ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
School of Computer and Communication Sciences

Handout 7 Modern Digital Communications
Solutions to Assignment 4 October 8, 2025

SOLUTION 1.

1. A valid answer is that if we delay the input we obtain the same output but delayed. The details follow for those who want to see them.

   With $h(t) = \delta(t - T_0)$ we get

$$R(t) = \sum_j s_j \psi(t - T_0 - jT) + N(t). \tag{1}$$

   Therefore,

$$\begin{aligned} Y(t) = R(t) \star h_{\mathrm{MF}}(t) &= \sum_j s_j \big(\psi(t - T_0 - jT) \star h_{\mathrm{MF}}(t)\big) + N(t) \star h_{\mathrm{MF}}(t) \quad (2)\\ &= \sum_j s_j \int \psi(\alpha - T_0 - jT) h_{\mathrm{MF}}(t - \alpha)\, d\alpha + \int N(\alpha) h_{\mathrm{MF}}(t - \alpha) d\alpha \\ &= \sum_j s_j \int \psi(\alpha - T_0 - jT) \psi^*(\alpha - t)\, d\alpha + \int N(\alpha) \psi^*(\alpha - t) d\alpha. \quad (3) \end{aligned}$$

   Consequently,

$$\begin{aligned} Y(kT + T_0) &= \sum_j s_j \int \psi(\alpha - T_0 - jT) \psi^*(\alpha - T_0 - kT)\, d\alpha + \int N(\alpha) \psi^*(\alpha - T_0 - kT) d\alpha \\ &= \sum_j s_j \int \psi(\alpha' - jT) \psi^*(\alpha' - kT)\, d\alpha' + \int N(\alpha) \psi^*(\alpha - T_0 - kT) d\alpha \quad (4)\\ &\overset{(i)}{=} \sum_j s_j \mathbb{1}\{j = k\} + Z_k = s_k + Z_k. \quad (5) \end{aligned}$$

   In the above (i) follows because ψ is a Nyquist pulse.

2.

   (a)

$$\begin{aligned} R_p^*(-t) &= \left(\int p(\alpha - t) p^*(\alpha) d\alpha \right)^* \\ &= \int p^*(\alpha - t) p(\alpha) d\alpha \\ &= \int p^*(\alpha') p(t + \alpha') d\alpha' = R_p(t). \end{aligned}$$

   (b) Note that $R_p(t) = \langle p(\alpha + t), p(\alpha) \rangle$. Thus $R_p(0) = \|p\|^2$. Moreover, using Cauchy–Schwarz inequality we have

$$|R_p(t)| = |\langle p(\alpha + t), p(\alpha) \rangle| \le \|p(\alpha + t)\| \|p(\alpha)\| = \|p\|^2.$$

SOLUTION 2.

1. See the provided MATLAB/Python routine.

2. (a) From the assignment statement we know that the received signal has the form $R(t) = p(t - T_0) + s_{\mathrm{data}}(t - T_0) + N(t)$. Hence,

$$\rho(\upsilon) = \langle R(t), p(t - \upsilon) \rangle = R_p(\upsilon - T_0) + \langle s_{\mathrm{data}}(t - T_0), p(t - \upsilon) \rangle + \langle N(t), p(t - \upsilon) \rangle.$$

When $p(t) = s_0\psi(t)$, $R_p(\upsilon - T_0) = |s_0|^2 R_\psi(\upsilon - T_0)$ and

$$\langle s_{\text{data}}(t - T_0), p(t - \upsilon)\rangle = \left\langle \sum_{k\neq 0} s_k \psi(t - T_0 - kT), s_0\psi(t - \upsilon) \right\rangle \tag{6}$$

$$= \sum_{k\neq 0} s_k s_0^* \langle \psi(t - T_0 - kT), \psi(t - \upsilon)\rangle \tag{7}$$

$$= \sum_{k\neq 0} s_k s_0^* R_\psi(\upsilon - T_0 - kT). \tag{8}$$

Putting everything together we obtain:

$$\rho(\upsilon) = \langle R(t), p(t-\upsilon)\rangle = |s_0|^2 R_\psi(\upsilon - T_0) + \sum_{k>0} s_k s_0^* R_\psi(\upsilon - T_0 - kT) + \langle N(t), p(t-\upsilon)\rangle.$$

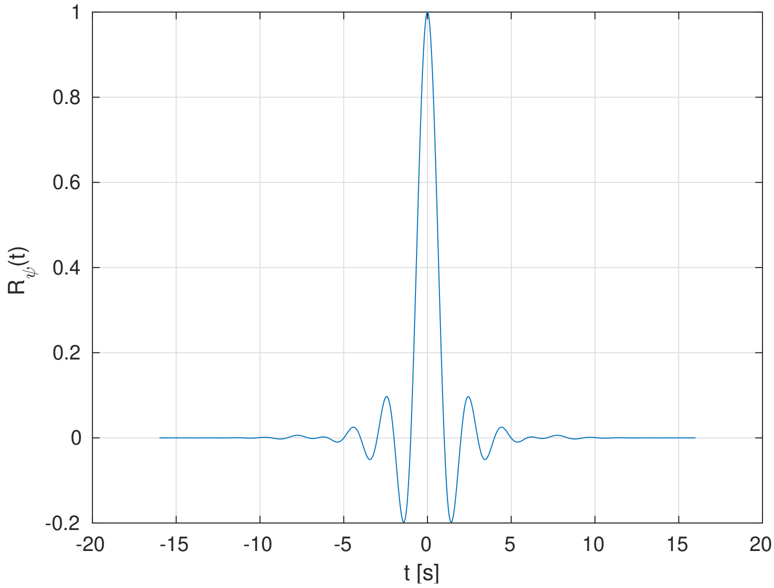
(b) If ψ is a Nyquist pulse, $R_\psi(kT) = \langle \psi(\alpha + kT), \psi(\alpha)\rangle = 0$ for $k \neq 0$. Using this in (a) the result follows.

(c) Proceeding as in the previous point, we obtain

$$\rho(T_0 + nT) = s_n s_0^* R_\psi(0) + \int N(\alpha) s_0^* \psi^*(\alpha - T_0 - nT) d\alpha,$$

which has indeed a similar form as $\rho(T_0)$.

(d) The self-similarity function of ψ (the root-raised-cosine pulse) used in our system looks as follows:



(e) Testing this solution, we see that the estimations are way off from the actual channel delay. The main reason is that while the contribution of data symbols to the inner product is zero for υ close to T_0, in general the same is happening for many other values of υ (i.e., $\upsilon = T_0 + nT$, as we have seen in the previous question). This is confirmed by plotting the result of the convolution in the function `my_estimateTau`: one can quickly see that there is no clear maximum for $\rho(\upsilon)$.

3. (a) We have

$$R_p(t) = \left\langle \sum_i B_i \psi(\alpha + t - iT), \sum_j B_j \psi(\alpha - jT) \right\rangle \tag{9}$$

$$= \sum_i \sum_j B_i B_j^* \langle \psi(\alpha + t - iT), \psi(\alpha - jT)\rangle \tag{10}$$

$$= \sum_{i,j} B_i B_j^* R_\psi(t + (j - i)T) \tag{11}$$

Taking the expectation of the above, and using the fact that B_is are i.i.d. BPSK symbols,

$$E\left[R_p(t)\right] = \sum_{i,j} E\left[B_i B_j^*\right] R_\psi(t + (j-i)T) \tag{12}$$

$$= \sum_{i,j} \mathbb{1}\{i = j\} R_\psi(t + (j-i)T) = l R_\psi(t). \tag{13}$$

(In the last step we used the fact that both dummy variables i and j go over the range 0 to $l-1$ in the summations.)

(b)

$$\langle q(t), p(t-\upsilon)\rangle = \sum_i B_i^* \langle q(t), \psi(t-\upsilon-kT)\rangle \tag{14}$$

Therefore,

$$E\left[\langle q(t), p(t-\upsilon)\rangle\right] = \sum_i E\left[B_i^*\right] E\left[\langle q(t), \psi(t-\upsilon-kT)\right] = 0. \tag{15}$$

(where we have used the independence of B_i and $q(t)$ and the fact that $E\left[B_i\right] = 0$.)

Running the system with the preamble generated from the P/N sequence, we see that the channel delay is usually estimated accurately.

Solution 3.

1. The signal at the output of the matched filter is

$$\begin{aligned} Y(t) &= R(t) \star h_{\mathrm{MF}}(t) \\ &= R(t) \star \psi^*(-t) \\ &= \int R(\alpha)\psi^*(\alpha - t) d\alpha \\ &= \langle R(\alpha), \psi(\alpha - t)\rangle. \end{aligned}$$

2.

$$\begin{aligned} \rho(\upsilon) &= \langle R(t), p(t-\upsilon)\rangle \\ &= \left\langle R(t), \sum_k b_k \psi(t-\upsilon-kT)\right\rangle \\ &= \sum_k b_k^* \left\langle R(t), \psi(t-\upsilon-kT)\right\rangle \\ &= \sum_k b_k^* Y(\upsilon + kT). \end{aligned}$$

Hence, $\rho(mT_s) = \sum_k b_k^* Y(mT_s + kT)$. This shows that we can obtain $\rho(mT_s)$ from the matched filter output.

To obtain all the values of $\rho(mT_s)$ efficiently, we can proceed as follows. Letting $T = NT_s$,

$$\begin{aligned} \rho[m] \coloneqq \sqrt{T_s}\rho(mT_s) &= \sum_k b_k^* \sqrt{T_s} Y(mT_s + kT) \\ &= \sum_k b_k^* Y[m + kN] \\ &= \sum_k \hat{b}_k^* Y[m + k], \end{aligned} \tag{16}$$

where $\hat{b}_k$ is the upsampled preamble sequence with upsampling factor N, i.e.,

$$\hat{b}_k \coloneqq \begin{cases} b_{k/N} & \text{if } k/N \text{ is an integer,} \\ 0 & \text{otherwise.} \end{cases}$$

Recall that in MATLAB/Python we can easily implement the upsampling, and use the convolution/correlation functions to compute the correlation (16) for all values of m.