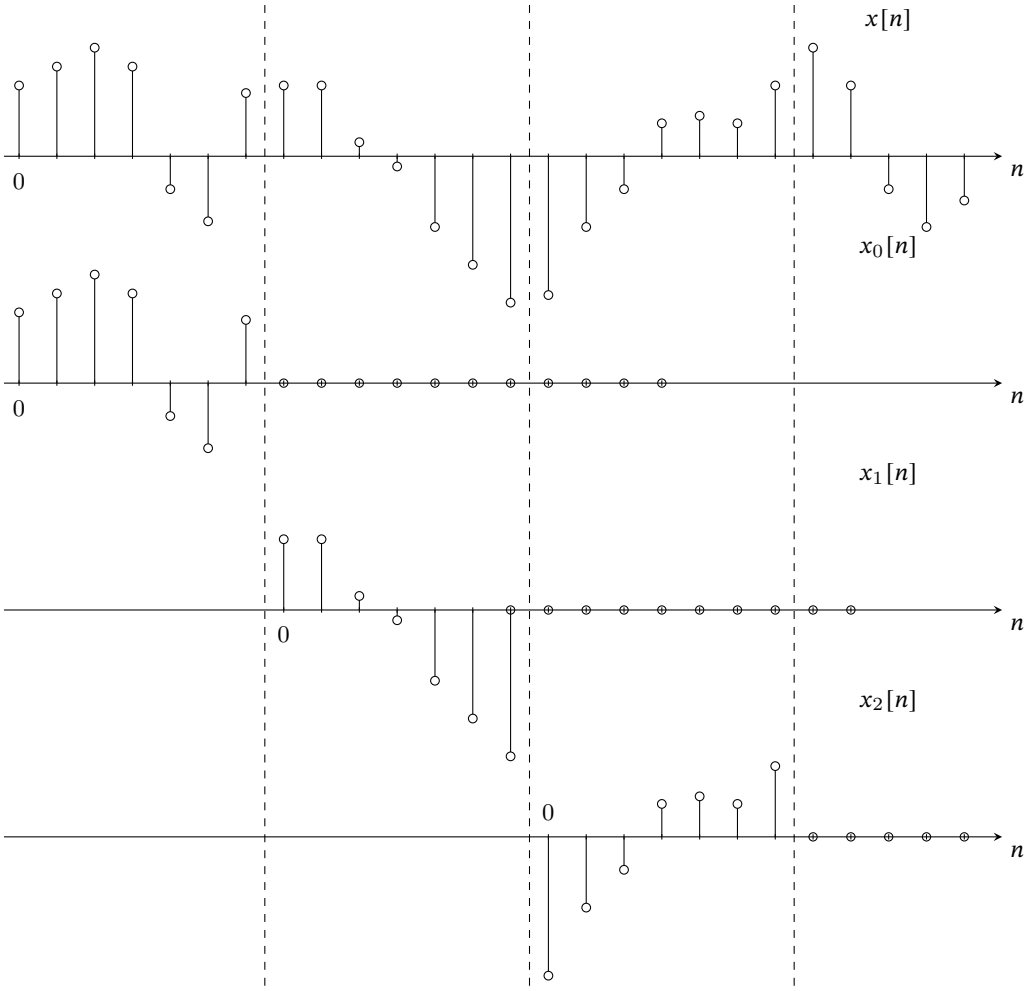


Figure 5.7: Partitioning of the sequence $x[n]$. Partitioning occurs with $N = 7$, so that each partition $x_m[n]$ counts for exactly 7 samples. Ideally, each segment is zero-padded with a protracted series of zeros as needed. Notice that all sequences are *causal* and are shifted only for illustration purpose.

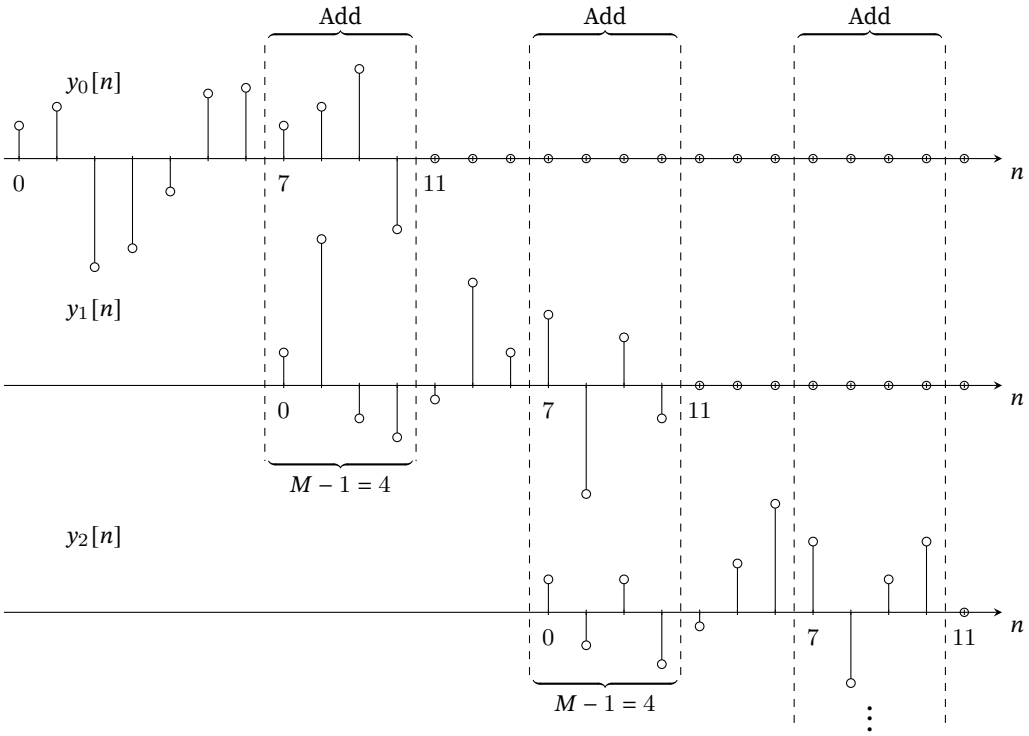


Figure 5.8: Overlap–Add of the partial output sequences, with increasing m. Merging the sequences involves overlapping and summing over $M-1$ samples, in this case $M=5$, so over $M-1=4$ samples. A remarkable attribute is that all sequences are *causal* since the corresponding $x_m[n]$ are causal as well. The random sequences are purely illustrative and are not generated by a computation process from sequences in Figure 5.7.

Chapter 6

Short Time Fourier Transform

The **short-time Fourier transform (STFT)**, is a Fourier-related transform used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time.[1] In practice, the procedure for computing STFTs is to divide a longer time signal into shorter segments of equal length and then compute the Fourier transform separately on each shorter segment. This reveals the Fourier spectrum on each shorter segment. One then usually plots the changing spectra as a function of time, known as a spectrogram or waterfall plot, such as commonly used in software defined radio (SDR) based spectrum displays. Full bandwidth displays covering the whole range of an SDR commonly use fast Fourier transforms (FFTs) with 2^{24} points on desktop computers[7].

Any Fourier Transform reveals which frequency components are present in a function, hence it can be regarded as a *frequency description* of the signal. Indeed, an exemplary case is the case of pure sinusoidal signals composed of a single sinusoid, whose Fourier Transform is generally a pair of delta-impulses located at the exact frequency at which the sinusoid oscillates, with one of the two impulses placed at negative frequencies. In some cases—when the frequency domain is periodic and can be described by the principal value, as in 4.10—the negative impulse of the pair is omitted since it can be described by the positive one. Sinusoids composed of the sum of multiple pure sinusoids will yield a frequency description that is the superposition of frequency descriptions of the single sinusoids. As each sinusoid yields a delta impulse, the sum is clearly visible from plots.

§TODO ADD PLOT

Bibliography

[1] Wikipedia, https://en.wikipedia.org/wiki/Time_domain, 2022.

[2] Wikipedia, https://en.wikipedia.org/wiki/Discrete_time_and_continuous_time, 2022.

[3] Wikipedia, https://en.wikipedia.org/wiki/Linear_time-invariant_system, 2022.

[4] Wikipedia, https://en.wikipedia.org/wiki/Frequency_domain, 2022.

[5] Wikipedia, https://en.wikipedia.org/wiki/Fourier_transform, 2022.

[6] Wikipedia, https://en.wikipedia.org/wiki/Discrete_Fourier_transform, 2022.

[7] Wikipedia, https://en.wikipedia.org/wiki/Short-time_Fourier_transform, 2022.