Symbols to pulse train

by Johan Östman

This tutorial aims at explaining the procedure of going from symbols to a pulse train by using MATLAB.

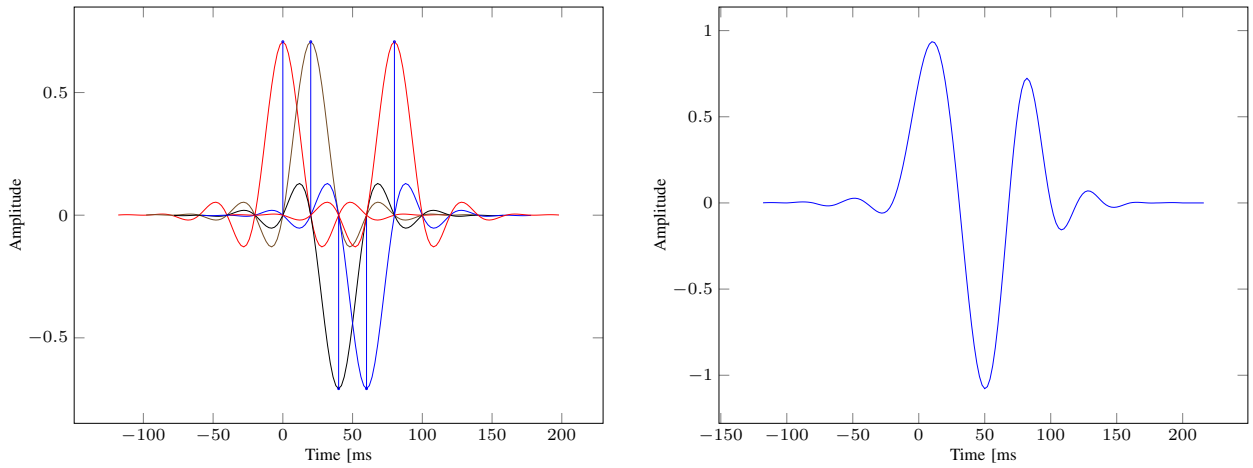
TABLE I: Notation

T_{symb}	=	symbol time
T_{samp}	=	sampling time
N	=	number of symbols to transmit
$x[kT_{\text{symb}}]$	=	symbol to be transmitted at time kT_{symb} where $k = \{0, \ldots, N-1\}$
$p[mT_{\text{samp}}]$	=	value of sampled pulse shape at time mT_{samp}
$y[mT_{\text{samp}}]$	=	value of the final signal at time mT_{samp}

We would like to form our signal to be transmitted according to

$$y[mT_{\text{samp}}] = \sum_{k=0}^{N-1} x[kT_{\text{symb}}]\, p[mT_{\text{samp}} - kT_{\text{symb}}]\,, \quad ,k = \{0, \ldots, N-1\} \tag{1}$$

In words: we would like to replace each symbol that we want to transmit with a pulse $p(t)$ that is sampled with T_{samp} seconds between each sample. This is illustrated in Fig. 1 for the real part of the symbol vector $\mathbf{x} = [1+i, 1+i, -1+i, -1-i, 1-i]$ and RC-pulses. We will now go through the steps needed in order to obtain this result.

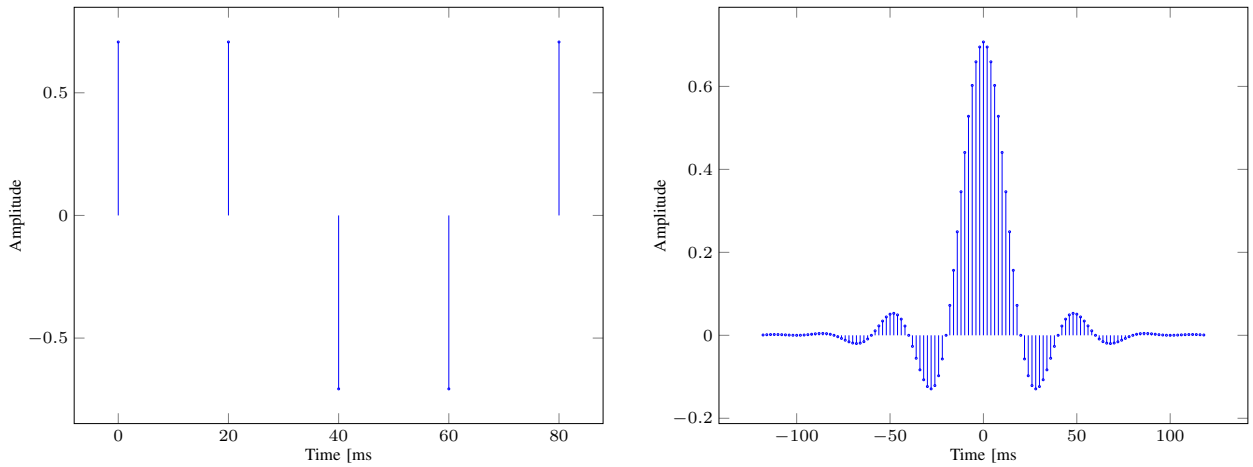


(a) An RC pulse on top of each symbol.

(b) Summation of pulses.

Fig. 1: Illustration of real part of symbols and pulse shape.

For illustration, we will assume that $T_{\text{symb}} = 0.02$ s, $T_{\text{samp}} = 0.002$ s and that a 4-QAM constellation is used. Note that it is very convinient to choose the sampling time as an integer multiple of the symbol time, here we have $\frac{\frac{1}{T_{\text{samp}}}}{\frac{1}{T_{\text{symb}}}} = 10$ samples per symbol. Fig. 3 illustrate the values of the real part of the symbols to be transmitted and the the pulse shape to be used, respectively.

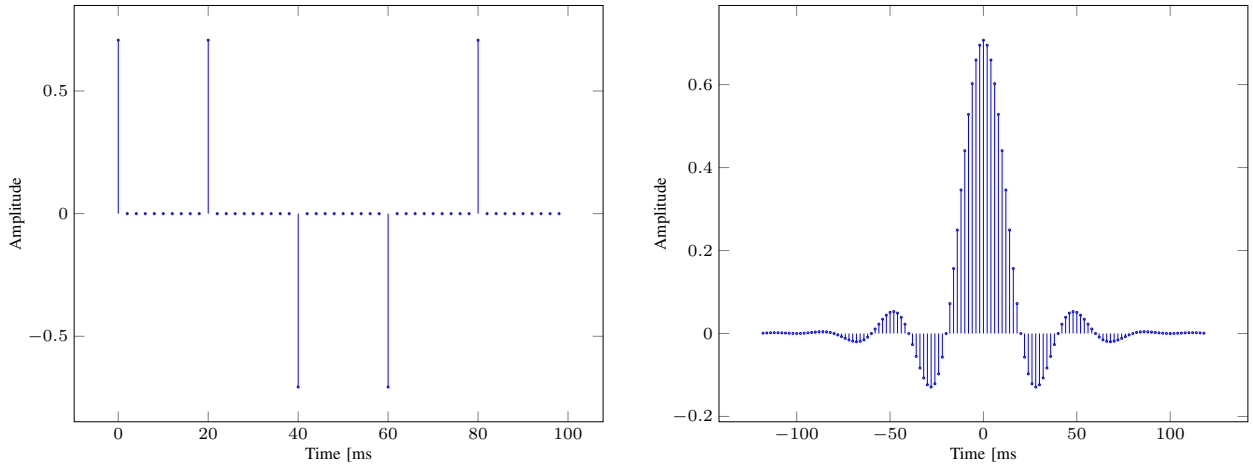


(a) $x\left[kT_{\text{symb}}\right]$ (b) $p[mT_{\text{samp}}]$ for 12 symbol times

Fig. 2: Illustration of symbols and pulse shape.

Now, note that (1) could easily be implemented in a for-loop, however, there is an alternative, more efficient, way of implementing it.

First, we notice that the pulse shape vector, $p[mT_{\text{samp}}]$ and the symbol vector $x[kT_{\text{symb}}]$ does not contain the same number of samples. In order to match the rates of the two signals, we need to upsample the symbol vector. This can be done by calculating how many new samples we need between each sample that we already have and then insert zeros at these positions, e.g., by using MATLAB's command *upsample*. The result is illustrated in Fig. 3. Now, the symbol vector has a value at every sampling instant rather than at every 10 sampling point.



(a) $x[mT_{\text{samp}}]$ (b) $p[mT_{\text{samp}}]$ for 12 symbol times

Fig. 3: Illustration of symbols and pulse shape.

The upsampled signal may be expressed as

$$\begin{cases} x[mT_{\text{samp}}] \neq 0, & \text{for } m = \frac{kT_{\text{symb}}}{T_{\text{samp}}}, \quad k = \{0, \ldots, N-1\} \\ x[mT_{\text{samp}}] = 0, & \text{for } m \neq \frac{kT_{\text{symb}}}{T_{\text{samp}}}, \quad k = \{0, \ldots, N-1\}. \end{cases} \tag{2}$$

Therefore, we may rewrite (1) in terms of the upsampled signal as follows:

$$\begin{aligned} y[mT_{\text{samp}}] &= \sum_{\substack{m=\frac{kT_{\text{symb}}}{T_{\text{samp}}} \\ k=\{0,\ldots,N-1\}}} x[mT_{\text{samp}}]\, p[nT_{\text{samp}} - mT_{\text{samp}}] + \underbrace{\sum_{\substack{m\neq\frac{kT_{\text{symb}}}{T_{\text{samp}}} \\ k=\{0,\ldots,N-1\}}} x[mT_{\text{samp}}]\, p[nT_{\text{samp}} - mT_{\text{samp}}]}_{=0} \\ &= \sum_{m} x[mT_{\text{samp}}]\, p[nT_{\text{samp}} - mT_{\text{samp}}] \\ &= x(mT_{\text{samp}}) * p(mT_{\text{samp}})\,. \end{aligned} \tag{3}$$

The result after performing the convolution is illustrated in Fig. 4. Note that the pulse train is zero in the beginning and in the end due to the transient of the convolution. Note also that this signal is plotted for positive times only.

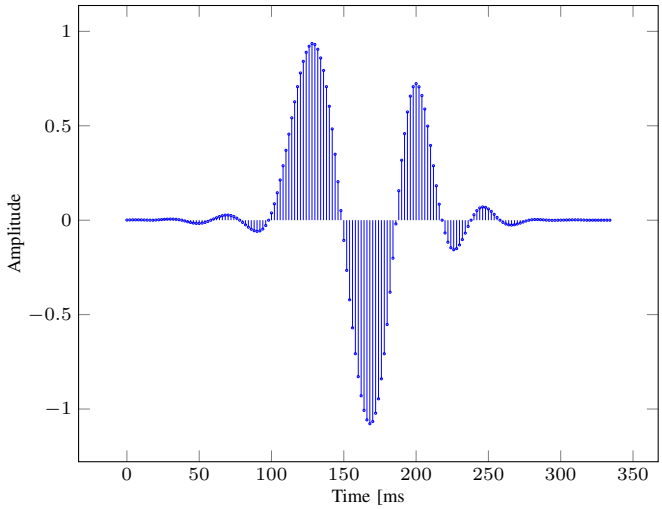


Fig. 4: $y[mT_{\text{samp}}]$