Signal Processing

Lab 2 – Continuous Time Signal

Introduction

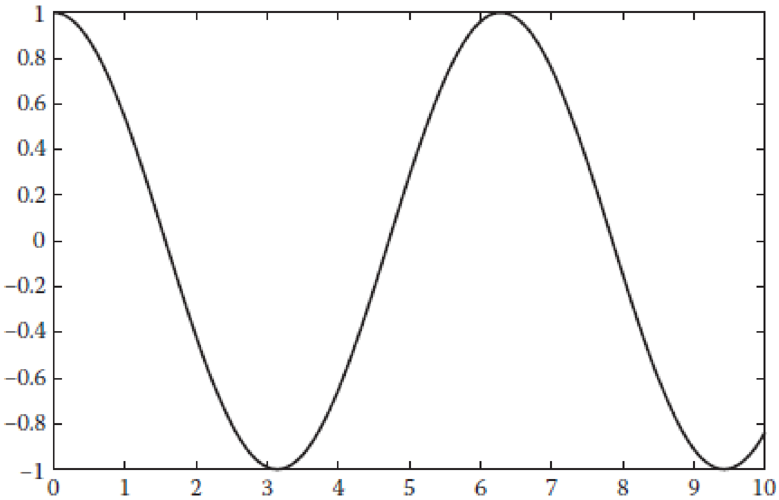
In this chapter, we introduce the concept of signals. Signals are classified according to their basic characteristics and properties. Furthermore, several elementary basic signals are introduced. A signal is defined as any natural quantity that varies according to one or more independent variables such as time or space. Time is usually the independent variable, but other variables like frequency can also be considered. Examples of signals are a sound, an image, an electrical current or voltage, a transmitted message, and many others. From a mathematical point of view, a signal is described by a function of one or more independent variables. According to the number of independent variables, a signal is characterized as a one-dimensional (1-D), a two-dimensional (2-D), or a multidimensional signal.

I. Categorization by the Variable Type

There are three main categories where a signal can be classified according to the type of the independent and dependent variables.

1. Continuous-Time Signals

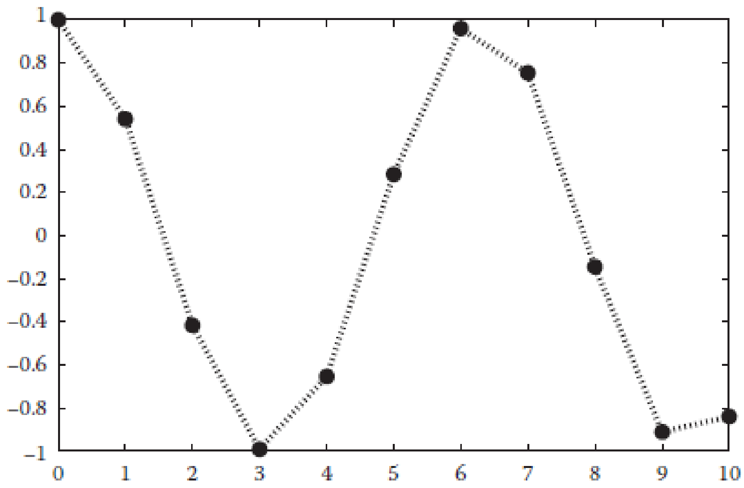
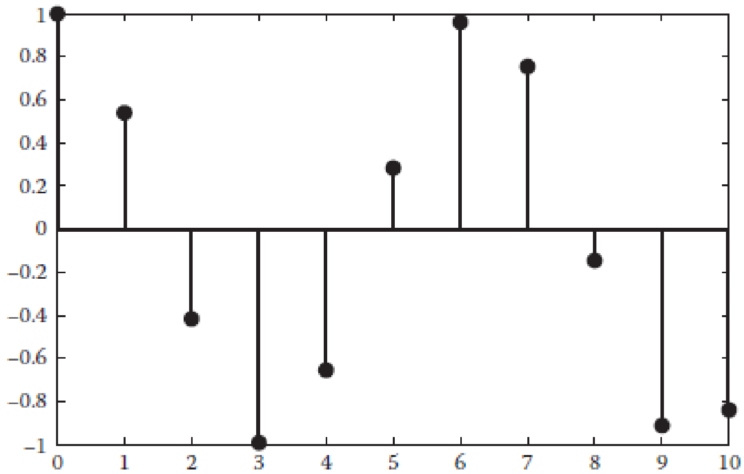
A signal is called continuous-time (or analog) signal if the independent variable (time) is defined in a continuous interval. For 1-D signals, the domain of a signal is a continuous interval of the real axis. In other words, for continuous-time signals the independent variable t is continuous. Moreover, the dependent value that usually denotes the amplitude of the signal is also a continuous variable. An example of such a signal is speech as a function of time. An analog signal is expressed by a function $x(t)$, where t takes real values. Unfortunately, in MATLAB, and generally on a computer, the work is done in discrete time. However, a continuous-time signal or function is approximated satisfactory by using the corresponding discrete-time functions with very small time step. In the following example, the analog signal $y(t) = \cos(t)$, $0 < t < 10$ is defined and plotted.

Commands	Results	Comments
`t=0:0.01:10`		Time (the independent variable t) is defined by using a very small step (time step = 0.01) in the continuous domain $0 \le t \le 10$.
`y=cos(t);`		The dependent variable $y(t)$ is defined in the continuous set of values $-1 \le y(t) \le 1$.
`plot(t,y)`		The analog signal is drawn by using the command `plot`.

2. Discrete-Time Signals

A signal is called a discrete-time signal if the independent variable (time) is defined in a discrete interval (e.g., the set of integer numbers), while the dependent variable is defined in a continuous set of values. In the following example, the discrete-time signal y[n]=cos[n] is plotted. Note that when referring to discrete time the variable n is typically used to represent the time.

Commands	Results	Comments
`n=0:10`		Discrete time n is defined with step 1.
`y=cos(n);`		The dependent variable $y[n]$ is defined in the continuous set of values $-1 \le y[n] \le 1$.

Commands	Results	Comments
`plot(n,y,':o')`		Graph of the discrete-time signal $y[n]$ by using the command `plot` with the proper syntax. The points determined by the circles are the values of $y[n]$.
`stem(n,y)`		Using the command `stem` is more appropriate when dealing with discrete-time signals.

A discrete-time signal $x[n]$ is usually obtained by sampling a continuous-time signal $x(t)$ at a constant rate. Suppose that Ts is the sampling period, that is every T_s s we sample the value of $x(t)$. Suppose also that $n \in \mathbb{Z}$, i.e., $n = \pm 0, \ \pm 1, \ \pm 2, \ldots$ The sequence of the samples $x[nT_s]$, $n \in \mathbb{Z}$ derived from the continuous-time signal $x(t)$ is sometimes called time series and denotes a discrete-time signal. The sampling period T_s is constant and thus can be omitted from the notation. In this book (see for example the two previous figures) we assume that $T_s = 1$.

3. Digital Signals

Digital signals are the signals that both independent and dependent variables take values from a discrete set. In the following example, the signal $y[n] = \cos[n]$ is again plotted, but we use the command round to limit the set of values that $y[n]$ can take. That is, $y[n]$ can be -1, 0, or 1.

We discuss continuous-time signals. The basic signals are introduced and the most common properties and categories are presented.

II. Basic Continuous-Time Signals

In this section, we present the basic continuous-time signals along with the way that they are implemented and plotted in MATLAB.

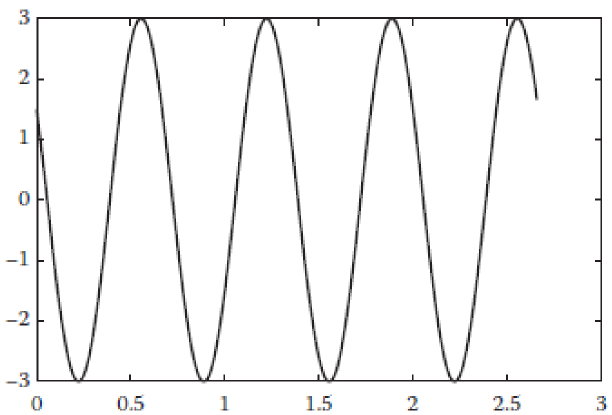
1. Sinusoidal Signals

The first basic category presented is that of sinusoidal signals. This type of signal is of the form $x(t)= A\cos(\Omega t+\theta)$, where Ω is the angular frequency, given in *rad*/*s*, A is the amplitude of the sinusoidal signal, and θ is the phase (in radians). Sinusoidal signals are periodic signals with fundamental period T given by $T = 2\pi/ \Omega s$. Finally, a useful quantity is the frequency f given in Hertz. Frequency f is defined by $f = 1/T$ or $f = \Omega/2\pi$.

Example

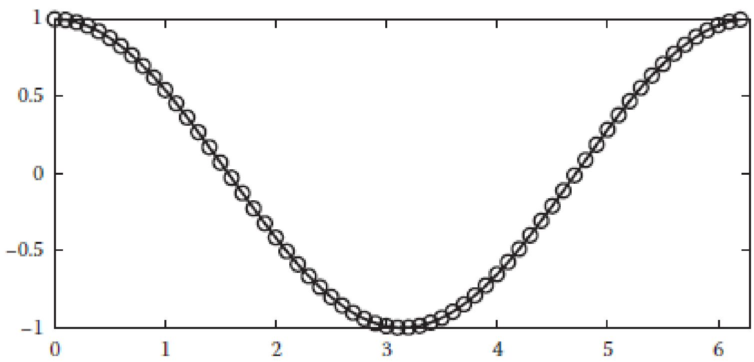
Plot the signal $x(t) = 3 \cos(3\pi t+\pi/3)$ in four periods.

First, the period T is calculated as $T=2\pi/\Omega=2\pi/3\pi=2/3$. Hence, the MATLAB implementation is as follows.

Commands	Results	Comments
`A=3;`		The amplitude of the signal is 3.
`Omega=3*pi;`		The angular frequency is 3π.
`thita=pi/3;`		The phase is $\pi/3$.
`T= 2*pi/omega;`		The period is 2/3.
`t=0:0.01:4*T;`		The time is defined from 0 to $4T$.
`x=A*cos(omega*t+thita);`		Signal definition.
`plot(t,x)`		The signal is plotted in time of four periods.

When referring to sinusoidal signals we refer both to cosines and sines, as a cosine and a sine are in fact the same signal with a $\theta=\pi/2$ phase difference. In the figure below the signals cos(t) and sin(t+$\pi/2$) are plotted for time of one period.

Commands	Results	Comments

`t=0:0.1:2*pi;`
`x1=cos(t);`
`x2=sin(t+pi/2);`
`plot(t,x1,t,x2,'o')`
`xlim([0 2*pi])`
		It is clear that the signals $\cos(t)$ and $\sin(t+\pi/2)$ are identical. The signal $\cos(t)$ is plotted with the solid line, while $\sin(t+\pi/2)$ is plotted with the circles. One can say that cosine is a sine with phase $\theta=\pi/2$.

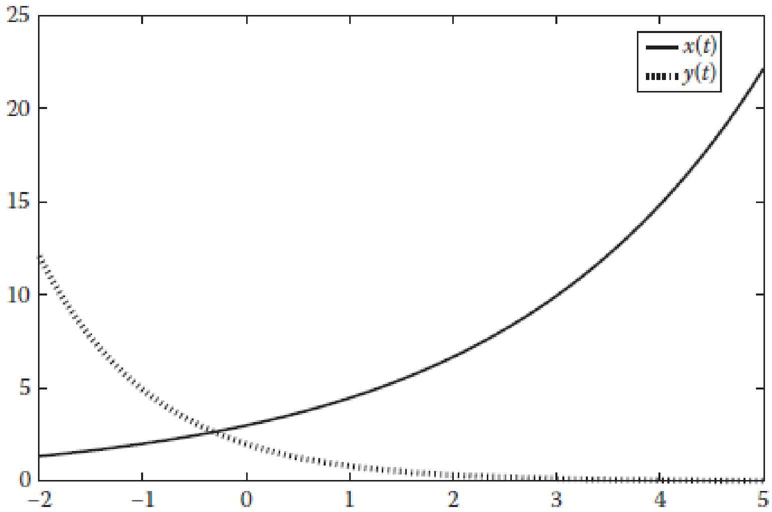
2. Exponential Signals

Exponential signals are signals of the form $x(t)$ = Ae^{bt}. If b > 0, $x(t)$ is an increasing function while if b < 0, $x(t)$ is a decreasing function. At t = 0 the signal takes the value $x(0)$ = A as e^{bt} = 1.

Example

Plot the signals $x(t) = 3e^{0.4t}$ and $y(t) = 2e^{-0.9t}$ in the time interval $-2 \leqslant t \leqslant 5$.

Commands	Results/Comments

`t=-2:.1:5;`
`x=3*exp(0.4*t);`
`y=2*exp(-0.9*t);`
`plot(t,x,t,y,':');`
`legend('x(t)','y(t)')`
	

3. Complex Exponential Signals

Another signal highly associated with the sinusoidal signals is the complex exponential signal $Ae^{i\Omega t+\theta}$, which is also periodic with fundamental period given by $T = 2\pi/\Omega$. This is derived straightforwardly from Euler's formula:

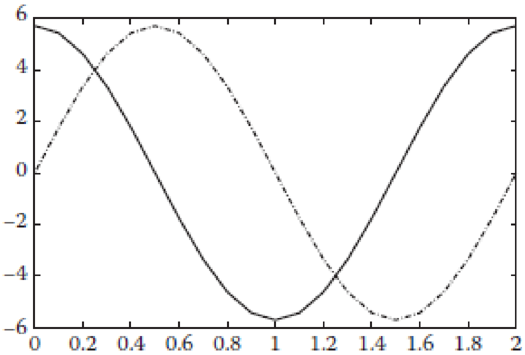
$$Ae^{j\Omega t+\theta} = A(\cos(\Omega t+\theta)+j\sin(\Omega t+\theta)). \tag{2.1}$$

From Equation 2.1, we conclude that $\mathrm{Re}\{Ae^{i\Omega t+\theta}\} = A\cos(\Omega t+\theta)$ and $\mathrm{Im}\{Ae^{i\Omega t+\theta}\}=A\sin(\Omega t+\theta)$, where $Re\{z\}$ is the real part and $\mathrm{Im}\{z\}$ is the imaginary part of a complex number z.

Example

Plot the real and imaginary parts of the signal $y(t) = 2e^{j\pi t+\pi/3}$ in time of one period.

First, the period is calculated as $T = 2\pi/\Omega = 2\pi/\pi = 2$. Thus,

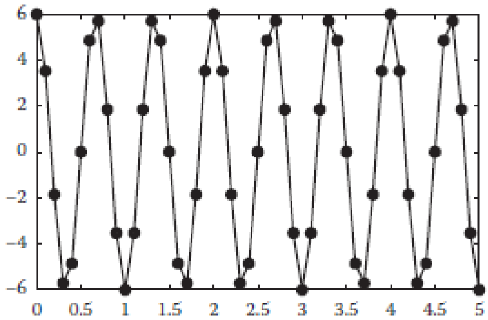
Commands	Results	Comments
`t=0:.1:2;` `y_re=real(2*exp(j*pi*t+pi/3))` `y_im=imag(2*exp(j*pi*t+pi/3));`		Definition of the real and imaginary parts of the signal $y(t)=2e^{j\pi t+\pi/3}$
`plot(t,y_re,t,y_im,'-.');`		The real part of $y(t)$ is $2\cos(\pi t+\pi/3)$ while the imaginary part of $y(t)$ is $2\sin(\pi t+\pi/3)$.

It is very easy to perform operations between complex exponential signals. For example, the product of two complex exponential signals is easily computed by simply adding their exponents.More precisely, if $y_1(t) = Ae^{j\Omega_1 t}$ and $y_2(t) = Ae^{j\Omega_2 t}$ then $y_3(t) = y_1(t)\cdot y_2(t) = A\cdot B\cdot Ae^{j(\Omega_1+\Omega_2)t}$.

Example

Plot the real parts of the signals $x(t) = 2e^{j\pi t}\, 3e^{j2\pi t}$ and $y(t) = 6e^{j3\pi t}$.

Commands	Results	Comments

`t=0:0.1:5;`
`x=(2*exp(j*pi*t)).*(3*exp(j*2*pi*t))`
`y=6*exp(j*3*pi*t);plot(t,real(x),t,`
`real(y),'ko')`
		The real parts of $x(t)$ and $y(t)$ are the same.

The signals $Ae^{j\Omega t+\theta}$, A cos(Ωt+θ), and A sin(Ωt+θ) have a key role in signal processing as any periodic signal can be expressed as a sum of single frequency signals.

4. Unit Step Function

Another basic signal is the unit step function u(t). The unit step function is given by

$$u(t) = \begin{cases} 1, & t > 0 \\ \frac{1}{2}, & t = 0 \\ 0, & t < 0 \end{cases}. \tag{2.2}$$

However, $u(t)$ is a continuous-time signal; thus the value at $t = 0$ can be omitted for convenience and $u(t)$ can be defined as

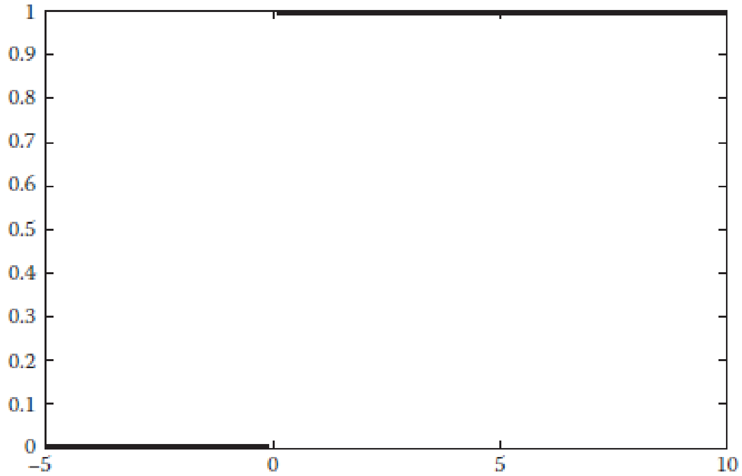
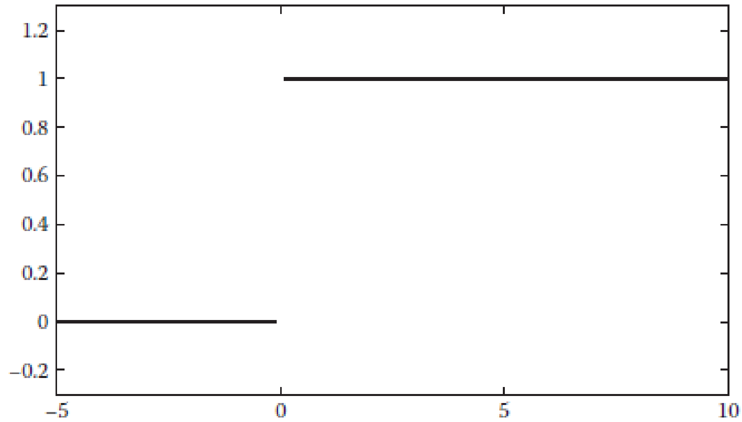
$$u(t) = \begin{cases} 1, & t \geq 0 \\ 0, & t < 0 \end{cases}. \tag{2.3}$$

The MATLAB command that generates the unit step function is the command *heaviside*(t). According to MATLAB programmers, unit step function is given by

$$u(t) = \begin{cases} 1, & t > 0 \\ 0, & t < 0 \end{cases}, \tag{2.4}$$

i.e., it is not defined at $t = 0$. In the following example three different methods of defining and plotting the unit step function are presented.

First Method
With use of the command *heaviside.*

Commands	Results	Comments
`t=-5:0.1:10;`	`t=-5,...,-0.1, 0, 0.1,...,10`	Definition of the time interval $-5 \le t \le 10$.
`u=heaviside(t)`	`u=0,...,0, NaN, 1,...,1`	Definition of $u(t)$.
`plot(t,u)`		Graph of $u(t)$.
`ylim([-0.3 1.3])`		Change of axis for better appearance of the graph.

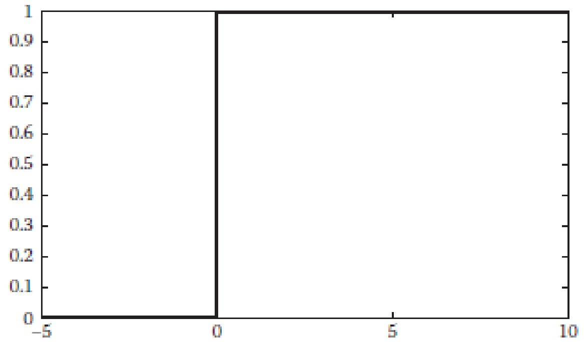
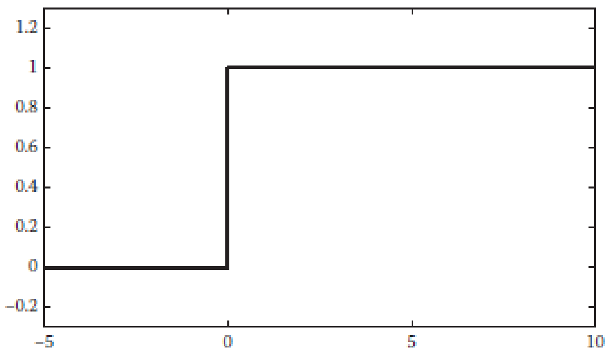
Second Method

Use of the technique of defining and plotting piecewise functions.

Commands	Results	Comments
`t1=-5:.1:0`	`t1=-5, -4.9,...,0`	Definition of the first time interval $-5 \le t \le 0$.
`t2=0:.1:10`	`t2=0, 0.1,...,10`	Definition of the second time interval $0 \le t \le 10$.
`u1=zeros(size(t1))`	`u1=0, 0,...,0`	Implementation of the part of $u(t)$ that corresponds to time $t1$.
`u2=ones(size(t2));`	`u2=1, 1,...,1`	Implementation of the part of $u(t)$ that corresponds to time $t2$.

(continued)

(continued)

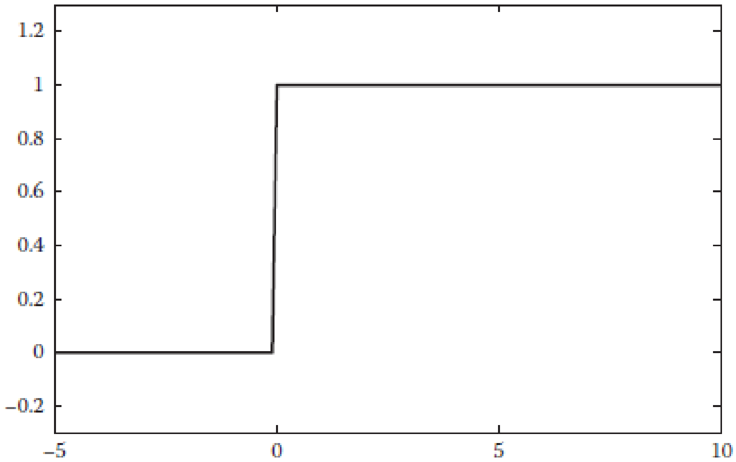
Commands	Results	Comments
`t=[t1 t2];`	`t=-5,...,-0.1, 0,...,10`	Concatenation of the two time vectors.
`u=[u1 u2];`	`u=0,...,0, 1,...,1`	Concatenation of the two function vectors.
`plot(t,u);`		Graph of the unit step function $u(t)$ in the time interval $0 \le t \le 10$.
`ylim([-0.3 1.3])`		Change of axis for better appearance of the graph.

Third Method

Implementation with specific number of zeros and ones.

Commands	Results	Comments
`t=-5:.1:10;`	`t=-5,...,-0.1, 0,...,10`	Time definition.
`u=[zeros(1,50) ones(1,101)];`	`u=0,...,0, 1,...,1`	The vector t consists of 151 elements. Thus, the first 50 elements of t are matched with zeros while the next 101 elements (including $t=0$) of t are matched with ones. The two vectors are concatenated.

(continued)

Commands	Results	Comments
`plot(t,u);` `ylim([−0.3 1.3])`		Graph of the unit step function $u(t)$ in the time interval $0 \leq t \leq 10$.

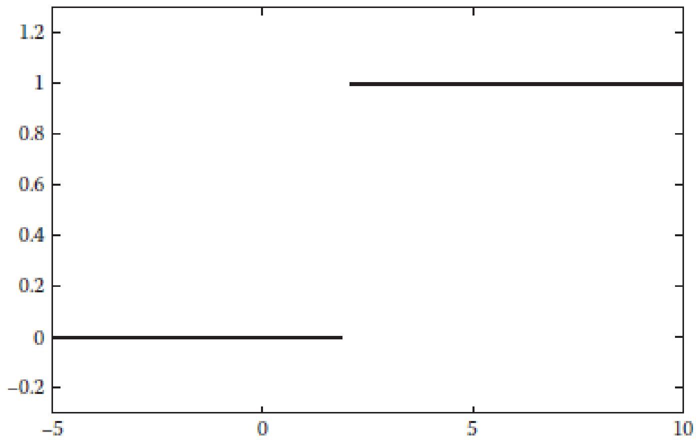
The general form of the unit step function is

$$u(t-t_0) = \begin{cases} 1, & t - t_0 \geq 0 \Rightarrow t \geq t_0 \\ 0, & t - t_0 < 0 \Rightarrow t < t_0 \end{cases} . \quad (2.5)$$

Suppose that we want to define and plot the unit step function for $t_0 = 2$, i.e., we want to define and plot the function $u(t - 2)$.

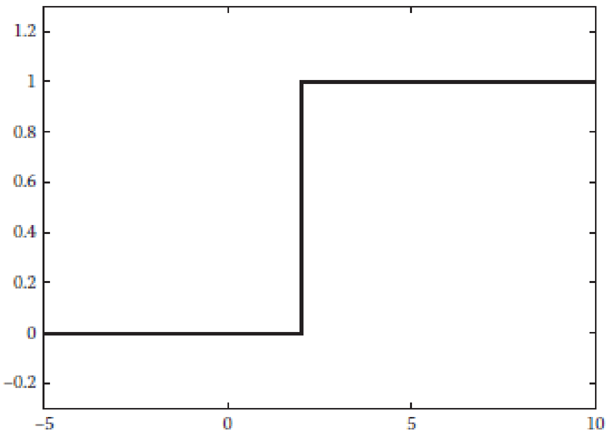
First Method

With use of the command heaviside.

Commands	Results	Comments
`t=−5:0.1:10;`		Definition of the time interval $-5 \leq t \leq 10$.
`u=heaviside(t-2)`		Definition of $u(t-2)$.
`plot(t,u)` `ylim([−0.3 1.3])`		Graph of $u(t-2)$. Notice that function $u(t-2)$ becomes 1 from the time instance $t=2$ and afterward. Thus, $u(t-2)$ is a shifted by 2 units to the right version of $u(t)$.

Second Method

Use of the technique of defining multipart functions.

Commands	Results	Comments
`t1=-5:.1:2`	`t1=-5, -4.9, . . . ,2`	Definition of the first time interval $-5 \le t \le 2$.
`t2=2:.1:10`	`t2=2, 0.1, . . . ,10`	Definition of the second time interval $2 \le t \le 10$.
`u1=zeros(size(t1));`	`u1=0, 0, . . . ,0`	Implementation of the vector of $u(t)$ that corresponds to time t_1.
`u2=ones(size(t2));`	`u2=1, 1, . . . ,1`	Implementation of the vector of $u(t)$ that corresponds to time $t2$.
`t=[t1 t2];`		Concatenation of the two time vectors.
`u=[u1 u2];`		Concatenation of the two function vectors.
`plot(t,u)` `ylim([-0.3 1.3])`		Graph of the signal $u(t-2)$.

Finally, we mention that the command heaviside can be also used with symbolic variables.

Commands	Results	Comments
`syms t`		Definition of t as a symbolic variable.
`u=heaviside(t)`	`u=heaviside(t)`	Definition of $u(t)$ as a symbolic expression.
`diff(u,t)`	`ans =dirac(t)`	The derivative of $u(t)$ is the Dirac delta function $\delta(t)$.

5. Unit Impulse or Dirac Delta Function

The Dirac function $\delta(t)$, strictly speaking, is not a function but is defined through its properties. The main property is

$$\int_{-\infty}^{\infty} f(t)\delta(t) = f(0), \tag{2.6}$$

where f(.) is an arbitrary function. Suppose that $f(t) = 1$, t∈(-∞, ∞). Then (2.6) becomes

$$\int_{-\infty}^{\infty} \delta(t) = 1. \tag{2.7}$$

For practical reasons, $\delta(t)$ can be loosely defined as a function that is infinite at $t = 0$ and zero elsewhere. This is the way that $\delta(t)$ is implemented from the MATLAB programmers. The mathematical expression is

$$\delta(t) = \begin{cases} \infty, & t = 0 \\ 0, & t \neq 0 \end{cases}. \tag{2.8}$$

An alternative definition for the Dirac function that is usually applicable when dealing with discrete-time signals is given now. In this case, we refer to δ(t) as the *Delta* or the *Kronecher* function. The mathematical definition of the delta function is

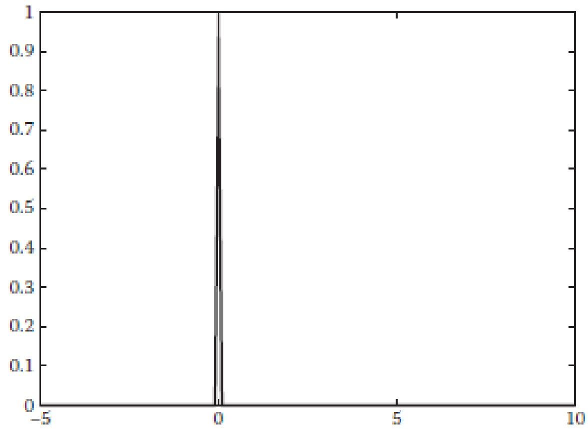
$$\delta(t) = \begin{cases} 1, & t = 0 \\ 0, & t \neq 0 \end{cases}. \tag{2.9}$$

Note that if the definition given by (2.9) is used, the graph of $\delta(t)$ we get in MATLAB is closer to the one met at the theory of signals and systems. The MATLAB implementation of δ(t) is as follows.

Commands	Results	Comments
`t1=-5:.1:-0.1;`		Definition of the first time interval $-5 \le t < 0$.
`t2=0;`		The second time interval is defined only for one time instance, namely for $t = 0$.
`t3=0.1:.1:10;`		Definition of the third time interval $0 < t \le 10$.
`d1=zeros(size(t1));`		Implementation of the part of $\delta(t)$ that corresponds to time $t1$.
`d2=1;`		Implementation of the part (vector of one element) of $\delta(t)$ that corresponds to time $t2$.

(continued)

(continued)

Commands	Results	Comments
`d3 = zeros(size(t3));`		Implementation of the part of $\delta(t)$ that corresponds to time $t3$.
`t= [t1 t2 t3];`		Concatenation of the three time vectors.
`D= [d1 d2 d3];`		Concatenation of the three function vectors.
`plot(t,d)`		Graph of the Dirac function $\delta(t)$.

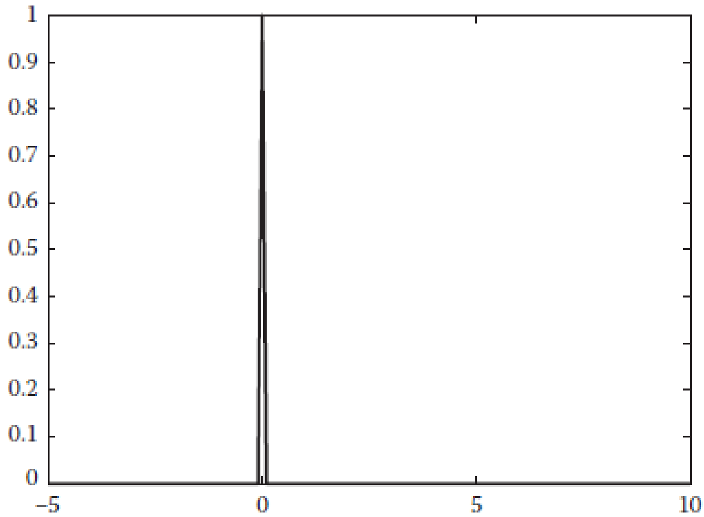
The signal defined according to (2.9) can be plotted with the help of the command *gauspuls(t)*.

Remark

The command *gauspuls* is not created in MATLAB for that purpose but it is convenient for us to use it in the special way that is presented in this book.

Commands	Results	Comments
`t=-5:.1:10;`		Definition of time.
`s=gauspuls(t)`		Definition of the unit pulse.

(continued)

Commands	Results	Comments
`plot(t,s)`		The two graphs are identical.

In case that the first definition of the Dirac function is used, i.e.,

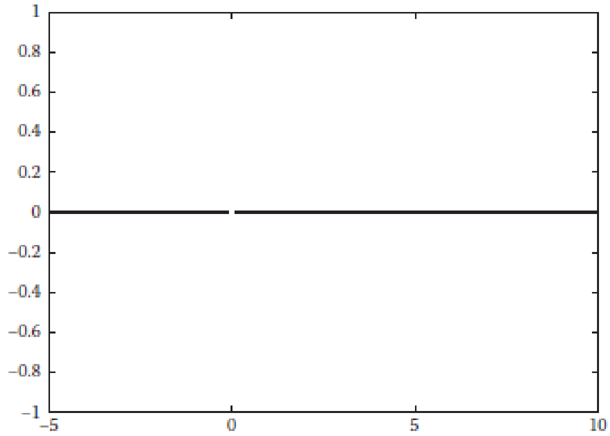
$$\delta(t) = \begin{cases} \infty, & t = 0 \\ 0, & t \neq 0 \end{cases}$$

the MATLAB implementation is

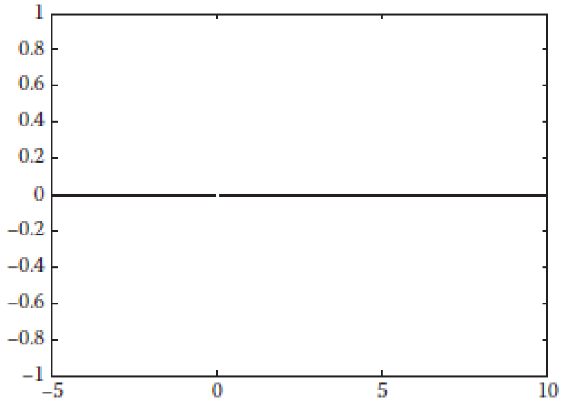
Commands	Results	Comments
`t=-5:.1:10`	`t=-5,...,-0.1, 0, 0.1,...,10`	Definition of the time interval.
`d=[zeros(1,50) inf zeros(1,100)];`	`u=0,...,0, inf, 0,...,0`	The vector t consists of 151 elements. Thus, the first 50 elements of t are matched with zeros, the element with index 51 (that corresponds to $t=0$) is matched with `inf`, while the next 100 elements of t are matched again with zeros. The three vectors are concatenated.

(continued)

(continued)

Commands	Results	Comments
`plot(t,d)`		Graph of $\delta(t)$. Notice that at the time instance $t=0$ there is a gap in the graph that denotes ∞.

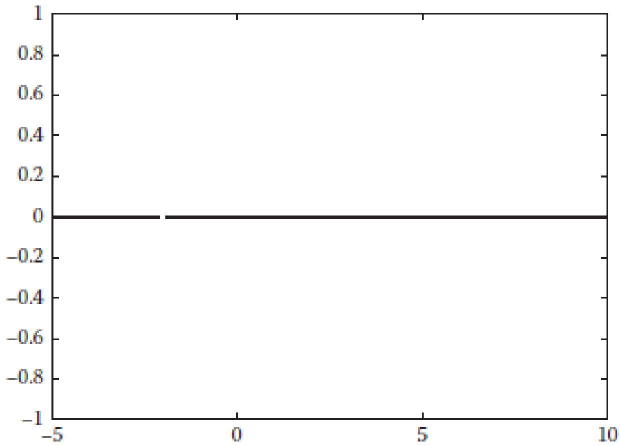
The MATLAB command that defines the Dirac function is the command *dirac(t)*. The command *dirac* is demonstrated in the following example.

Commands	Results	Comments
`t=-5:.1:10`	`t=-5,...-0.1, 0,0.1,......10`	Definition of the time interval $-5 \leq t \leq 10$.
`d=dirac(t);`	`d=0,........0,inf,0,..........0`	Definition of $\delta(t)$.
`plot(t,d)`		Graph of $\delta(t)$. The obtained graph is similar to that of the previous example.

The general form of the Dirac function is

$$\delta(t-t_0) = \begin{cases} \infty, & t = t_0 \\ 0, & t \neq t_0 \end{cases}. \qquad (2.10)$$

Suppose that we want to define and plot the Dirac function for $t_0 = -2$, i.e., we want to define and plot the function $\delta(t+2)$.

Commands	Results	Comments
`t=-5:0.1:10;`		Definition of the time interval $-5 \le t \le 10$.
`d=dirac(t+2)`		Definition of $\delta(t+2)$.
`plot(t,d)`		Graph of $\delta(t+2)$. The function $\delta(t+2)$ becomes infinite at $t=-2$. The signal $\delta(t)$ is shifted by 2 units to the left.

Finally, we mention that the dirac command can be also used with symbolic variables.

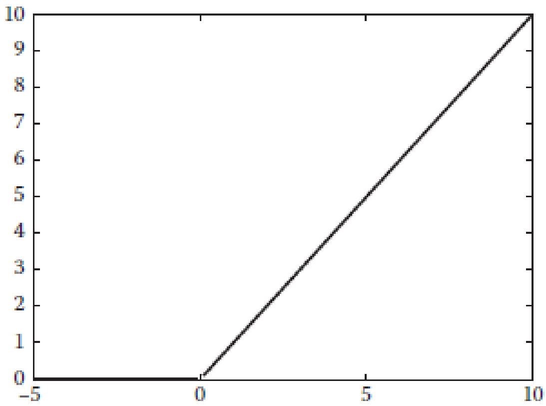
Commands	Results	Comments
`syms t`		Definition of the symbolic variable t.
`d=dirac(t)`	`d=dirac(t)`	Definition of $\delta(t)$.
`int(d,t,-inf,inf)`	`ans = 1`	Confirmation of Equation 2.7, i.e., $\int_{-\infty}^{\infty} \delta(t) = 1$.

6. Ramp Function

The unit-ramp function $r(t)$ is defined in terms of the unit step function $u(t)$ as

$$r(t) = t \cdot u(t) = \begin{cases} t, & t \ge 0 \\ 0, & t < 0 \end{cases}. \tag{2.11}$$

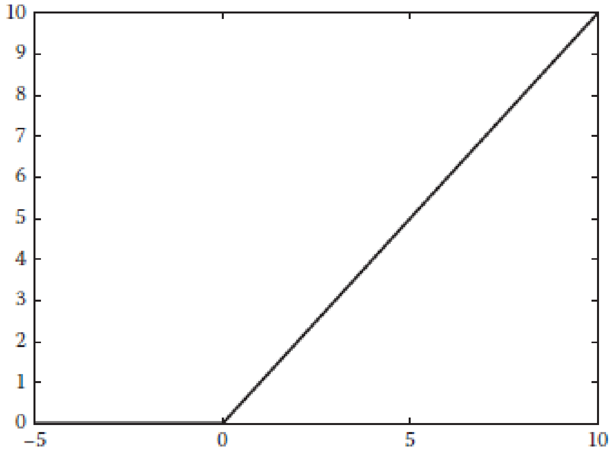
Therefore, in order to define the ramp function, first we have to construct the unit step function.

Commands	Results	Comments
`t=-5:0.1:10;`		Definition of the time interval $-5 \le t \le 10$.
`r=t.*heaviside(t);`		Definition of the ramp function $r(t) = t \cdot u(t)$.
`plot(t,r)`		Graph of $r(t)$.

An alternative way to construct the ramp function is by using the technique of piecewise functions.

Commands	Results	Comments
`t1=-5:.1:-0.1;`		Definition of the first time interval $-5 \le t < 0$.
`t2=0:.1:10;`		Definition of the second time interval $0 \le t \le 10$.
`r1=zeros(size(t1));`		The first part of $r(t)$ that corresponds to time $t1$ is constructed.
`r2=t2;`		The second part of $r(t)$ that corresponds to time $t2$ is constructed.
`t=[t1 t2];`		Time concatenation.

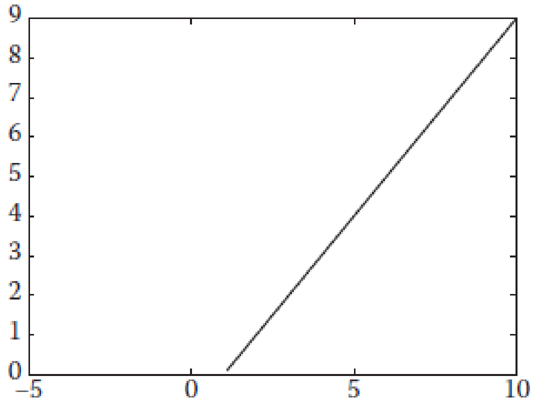
(continued)

Commands	Results	Comments
`r=[r1 r2];`		Function concatenation.
`plot(t,r)`		Graph of the ramp function $r(t)$.

The general form of the (unit) ramp function is

$$r(t-t_0) = (t-t_0)u(t-t_0) = \begin{cases} t-t_0, & t \geq t_0 \\ 0, & t < t_0 \end{cases}. \tag{2.12}$$

Suppose that we want to define and plot the ramp function for $t_0 = 1$, i.e., we want to define and plot the function $r(t - 1)$. The MATLAB implementation is

Commands	Results	Comments
`t=-5:.1:10;`		Definition of the first time interval $-5 \leq t \leq 10$.
`r=(t-1).*heaviside(t-1)`		The function $r(t-1)$ is defined as $r(t-1)=(t-1)u(t-1)$.
`plot(t,r)`		Graph of $r(t-1)$. The function $r(t-1)$ is the function $r(t)$ shifted by 1 unit to the right.

Finally, it is noted that the derivative of the unit-ramp function $r(t)$ is the unit step function $u(t)$, or equivalently the integral of $u(t)$ equals $r(t)$.

Commands	Results	Comments
`syms t`		Definition of the symbolic variable t.
`u=heaviside(t)`	`u=heaviside(t)`	Definition of $u(t)$.
`int(u,t)`	`ans = heaviside(t)*t`	$\int u(t)dt = tu(t) = r(t)$.

7. Rectangular Pulse Function

The rectangular pulse function $pT(t)$ is a rectangular pulse with unit amplitude and duration T. It is defined in terms of the unit step function $u(t)$ as

$$pT(t) = u\left(t + \frac{T}{2}\right) - u\left(t - \frac{T}{2}\right) = \begin{cases} 1, & -T/2 \le t \le T/2 \\ 0, & \text{elsewhere} \end{cases} . \tag{2.13}$$

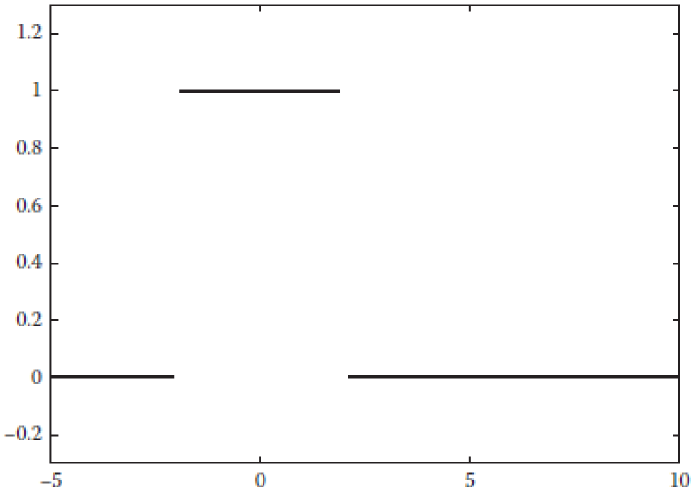
Thus, for the implementation of a rectangular pulse function, two unit step functions have to be first defined. Suppose that the rectangular pulse has duration T = 4, namely the signal is p4(t). Substituting the term T = 4 in (2.13) yields

$$p4(t) = u\left(t + \frac{4}{2}\right) - u\left(t - \frac{4}{2}\right) = u(t+2) - u(t-2). \tag{2.14}$$

The MATLAB implementation is

Commands	Results	Comments
`t=-5:.1:10;`	`t= -5,...,-2, -1.9,...,2, 2.1,...,10`	Definition of the time interval $-5 \le t \le 10$.
`u1=heaviside(t+2);`	`u1=0,...,NaN, 1,...,1 , 1,....1`	Definition of $u(t+2)$.
`u2=heaviside(t-2);`	`u1=0,...0, 0,..., NaN, 1,...,1`	Definition of $u(t-2)$.
`p=u1-u2;`	`p=0,...,NaN, 1,...,NaN, 0,...,0`	The rectangular pulse is built according to (2.14).

(continued)

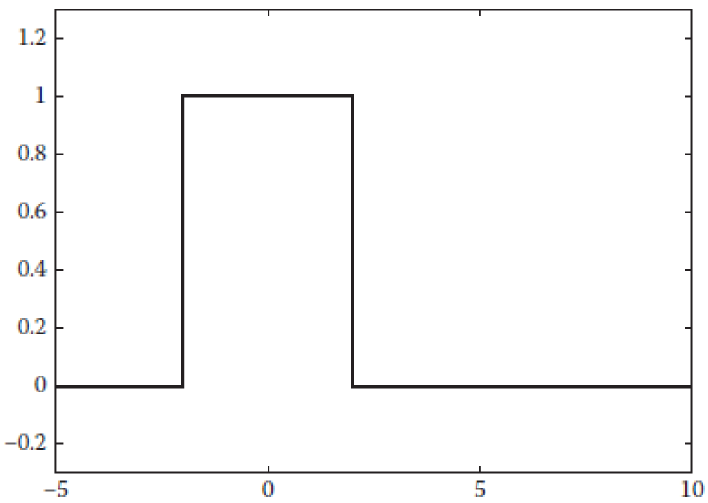
Commands	Results	Comments
`plot(t,p)` `ylim([-0.3 1.3])`		Graph of the rectangular pulse.

The signal $p4(t)$ is a rectangular pulse of unit magnitude and duration T = 4, centered at $t = 0$. A second way that can be used is to define the signal as a three-part function. Thus, in MATLAB we type

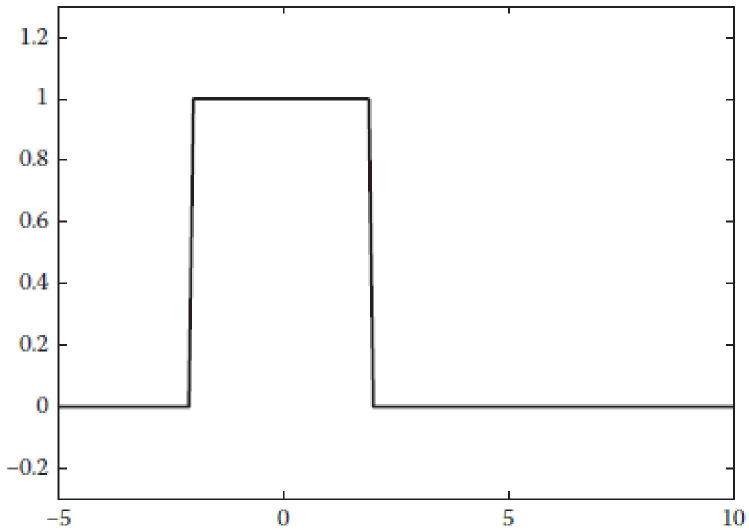
Commands	Results	Comments
`t1=-5:.1:-2;`		Definition of the first time interval $-5 \le t \le -2$.
`t2=-2:.1:2;`		Definition of the second time interval $-2 \le t \le 2$.
`t3=2:.1:10;`		Definition of the third time interval $2 \le t \le 10$.
`p1=zeros(size(t1));`		Function $pT(t)$ is zero for $-5 \le t \le -2$.
`p2=ones(size(t2));`		Function $pT(t)$ is one for $-2 \le t \le 2$.
`p3=zeros(size(t3));`		Function $pT(t)$ is zero for $2 \le t \le 10$.
`t=[t1 t2 t3];`		Time concatenation.

(continued)

(continued)

Commands	Results	Comments
`p= [p1 p2 p3];`		Function concatenation.
`plot(t,p);` `ylim([-0.3 1.3]);`		Graph of the signal and modification of the *y*-axis for better appearance.

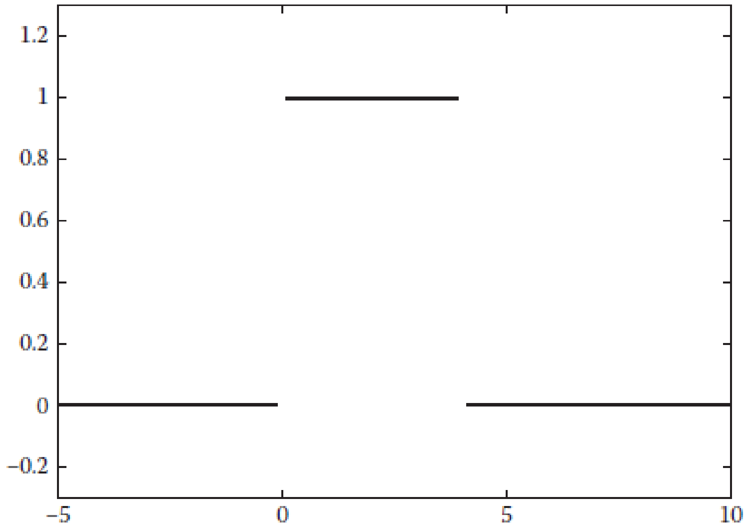
Finally, a third way to create a rectangular pulse signal is by using the command *rectpuls*. Using the syntax *rectpuls*(*t*) we get a rectangular pulse of duration T = 1. Typing *rectpuls(t, a)* results in a rectangular pulse of duration T = a. Therefore, in order to define the signal *p4(t)*, one types

Commands	Results	Comments
`t=-5:.1:10;`		Time definition.
`s=rectpuls(t,4);`		Rectangular pulse of width $T=4$.
`plot(t,s)` `ylim([-.3 1.3])`		Graph of the signal and modification of the *y*-axis for better appearance.

Suppose now that the signal *p4*(t - 2) is needed. According to the definition of rectangular pulse given in (2.13), we get

$$p4(t-2) = u\left(t-2+\frac{4}{2}\right) - u\left(t-2-\frac{4}{2}\right) = u(t) - u(t-4). \quad (2.15)$$

Therefore, in MATLAB we type

Commands	Results	Comments
`t=-5:.1:10`		Definition of the time interval $-5 \le t \le 10$.
`u1=heaviside(t)`		Definition of $u(t)$.
`u2=heaviside(t-4)`		Definition of $u(t-4)$.
`p=u1-u2`		The rectangular pulse is defined according to Equation 2.15.
`plot(t,p)` `ylim([-0.3 1.3])`		Graph of $p4(t-2)$.

We observe that the signal *p4(t - 2)* is similar to *p4(t)* but shifted by 2 units to the right. Hence, by changing appropriately the time, a signal can be shifted to the left or to the right on the horizontal axis.

The End of Practice