Université Catholique de Louvain

LELEC2795
Laboratories

François De Saint Moulin
Florian Quatresooz
Guillaume Thiran

September 19, 2023

Beyond the noise - estimating and correcting channel effects

When analyzing a communication link, the impact of the noise on the communication is of crucial interest. Such noise is usually modelled as an additive white Gaussian noise: when transmitting a signal $x(t)$, the received signal reads as $z(t) = x(t) + w(t)$ with $w(t)$ an AWGN. While this AWGN model provides excellent insights (Matched filter, Nyquist criterion, BER curves), it fails to model some impairments of the channel. First, at the receiver side, one must recover the timing of the signals before being able to perform sampling and decision. Secondly, contrarily to the AWGN where a single perfect propagation path is assumed, real channels are usually made of several paths with different characteristics (delay, phase shift).

The goals of these labs will be to understand 1) the impacts of these impairments, 2) the methods employed to counteract them, 3) the pros and cons of these methods.

The labs will be divided as follows:

- **Preliminary Lab: AWGN Numeric Transmission Chain.** The basic structure of the communication chain (Symbol mapping, pulse shaping, matched filtering, decision) enables to study the impact of the noise on the communication.
- **Lab 1: Synchronisation - Symbol Timing Recovery in Narrowband Channels.** As the name hints, the goal of this lab is to recover the symbol timing, which enables to sample at the right time instants[1].
- **Lab 2: Channel Estimation and Equalisation.** In case of multipath propagation, several versions of the signal might be received. In order to achieve good performances, one needs to remove the effect of the channel, this operation being called the *equalisation*. Beforehand, channel estimation needs obviously to be performed.
- **Lab 3: OFDM Modulation and Frequency Domain Equalisation.** As you will see in Lab 2, equalisation is not always an easy task, and can lead to severe degradation of the communication. A modulation technique called Orthogonal Frequency Division Multiplexing (OFDM) circumvents this problem. This advantage, along with many others, explains why OFDM is often used today, in 4G networks for instance.
- **Supplementary Lab: Synchronisation - Frame synchronisation and Frequency Offset Correction.** The goal of this lab is to determine where the frame begins in order to be able to extract training sequences, etc. Moreover, it can happen the frequency of the emitter and receiver differs a bit, in which case this frequency offset must be corrected.

Lab schedule

- Week 1 and week 2: *Lab 1 - Synchronisation: Symbol Timing Recovery in Narrowband Channels*.
  For the students who have never used LabVIEW before, this lab is replaced by the *Preliminary Lab - AWGN Numeric Transmission Chain*.
- Week 6 and week 9: *Lab 2 - Channel Estimation and Equalisation*.
- Week 10 and week 12: *Lab 3 - OFDM modulation and Frequency Domain Equalisation*.
- Week 14: Final oral presentation.

Organization

During this semester, you will have each week either an exercise session, either a lab session. For the lab sessions, you should form groups of 3 students and register your group here. The students who have never used LabVIEW before should form groups together as the lab schedule is adapted for them.

During the lab sessions, which will take place in the Marconi Lab, teaching assistants will be present to guide you through the labs. We therefore strongly advise you to prepare the labs **before the sessions**.

[1]Note that the word *Synchronisation* encompasses sometimes also other methods which will not be covered in these labs, such as the frequency offset correction.

The lab statements[2] are structured as follows: first, the theoretical material instrumental to the understanding of the lab is presented. Then, the corresponding blocks which should be implemented in LabVIEW are described. Finally, a few experiments which enable to compare the theoretical foundation with the output of the blocks are detailed. Throughout the labs, a series of questions is stated, to which you should be able to answer at the oral presentation.

Evaluation

The evaluation takes the form of an oral presentation followed by a question and answer session. This presentation will focus on the lab results and conclusions, both on the theoretical and experimental material of the labs. However, the goal of this presentation is not to evaluate the LabVIEW part of the project. This presentation will count for 25% of the final grade of the course.

Document outline

[2]The lab statements material and LabVIEW project are directly inspired from the manual “HEATH R., D. Ph., E. P., Digital Communications: Physical Layer exploration lab using the NU USRP platform", which can be found on Moodle.

Introduction and LabVIEW installation

Along the years, lots of ways of transmitting information have been invented. On the one hand, fully analog schemes can be used, for instance Amplitude Modulation (AM) and Frequency Modulation (FM). In such schemes, the idea is to modulate the continuous message signal up to a higher frequency[3] and to demodulate it at the reception side. On the other hand, when dealing with discrete data (i.e. bits of information), part of the process can be made digital: they take place in the discrete world. Yet, the wireless transmission remains analog as it is made of waves propagating through a medium.

The transition between the digital world and the analog world can take place at different steps of the communication chain. Among others, one can push the digital operations at their maximum by restricting the analog part to its minimum: i.e. the up-conversion to higher frequencies and the transmission. In such case, the communication structure is called a Software Defined Radio (SDR) as represented in Figure A.1.

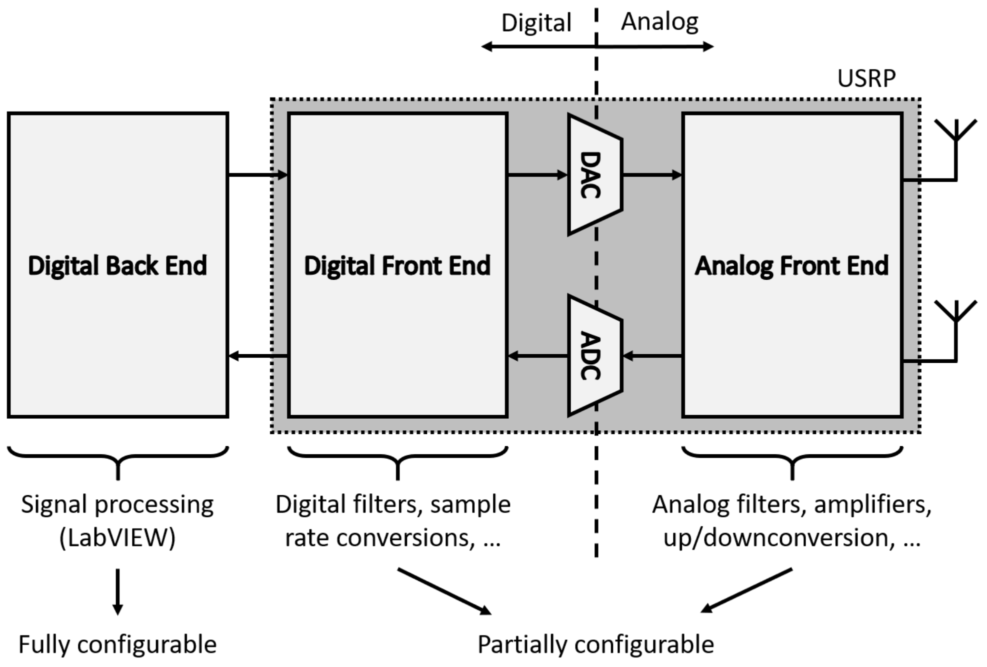

Digital
Analog
USRP
Digital Back End
Digital Front End
DAC
ADC
Analog Front End
Signal processing (LabVIEW)
Digital filters, sample rate conversions, ...
Analog filters, amplifiers, up/downconversion, ...
Fully configurable
Partially configurable

Figure A.1: Software Defined Radio.

1: Software Defined Radio

An SDR is a radio communication system where most of the transmission blocks are implemented in software instead of hardware, contrariwise to usual communication systems. Typically, an SDR is composed of a digital back-end (often a computer) and an analog part which is in our case a Universal Software Radio Peripheral (USRP). The digital back-end is programmed to execute all the signal processing needed to generate from scratch a digital transmit signal in baseband, or to process a digital received signal also in baseband. This part is fully configurable and we will use in these labs the LabVIEW software. The USRP makes the conversion of the signals from the digital to the analog domain, and up-converts these signals from the baseband to the radio-frequencies. Inversely, it can also down-convert a received signal from the radio-frequencies to the baseband, and make the conversion from the analog domain to the digital domain.

USRPs can also handle some additional operations in the digital domain (filtering, rate conversions, etc.). This part is partially configurable. This structure enables to use a wide range of different communication protocols or signal processing algorithms thanks to the flexibility of the software.

[3]By the way, do you remember the reasons why one needs to consider high frequencies?

In these labs, the USRP-2920 model from National Instrument (NI) will be used. For the transmission, the digital to analog converters have a resolution up to 16 bits, and can achieve a maximum sample rate of 25Msps. For the reception, complex I and Q signals in baseband are obtained at a rate of 100Msps for carrier frequencies between 50MHz and 2.2GHz. Then, they are digitally downsampled to a rate between 200ksps and 25Msps, with 14 bits of resolution.

2: The LabVIEW software

Introduction

Laboratory Virtual Instrument Engineering Workbench (LabVIEW) is a system engineering software dedicated for tests, measurements and control with rapid access to data insights and National Instrument (NI) hardware. It is a visual programming language based on dataflow. The dataflow programming is based on data availability: a function will be executed if there is enough data available at its inputs. The structure is constructed as a graphical block diagram, where differents blocks are connected through wires. These wires propagate data from one block to another as soon as there is an available input data. If there is multiple data available simultaneously, LabVIEW can execute inherently in parallel, exploiting multi-processing and multi-threading hardware.

Every LabVIEW program is called a Virtual Instrument (VI). Each VI has three components:

- a front panel, built using controls and indicators. Controls are inputs which allow the user to modify some parameters. Indicators are outputs, displaying the results produced by the VI;
- a block diagram, composed of basic LabVIEW blocks (multiply, divide, square, ...) processing different data types, and other VIs called sub-VIs;
- a connector pane, where the block containing the VI can be personalised. The inputs/outputs of the block are also specified here by the user.

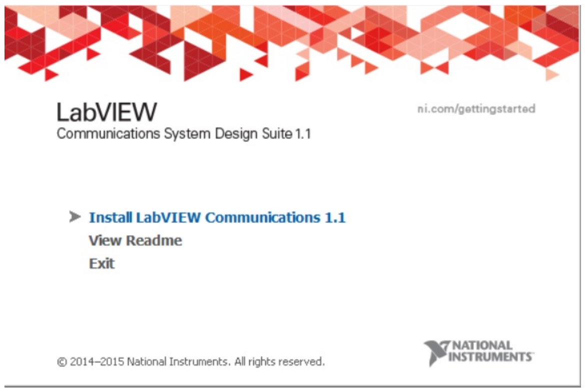
For these labs, the software *LabVIEW Communication System Design Suite* will be helpful to prototype easily a wireless system.

Download LabVIEW

Download the LabVIEW installation files by clicking here. The installation file size is nearly 6.5GB, so it can take time and it is advised to download it on a stable connection.

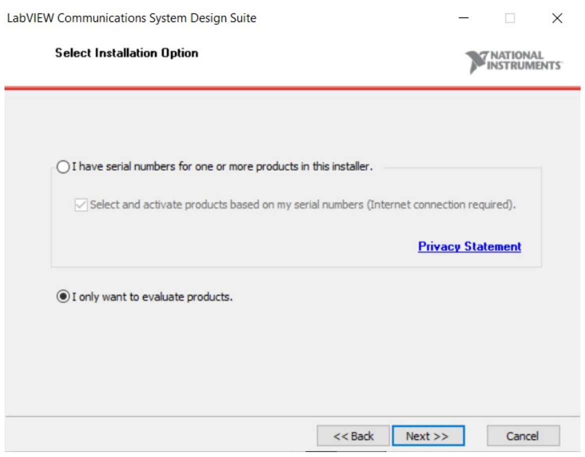
Installation of LabVIEW

1. Enter the downloaded folder and double-click on the `autorun` application.


LabVIEW
Communications System Design Suite 1.1
ni.com/gettingstarted
Install LabVIEW Communications 1.1
View Readme
Exit
© 2014–2015 National Instruments. All rights reserved.
NATIONAL INSTRUMENTS

2. Click on "Install LabVIEW Communications 1.1", then click next.

3. Choose "I only want to evaluate product".


LabVIEW Communications System Design Suite

Select Installation Option

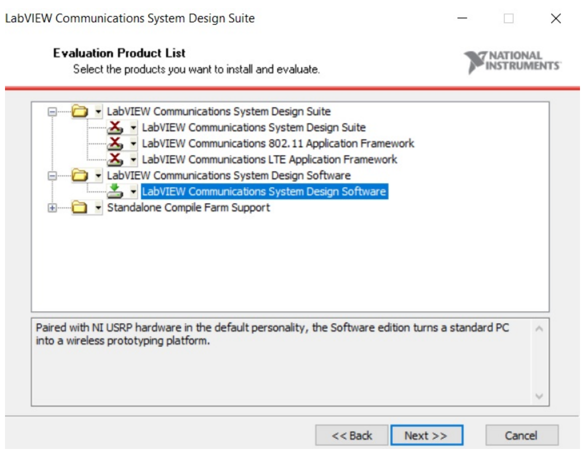
( ) I have serial numbers for one or more products in this installer.

[✓] Select and activate products based on my serial numbers (Internet connection required).

Privacy Statement

(●) I only want to evaluate products.

4. We only need to install "LabVIEW Communications System Design Software". To do so, you should see a green arrow going down next to "LabVIEW Communications System Design Software" **only**.


LabVIEW Communications System Design Suite

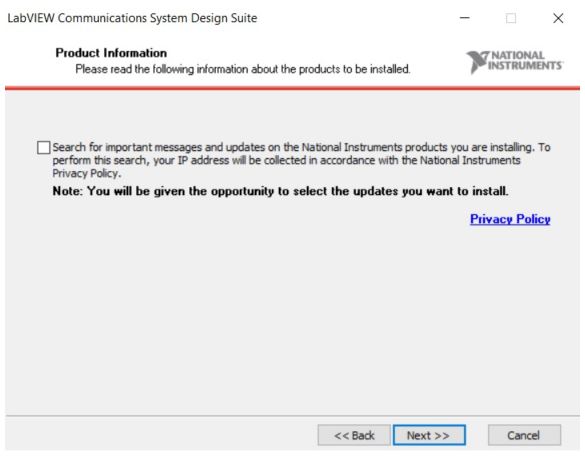
Evaluation Product List

Select the products you want to install and evaluate.

- LabVIEW Communications System Design Suite
  - LabVIEW Communications System Design Suite
  - LabVIEW Communications 802.11 Application Framework
  - LabVIEW Communications LTE Application Framework
- LabVIEW Communications System Design Software
  - LabVIEW Communications System Design Software
- Standalone Compile Farm Support

Paired with NI USRP hardware in the default personality, the Software edition turns a standard PC into a wireless prototyping platform.

5. Click next.
6. Uncheck the box and click next.


LabVIEW Communications System Design Suite

Product Information

Please read the following information about the products to be installed.

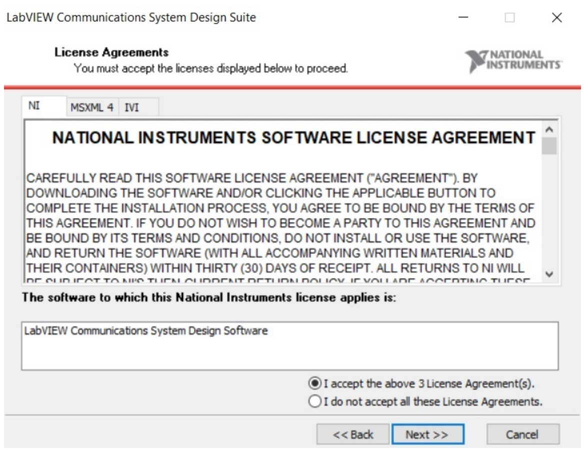
[ ] Search for important messages and updates on the National Instruments products you are installing. To perform this search, your IP address will be collected in accordance with the National Instruments Privacy Policy.

Note: You will be given the opportunity to select the updates you want to install.

Privacy Policy

7. Choose the installation directory and click next.

8. Accept the license agreements and click next.



LabVIEW Communications System Design Suite

License Agreements
You must accept the licenses displayed below to proceed.

NI | MSXML 4 | IVI

NATIONAL INSTRUMENTS SOFTWARE LICENSE AGREEMENT

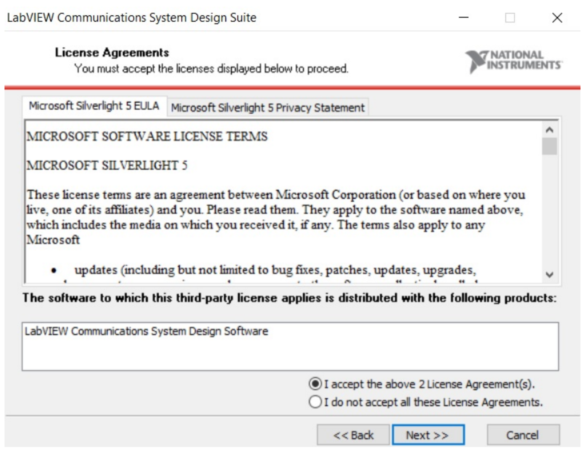
CAREFULLY READ THIS SOFTWARE LICENSE AGREEMENT ("AGREEMENT"). BY DOWNLOADING THE SOFTWARE AND/OR CLICKING THE APPLICABLE BUTTON TO COMPLETE THE INSTALLATION PROCESS, YOU AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT WISH TO BECOME A PARTY TO THIS AGREEMENT AND BE BOUND BY ITS TERMS AND CONDITIONS, DO NOT INSTALL OR USE THE SOFTWARE, AND RETURN THE SOFTWARE (WITH ALL ACCOMPANYING WRITTEN MATERIALS AND THEIR CONTAINERS) WITHIN THIRTY (30) DAYS OF RECEIPT. ALL RETURNS TO NI WILL

The software to which this National Instruments license applies is:

LabVIEW Communications System Design Software

[✓ I accept the above 3 License Agreement(s).]
I do not accept all these License Agreements.

9. Accept again the license agreements and click next.



LabVIEW Communications System Design Suite

License Agreements
You must accept the licenses displayed below to proceed.

Microsoft Silverlight 5 EULA | Microsoft Silverlight 5 Privacy Statement

MICROSOFT SOFTWARE LICENSE TERMS

MICROSOFT SILVERLIGHT 5

These license terms are an agreement between Microsoft Corporation (or based on where you live, one of its affiliates) and you. Please read them. They apply to the software named above, which includes the media on which you received it, if any. The terms also apply to any Microsoft

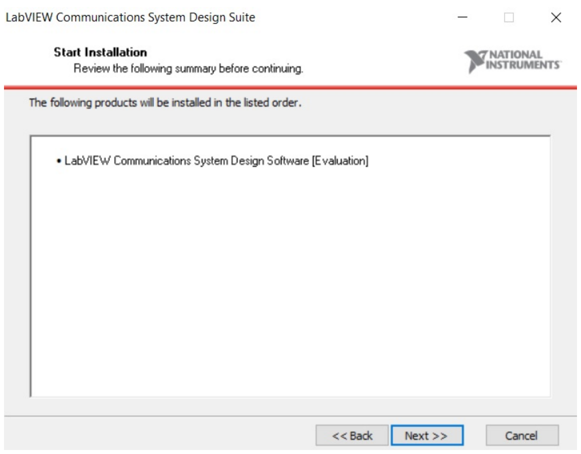
- updates (including but not limited to bug fixes, patches, updates, upgrades,

The software to which this third-party license applies is distributed with the following products:

LabVIEW Communications System Design Software

[✓ I accept the above 2 License Agreement(s).]
I do not accept all these License Agreements.

10. The following window should appear:



LabVIEW Communications System Design Suite

Start Installation
Review the following summary before continuing.

The following products will be installed in the listed order.

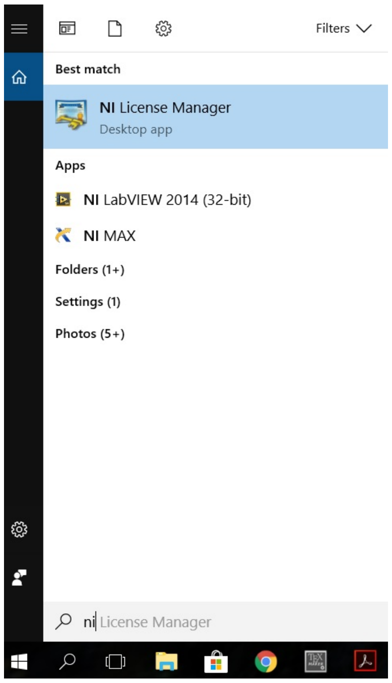
- LabVIEW Communications System Design Software [Evaluation]

11. Click next.

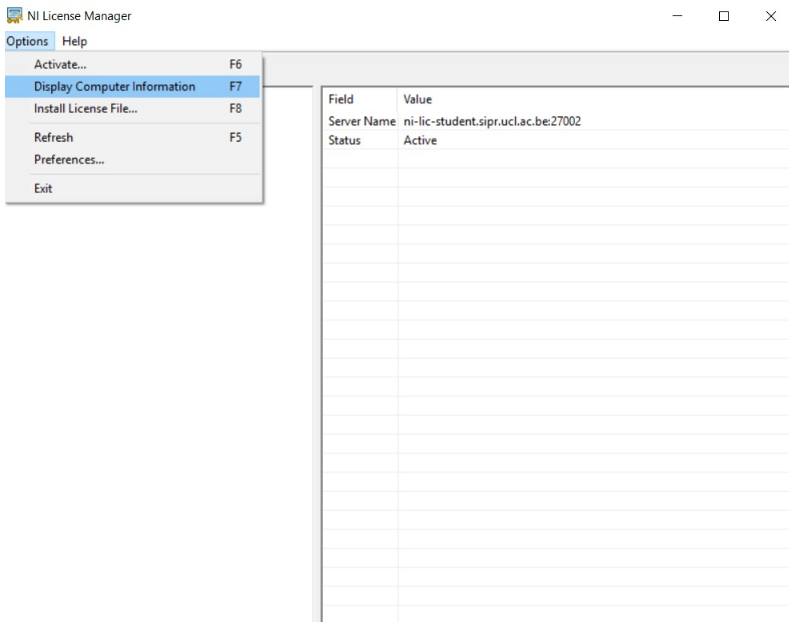
12. Wait for the installation. Once the installation completed, you may need to restart your computer.

Set up a LabVIEW license

1. Open the NI License Manager application.



2. Click on "Options", then click on "Display the computer informations".

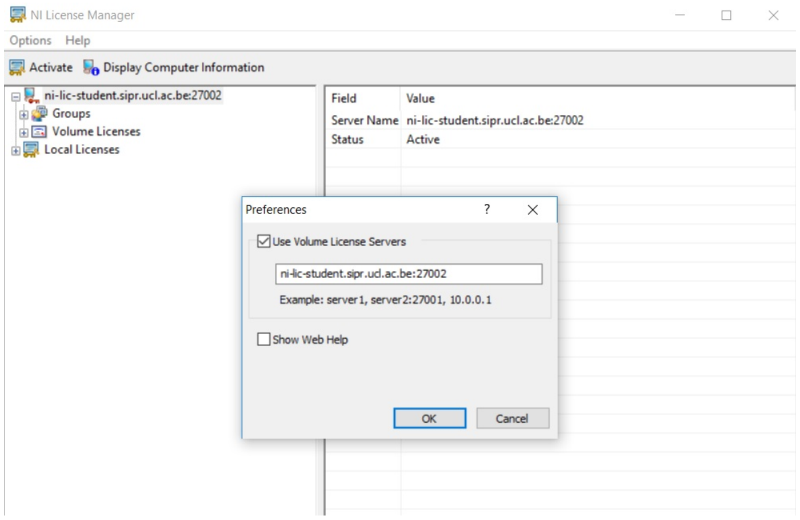


Field	Value
Server Name	ni-lic-student.sipr.ucl.ac.be:27002
Status	Active

3. Copy the name and ID of your computer. Send them to equipe-sdj@listes.uclouvain.be with the following informations:
   - Your name and surname;
   - Your UCLouvain e-mail address;
   - Your study year.

They will authorize your computer to connect to the token server. Once the connection is established, you will have access to LabVIEW for 15 days without connection. You only need to reconnect to be able to use the software.

4. As a last step, open again the License Manager and click on “Options" then “Preferences". Check “Use Volume License Servers" and insert: `ni-lic-student.sipr.ucl.ac.be:27002`.



Field	Value
Server Name	ni-lic-student.sipr.ucl.ac.be:27002
Status	Active

☑ Use Volume License Servers

ni-lic-student.sipr.ucl.ac.be:27002

Example: server1, server2:27001, 10.0.0.1

☐ Show Web Help

Congratulations, LabVIEW should now run on your computer

LabVIEW Tutorials

A lot of tutorials are available to learn how to use LabVIEW. Open *LabVIEW Communications System Design Software* and click on *Learn* > *Programming Basics*. Don’t hesitate to take some time to become more familiar with the software! However, note that the goal of these labs are to understand the presented concepts rather than to become an expert in LabVIEW coding.

3: The analog part: up and down conversion

Generally speaking, the goal of the LabVIEW software is to generate a complex baseband waveform from the input bit sequence. For example, such a signal can be obtained by mapping bits to complex symbols, which are themselves converted into pulse-shapes (complex pulse modulation). If the symbol sequence $I[n] = a[n] + jb[n]$ is generated, with a pulse shape $g_{\text{tx}}(t)$, the complex baseband waveform to transmit is given by

$$x(t) = \sum_n I[n] g_{\text{tx}}(t - nT),$$

where T is the period between two successive symbols. The analog part of the SDR then works on this continuous signal. As you have seen during previous telecommunication courses, a complex baseband signal $x(t) = x_I(t) - jx_Q(t)$ can be up-converted to radio-frequencies around a carrier frequency f_c and give the signal

$$x_{\text{RF}}(t) = \text{Re}\left\{x(t)\, e^{j2\pi f_c t}\right\} = x_I(t)\cos(2\pi f_c t) + x_Q(t)\sin(2\pi f_c t),$$

as represented in Figure A.2. Note that it is always assumed that the carrier frequency is much higher than the frequency content of x. At the receiver side, the inverse operation is performed on the received signal $y_{\text{RF}}(t)$ thanks to mixers and Low-Pass Filters (LPF). With an ideal LPF of cut-off frequency f_c,

$$\begin{aligned} & 2y_{\text{RF}}(t)\cos(2\pi f_c t) = y_I(t) + y_I(t)\cos(4\pi f_c t) + y_Q(t)\sin(4\pi f_c t) \\ \Rightarrow\quad & \text{LPF}\left\{2y_{\text{RF}}(t)\cos(2\pi f_c t)\right\} = y_I(t), \end{aligned}$$

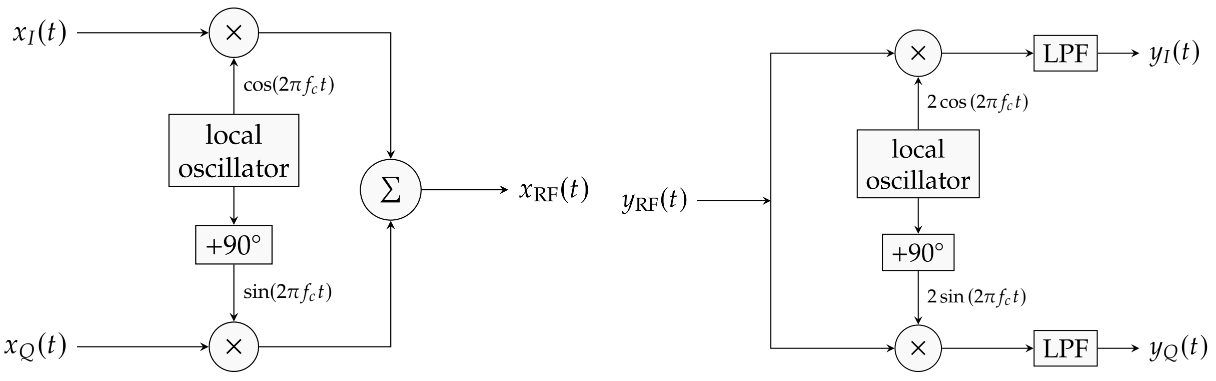


Figure A.2: Analog conversions between baseband and radio-frequencies.

and

$$\begin{aligned} & 2y_{\mathrm{RF}}(t)\sin(2\pi f_c t) = y_I(t)\sin(4\pi f_c t) + y_Q(t) - y_Q(t)\cos(4\pi f_c t) \\ \Rightarrow\ & \mathrm{LPF}\left\{2y_{\mathrm{RF}}(t)\sin(2\pi f_c t)\right\} = y_Q(t). \end{aligned}$$

4: The analog part: the communication channel

Thanks to these up and down conversion, one can instead study the complex baseband signals $x(t)$ and $y(t)$ instead of the real ones $x_{\mathrm{RF}}(t)$ and $y_{\mathrm{RF}}(t)$. The way $x(t)$ and $y(t)$ are related depends on the type of channel we consider.

If the channel is modelled as an Additive White Gaussian Noise (AWGN) channel, the received signal $y(t)$ is equal to the transmit signal $x(t)$ affected by a white Gaussian noise $w(t)$:

$$y(t) = x(t) + w(t).$$

Such channel is studied in the Preliminary Lab, where, among others, the impact of the pulse shaping is studied.

However, channels in real wireless applications suffer from additional impairments which are not modelled with an AWGN channel. A first improvement of the model is obtained by introducing a delay of transmission τ an attenuation a and a phase change ϕ in baseband:

$$y(t) = ae^{j\phi}x(t-\tau) + w(t).$$

This channel model is called *flat fading channel*. Owing to the presence of a delay, the received signal will not be sampled at the right time instants. Therefore, this delay (or at least its fractional part in regard to the symbol period) must be estimated and compensated to limit the induced distortions. This is the topic of Lab 1.

Next, the presence of multiple propagation paths between the transmitter and receiver can also be introduced. Denoting respectively by τ_i, a_i and ϕ_i the delay of transmission, the attenuation and the phase change of the path i, the received signal is given by

$$y(t) = \sum_i a_i e^{j\phi_i} x(t-\tau_i) + w(t).$$

This kind of channel is called a *frequency selective channel*. In that situation, the presence of multiple copies of the transmitted signal with different phase changes and delays can induce destructive interference, but also corrupt the received signal such that the transmitted symbol sequence could not be recovered. In such situations, the effect of the multi-path propagation should be compensated, and this is analysed in Lab 2.

Another way of dealing with frequency selective channels is to develop other modulation techniques, as for instance the Orthogonal Frequency Division Multiplexing (OFDM) studied in Lab 3. This technique has been widely adopted in last generations of cellular networks thanks to its interesting features.

Finally, yet another impairment can affect the communications: a mismatch of the frequencies generated by the local oscillators of the transmitter and receiver will induce an error during the conversions from baseband to radio-frequencies and vice-versa. In baseband, denoting by $f_{c,\mathrm{tx}}$ and $f_{c,\mathrm{rx}}$ the carrier frequencies respectively generated at the transmitter and receiver, for a frequency offset $f_0 = f_{c,\mathrm{tx}} - f_{c,\mathrm{rx}}$, the received signal becomes

$$y(t) = e^{j2\pi f_0 t} \sum_i a_i e^{j\phi_i} x(t - \tau_i) + w(t).$$

This frequency offset induces a distortion which should be compensated. Additionally, when the delay is greater than the symbol period, even if the fractional part is estimated and compensated such that the signal is sampled at the right time instants, the start of the transmitted sequence should be detected to know when the symbol sequence starts. These two issues are handled in the Supplementary Lab. The effect of the frequency offset on OFDM systems will also be studied in the corresponding lab.

5: The digital part: the signal processing

In these labs, you will study several part of the digital signal processing chain. To help you visualize the different operations, Figures A.4 and A.5 illustrate a complete numeric transmitter and receiver, where all the blocks implemented during these labs are presented. Note that an SDR architecture is adopted: the conversion from the analog to digital domain (and vice-versa) and the up/down-conversion are visible on the schematic, and the pulse shaping/matched filtering are implemented in the digital domain.

Note however that the LabVIEW project you are provided with is equipped with a channel simulator, in the sense that the channel impairments are generated numerically. We will mainly work with this simulator, as it enables to precisely monitor the impairments and the results. In some cases, a real transmission through USRPs will be analyzed, in which case it is mandatory to use LabVIEW on the Marconi/Faraday computers.

6: Tips

- **Contrarily to most programming languages, when you execute a LabVIEW program, it is not saved automatically. You should therefore save your work manually very frequently throughout the labs.**
- Contrarily to Matlab or Python, LabVIEW is not designed and optimized for matrix computations. On the contrary, it is suited for on-line sequential operations as a real transmission chain would operate. When building the different blocks, remember that one should use for loops instead of big matrix operations. Not doing so will result in very slow execution times.
- The *TX sample rate* and *RX sample rate* are the sample rates of the oversampled signal. Letting $f_{s,\mathrm{tx}}$ and $f_{s,\mathrm{rx}}$ be the sample rates, and L and M the oversample factors of the transmitter and receiver, the symbol duration T can thus be computed as

  $$T = \frac{L}{f_{s,\mathrm{tx}}} = \frac{M}{f_{s,\mathrm{rx}}}.$$

  As T needs to be the same at the emitter and receiver, the oversample factor of the receiver needs to be chosen in function of the transmitter and receiver sample rates, as well as the transmitter oversample factor.
- Only **even oversample factors** are allowed in the LabVIEW implementation.

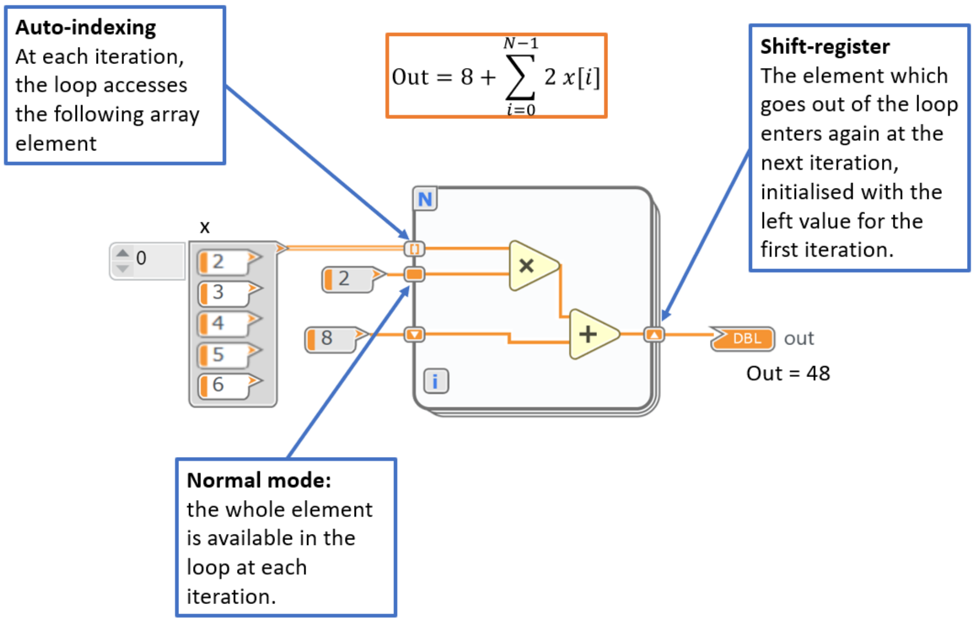

Auto-indexing
At each iteration, the loop accesses the following array element
$\text{Out} = 8 + \sum_{i=0}^{N-1} 2\,x[i]$
Shift-register
The element which goes out of the loop enters again at the next iteration, initialised with the left value for the first iteration.
x
0
2
3
4
5
6
2
8
N
i
DBL
out
Out = 48
Normal mode:
the whole element is available in the loop at each iteration.

Figure A.3: LabVIEW for loop indexing. Note that this is an illustrative example, and definitely not the best way to perform this computation.

- For loops have different type of indexing modes, as represented in Figure A.3.
- In order to obtain graphs of, for instance, the BER in function of the SNR, it might be a good idea to create the *simulator.vi* icon. Creating a for loop with this block inside will enable to sweep through all SNRs automatically. In the first labs, we have provided you such a block.
- When using the USRPs, you need to chose the capture time. This capture time directly impacts the number of symbols that LabVIEW will need to store. When a memory error occurs in LabVIEW, it has the bad habit to just crash. Take therefore the time to adapt the capture time to your transmission.

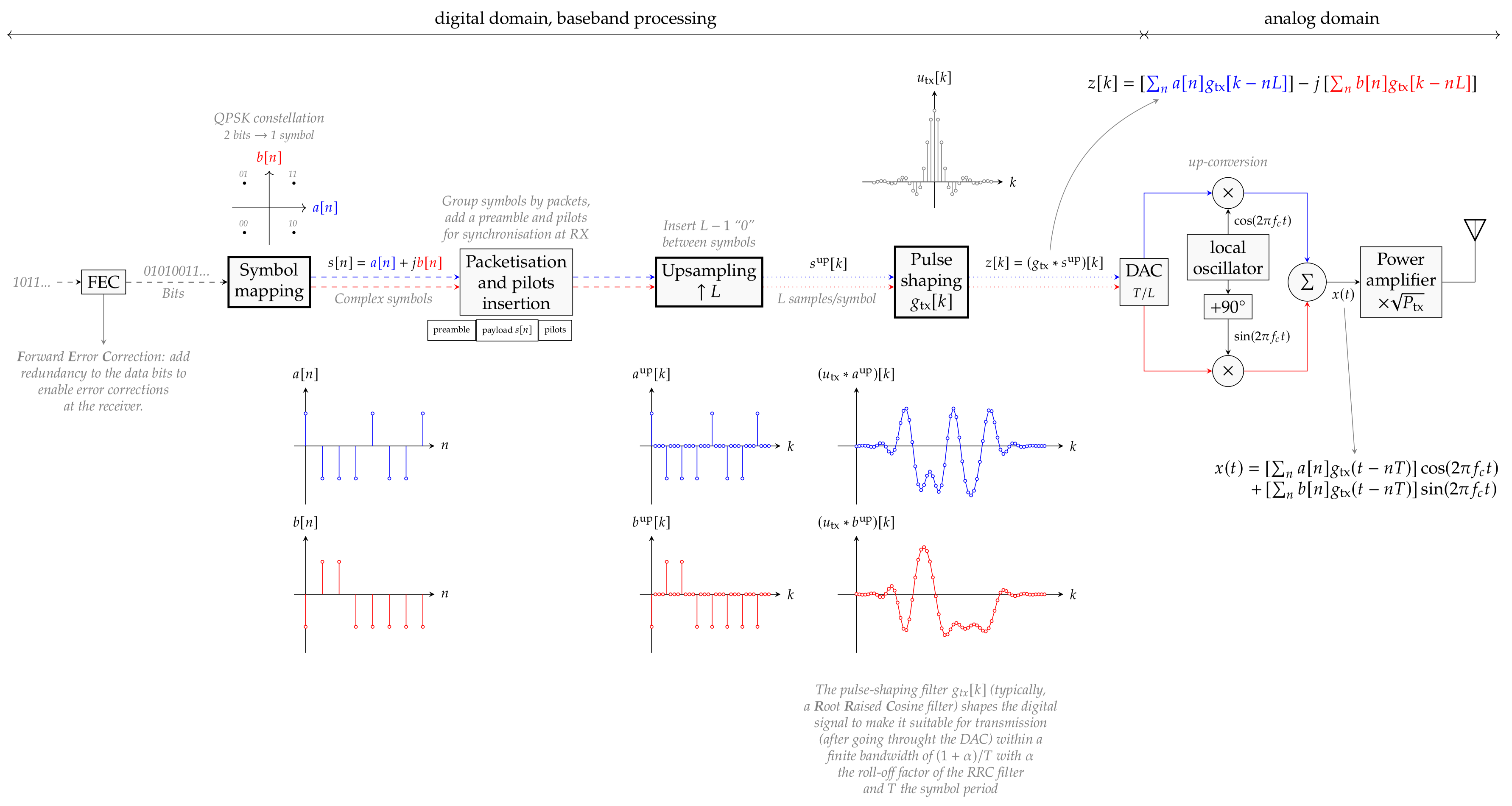

digital domain, baseband processing
analog domain
1011...
FEC
01010011...
Bits
Forward Error Correction: add redundancy to the data bits to enable error corrections at the receiver.
QPSK constellation
2 bits → 1 symbol
$b[n]$
$a[n]$
01
11
00
10
Symbol mapping
$s[n] = a[n] + jb[n]$
Complex symbols
Group symbols by packets, add a preamble and pilots for synchronisation at RX
Packetisation and pilots insertion
preamble
payload $s[n]$
pilots
Insert $L-1$ "0" between symbols
Upsampling $\uparrow L$
$s^{\text{up}}[k]$
L samples/symbol
$u_{\text{tx}}[k]$
k
Pulse shaping $g_{\text{tx}}[k]$
$z[k] = (g_{\text{tx}} * s^{\text{up}})[k]$
$z[k] = [\sum_n a[n] g_{\text{tx}}[k-nL]] - j\,[\sum_n b[n] g_{\text{tx}}[k-nL]]$
DAC
T/L
up-conversion
local oscillator
$\cos(2\pi f_c t)$
$+90°$
$\sin(2\pi f_c t)$
Σ
$x(t)$
Power amplifier $\times\sqrt{P_{\text{tx}}}$
$x(t) = [\sum_n a[n] g_{\text{tx}}(t-nT)] \cos(2\pi f_c t) + [\sum_n b[n] g_{\text{tx}}(t-nT)] \sin(2\pi f_c t)$
$a[n]$
n
$b[n]$
$a^{\text{up}}[k]$
$b^{\text{up}}[k]$
$(u_{\text{tx}} * a^{\text{up}})[k]$
$(u_{\text{tx}} * b^{\text{up}})[k]$
The pulse-shaping filter $g_{tx}[k]$ (typically, a Root Raised Cosine filter) shapes the digital signal to make it suitable for transmission (after going throught the DAC) within a finite bandwidth of $(1+\alpha)/T$ with α the roll-off factor of the RRC filter and T the symbol period

Figure A.4: Digital communications: simple transmitter (TX) chain overview.

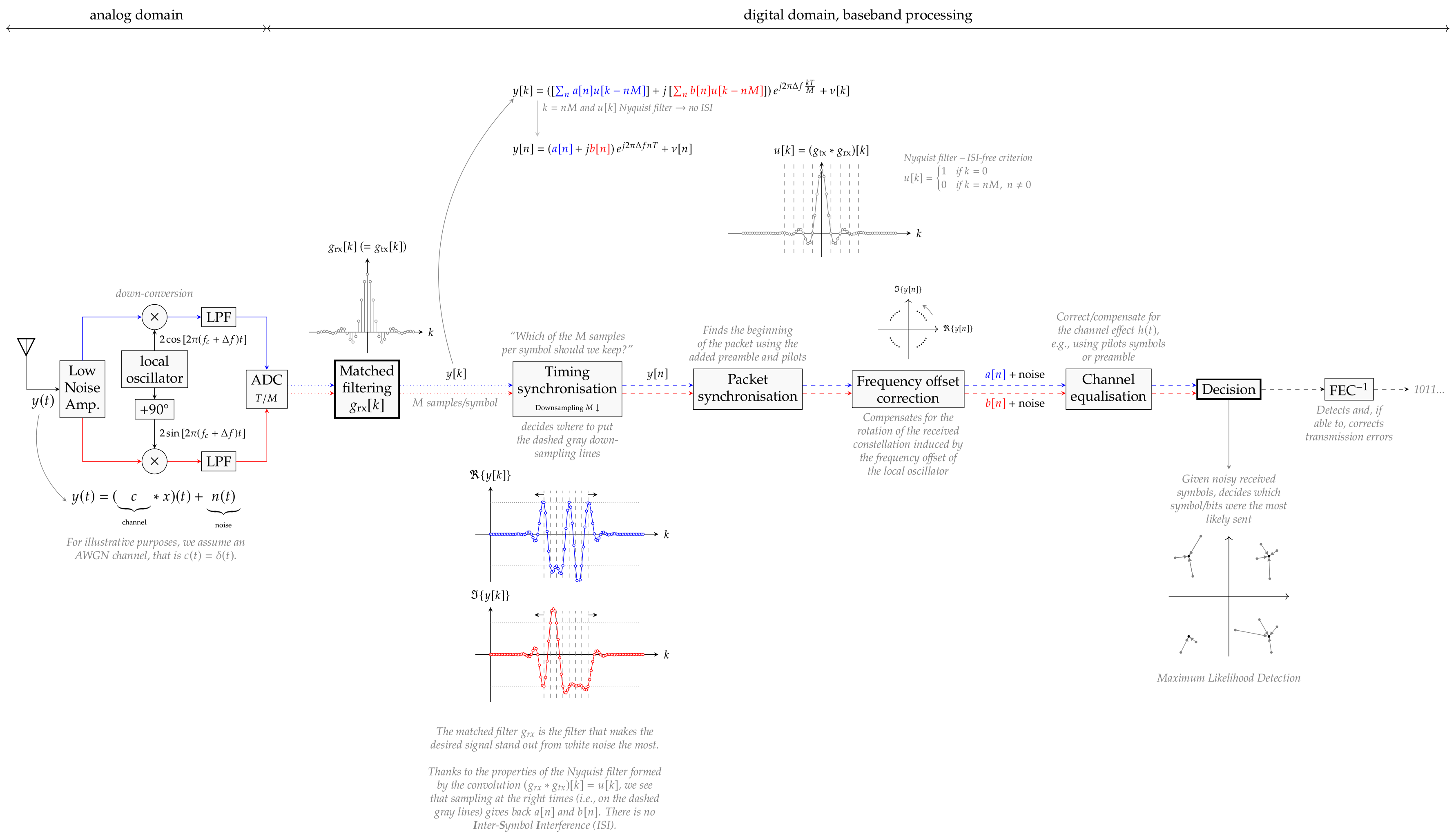

analog domain
digital domain, baseband processing
$y(t)$
Low Noise Amp.
down-conversion
local oscillator
+90°
$2\cos[2\pi(f_c + \Delta f)t]$
$2\sin[2\pi(f_c + \Delta f)t]$
LPF
LPF
ADC T/M
$y(t) = (\underbrace{c}_{\text{channel}} * x)(t) + \underbrace{n(t)}_{\text{noise}}$
For illustrative purposes, we assume an AWGN channel, that is $c(t) = \delta(t)$.
$g_{rx}[k]$ $(= g_{tx}[k])$
k
Matched filtering $g_{rx}[k]$
$y[k]$
M samples/symbol
$y[k] = ([\sum_n a[n]u[k - nM]] + j[\sum_n b[n]u[k - nM]])\, e^{j2\pi\Delta f \frac{kT}{M}} + v[k]$
$k = nM$ and $u[k]$ Nyquist filter $\longrightarrow$ no ISI
$y[n] = (a[n] + jb[n])\, e^{j2\pi\Delta f nT} + v[n]$
$u[k] = (g_{tx} * g_{rx})[k]$
Nyquist filter – ISI-free criterion
$u[k] = \begin{cases} 1 & \text{if } k = 0 \\ 0 & \text{if } k = nM,\ n \neq 0 \end{cases}$
"Which of the M samples per symbol should we keep?"
Timing synchronisation
Downsampling $M\downarrow$
decides where to put the dashed gray down-sampling lines
$y[n]$
Finds the beginning of the packet using the added preamble and pilots
Packet synchronisation
$\Im\{y[n]\}$
$\Re\{y[n]\}$
Frequency offset correction
Compensates for the rotation of the received constellation induced by the frequency offset of the local oscillator
$a[n]$ + noise
$b[n]$ + noise
Correct/compensate for the channel effect $h(t)$, e.g., using pilots symbols or preamble
Channel equalisation
Decision
FEC^{-1}
1011...
Detects and, if able to, corrects transmission errors
Given noisy received symbols, decides which symbol/bits were the most likely sent
Maximum Likelihood Detection
$\Re\{y[k]\}$
$\Im\{y[k]\}$
The matched filter g_{rx} is the filter that makes the desired signal stand out from white noise the most.
Thanks to the properties of the Nyquist filter formed by the convolution $(g_{rx} * g_{tx})[k] = u[k]$, we see that sampling at the right times (i.e., on the dashed gray lines) gives back $a[n]$ and $b[n]$. There is no Inter-Symbol Interference (ISI).

Figure A.5: Digital communications: simple receiver (RX) chain overview.

Preliminary Lab: AWGN Numeric Transmission Chain

1: Theoretical Background

The goal of this lab is to study the operations needed to go from a discrete bit sequence to a continuous waveform which will be transmitted on a channel. Such process can be split in two steps (at least this is one possibility): first, the bit sequence is mapped into a symbol sequence (symbol mapping) and secondly, continuous waveforms are attached to each symbol (pulse shaping). At the receiver side, corresponding operations are performed in the reverse order: a continuous waveform filters the received signal (matched filtering), this signal is sampled to recover the symbol sequence and then, the bits are deduced from the symbols (decision). Different kinds of waveforms are designed depending on the considered scenario (static or dynamic, urban or rural, indoor or outdoor, etc.). Such process (i.e. symbol then pulse) belongs to the family of complex pulse modulations. Other techniques exist (such as Frequency Shift Keying) but are out of the scope of these labs.

The goal of this lab is consequently to implement and understand these four steps (symbol mapping, pulse shaping, matched filtering and decision).

Symbol Mapping

The goal of the symbol mapping is to build a sequence of complex symbols with better properties than the original bit stream (higher amount of information per symbol, better use of the energy, etc).

Let us consider a sequence of bits $\mathbf{b}$. These bits are mapped to a finite set $C = \{I_0, ..., I_{N_c-1}\}$ of complex symbols to produce a sequence $\mathbf{I}$. This set is called the *constellation*. Each symbol corresponds to a specific sequence, and therefore, the number of symbols $|C| = N_s$ into the constellation is linked to the number of bits N_b allocated to each symbol as

$$N_s = 2^{N_b}.$$

Therefore, in the symbol mapping, the sequence $\mathbf{b}$ is separated in blocks of size N_b, mapped to a complex symbol $I \in C$. Every constellation is usually normalised to have a unit variance. Assuming equiprobable symbols,

$$\sigma_I^2 = \frac{1}{N_s} \sum_{I_m \in C} |I_m|^2 = 1.$$

Here are some common families of constellations which are widely used:

- *Pulse Amplitude Modulation* (PAM): with an M-PAM constellation, M symbols are equally spaced on the real axis, and centered around 0;
- *Quadrature Amplitude Modulation* (QAM): an M-QAM constellation is made of a regular square grid in the imaginary plane. It can be viewed as the combination of two $\sqrt{M}$-PAM constellation, one on the real axis and one on the imaginary axis;
- *Phase Shift Keying* (PSK): the M symbols of an M-PSK constellation are equally spaced on a circle with unit radius.

Among all these constellations, we will consider in this lab two common constellations:

- Binary PSK (BPSK), also known as 2-QAM or 2-PAM:

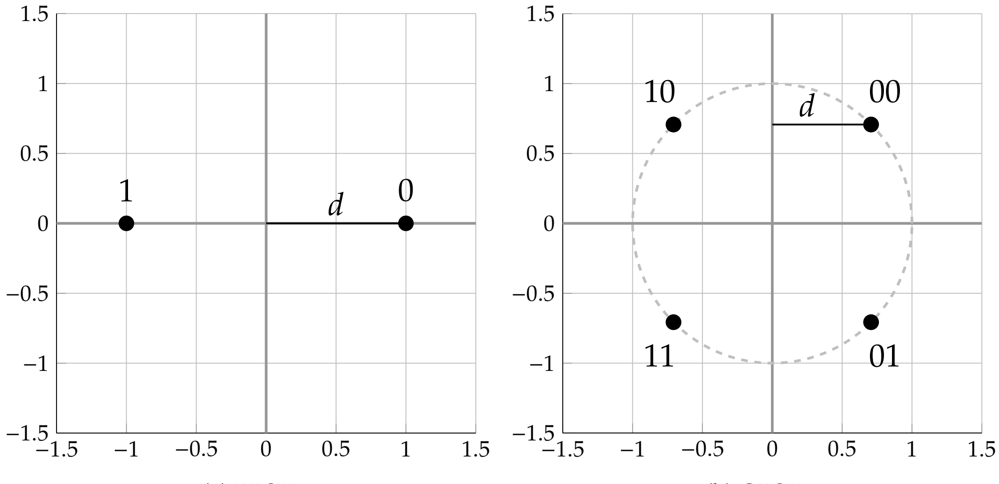
$$C = \{1, -1\};$$

- Quadrature PSK (QPSK), also known as 4-QAM:

$$C = \{1 + j, -1 + j, 1 - j, -1 - j\}.$$

  Note that $\sigma_I^2 = 2$, so every symbol must be divided by $\sqrt{2}$ to obtain a normalised constellation.

Figure B.1 illustrates these two constellations. The way bits are mapped onto the symbols follows a Gray coding: two direct neighbouring symbols differ on only one bit. This means than when selecting the wrong symbol, we can hope that at least one of the two bits is correct.



(a) BPSK (b) QPSK

Figure B.1: BPSK and QPSK constellations (normalised).

Question 1: What is the relation between the symbol variance and the minimum distance $2d$ between two symbols of a BPSK and QPSK constellation?

Question 2: Draw an 8-PSK constellation and the associated Gray mapping.

Pulse Shaping and Matched Filtering

Transmission chain: Once the symbol sequence is obtained, one needs to obtain a continuous signal to be able to transmit it to the receiver. To that aim, the symbol sequence is pulse shaped before the transmission as follows:

$$x(t) = \sum_{m=-\infty}^{\infty} I_m \, g_{\text{tx}}(t - mT),$$

where g_{tx} is the pulse shaping filter, and T is the symbol period. This symbol period is directly related to the bandwidth of g_{tx}, depending on the adopted pulse shape. Intuitively, during a pulse shaping, a waveform is attached to each symbol and the resulting signal $x(t)$ is the addition of all waveforms.

Over an Additive White Gaussian Noise (AWGN) channel, the received signal is given by

$$r(t) = x(t) + w(t),$$

where w is a complex normal noise. The Power Spectral Density (PSD) of the noise is $\gamma_w^2 = 2N_0$ in baseband, implying that $E\left[w(t)w^*(t')\right] = 2N_0\delta(t - t')$. Usually, the noise power is related to the ambient temperature and the hardware of the analog front end: $N_0 = k\mathcal{T}$, where k is the Boltzmann's constant, and $\mathcal{T}$ is the equivalent noise temperature at the input of the receiver.

Prior to the sampling operation, a filter $g_{\text{rx}}(t)$ is applied on the signal $r(t)$. Its goal is to make the useful part of the signal stands out as much as possible. The filtered signal is therefore given by:

$$y(t) = \int_{-\infty}^{\infty} r(\tau) g_{\text{rx}}(t - \tau) \mathrm{d}\tau = \sum_{m=-\infty}^{\infty} I_m \, C_g(t - mT) + v(t),$$

where the correlation between the two filters is defined as

$$C_g(t) = \int_{-\infty}^{\infty} g_{\text{tx}}(\tau) g_{\text{rx}}(t-\tau)\,\mathrm{d}\tau,$$

and the filtered noise as

$$v(t) = \int_{-\infty}^{\infty} w(\tau) g_{\text{rx}}(t-\tau)\,\mathrm{d}\tau.$$

After sampling, assuming a perfect synchronisation (i.e. we exactly know when to sample)

$$y[n] = y(nT) = \sum_{m=-\infty}^{\infty} I_m\, C_g[n-m] + v[n],$$

with $C_g[n] = C_g(nT)$ and $v[n] = v(nT)$. At this point, it would be useful to have $y[n]$ depending only on I_n as in that case one could find the corresponding symbol easily. In other words, we want to avoid Inter-Symbol Interference (ISI). This is possible when

$$C_g[n] = E_g\,\delta[n],$$

where

$$E_g = \int_{-\infty}^{\infty} g_{\text{tx}}(\tau) g_{\text{rx}}(-\tau)\,\mathrm{d}\tau,$$

leading to

$$y[n] = E_g I_n + v[n].$$

The above can be slightly transform to give the decision variable

$$\tilde{y}[n] = I_n + \tilde{v}[n]$$

with $\tilde{v}[n] = \frac{v[n]}{E_g}$, from which one can observe that the received symbol is the addition between the original one and some noise. Note that when $E_g = 1$ (i.e. when the filters are normalized), then $y[n] = \tilde{y}[n]$.

SNR maximisation and ISI cancellation: It remains to design $g_{\text{tx}}(t)$ and $g_{\text{rx}}(t)$ in order to i) avoid ISI; ii) maximise the Signal to Noise Ratio (SNR). Fortunately for us, these two criteria can be fulfilled simultaneously, i) by having two filters satisfying the Nyquist criterion and ii) by having $g_{\text{rx}}(t)$ being matched to the pulse shaping filter on the other hand.

Let's start with the SNR maximisation. It can be proven that in order to maximise the Signal-to-Noise Ratio (SNR), g_{rx} must be matched to the pulse shaping filter in the sense that

$$g_{\text{rx}}(t) = g_{\text{tx}}^*(-t).$$

Intuitively, using a matched filter enables to filter as much noise as possible without degrading the useful part of the signal.

Secondly, in order to avoid Inter-Symbol Interference (ISI), the pulse shaping and matched filter should be designed jointly such that

$$C_g[n] = \delta[n],$$

assuming the filters are scaled to a unit energy. In the frequency domain, this criterion translates into the Nyquist criterion:

$$\frac{1}{T}\sum_{k=-\infty}^{\infty} G_{\text{tx}}\left(f - \frac{k}{T}\right) G_{\text{rx}}^*\left(f - \frac{k}{T}\right) = 1,$$

which becomes with a matched filter

$$\frac{1}{T}\sum_{k=-\infty}^{\infty} \left|G_{\text{tx}}\left(f - \frac{k}{T}\right)\right|^2 = \frac{1}{T}\sum_{k=-\infty}^{\infty} \left|G_{\text{rx}}\left(f - \frac{k}{T}\right)\right|^2 = 1.$$

Some filters[4]*:* There exists lots of filters which satisfy the two above criteria, some of them being represented[5] in Figure B.3. The most simple one is probably the rectangular pulse shaping and matched filter, which can be defined as

$$g_{\text{tx}}(t) = g_{\text{rx}}(t) = \begin{cases} \dfrac{1}{\sqrt{T}} & \text{if} \quad |t| < \dfrac{T}{2}, \\ \dfrac{1}{2\sqrt{T}} & \text{if} \quad |t| = \dfrac{T}{2}, \\ 0 & \text{otherwise.} \end{cases} \triangleq \frac{1}{\sqrt{T}} \Pi\left(\frac{t}{T}\right),$$

with $\Pi(.)$ the so-called gate function. While this filter has excellent properties in the time domain (as it is extremely localised), it requires an infinite bandwidth (as it has infinitely sharp transitions between the different symbols).

Another choice is the cardinal sine pulse shaping filter with no matched filter:

$$\begin{aligned} g_{\text{tx}}(t) &= \operatorname{sinc}\left(\frac{t}{T}\right) \\ g_{\text{rx}}(t) &= \delta(t). \end{aligned}$$

The sinc pulse shaping filter has a perfect spectral content as it is a rectangle of bandwidth $1/T$, the minimum possible. Unfortunately, its drawback is that it is widely spread in the time domain, with important sidelobes.

This issue has been solved with Raised Cosine (RC) filters, which find a trade-off between the time and frequency localisation. At the price of an additional excess bandwidth α/T, where $\alpha \in [0, 1]$ is called *roll-off factor*, the filters have lower sidelobe levels. Their impulse response is given by

$$\begin{aligned} g_{\text{tx}}(t) &= \begin{cases} \dfrac{\pi}{4} \operatorname{sinc}\left(\dfrac{1}{2\alpha}\right) & \text{if} \quad t = \pm\dfrac{T}{2\alpha}, \\ \operatorname{sinc}\left(\dfrac{t}{T}\right) \dfrac{\cos\left(\frac{\pi\alpha t}{T}\right)}{1 - \left(\frac{2\alpha t}{T}\right)^2} & \text{otherwise,} \end{cases} \\ g_{\text{rx}}(t) &= \delta(t). \end{aligned} \tag{B.1}$$

One can check that when $\alpha = 0$, this filter boils down to the sinc one.

However, even if the sinc and RC filters fulfill the Nyquist criterion, they do not maximise the SNR as the receive filter is not matched to the pulse shaping one. Therefore, another family of pulse shaping filters has been designed with as goal to possess the same correlation function $C_g(t)$ as (B.1) while being matched, taking the name of Root Raised Rosine (RRC) filters. Their impulse response is given by

$$g_{\text{tx}}(t) = g_{\text{rx}}(t) = \begin{cases} \dfrac{1}{\sqrt{T}} \left[1 + \alpha\left(\dfrac{4}{\pi} - 1\right)\right] & \text{if} \quad t = 0, \\ \dfrac{\alpha}{\sqrt{2T}} \left[\left(1 + \dfrac{2}{\pi}\right) \sin\left(\dfrac{\pi}{4\alpha}\right) + \left(1 - \dfrac{2}{\pi}\right) \cos\left(\dfrac{\pi}{4\alpha}\right)\right] & \text{if} \quad t = \pm\dfrac{T}{4\alpha}, \\ \dfrac{1}{\sqrt{T}} \dfrac{\sin\left(\frac{\pi(1-\alpha)}{T} t\right) + 4\alpha \frac{t}{T} \cos\left(\frac{\pi(1+\alpha)}{T} t\right)}{\pi \frac{t}{T}\left[1 - \left(4\alpha\frac{t}{T}\right)^2\right]} & \text{otherwise.} \end{cases} \tag{B.2}$$

Table B.1 summarises the described pulse shapes, all fulfilling the Nyquist criterion.

[4]The filters have been normalised such that $E_g = 1$. Other definitions (with different scaling factors) of such filters can be valid.
[5]Not all filters of the figure are matched filters.

Name	g_{tx}	g_{rx}	B vs T	Comments
Rectangular	$\frac{1}{\sqrt{T}}\Pi\left(\frac{t}{T}\right)$	$\frac{1}{\sqrt{T}}\Pi\left(\frac{t}{T}\right)$	$B = \infty$ (main lobe width: $2/T$)	- Poor spectral containment
Cardinal sine	$\text{sinc}\left(\frac{t}{T}\right)$	$\delta(t)$	$B = \frac{1}{T}$	- Not suited for digital implementation - Difficult time synchronisation - No matched filter used at RX
Raised cosine	(B.1)	$\delta(t)$	$B = \frac{1+\alpha}{T}$	- Excess bandwidth - No matched filter used at RX
Root raised cosine	(B.2)	(B.2)	$B = \frac{1+\alpha}{T}$	- Excess bandwidth

Table B.1: Description and properties of some pulse shaping and matched filters.

Question 3: For diverse reasons, we need to design a pulse shaping and matched filter with the requirements that both filters do not have any DC component, while maximising the SNR and respecting the Nyquist criterion. To that aim, the spectrum has been divided as represented in Figure B.2, where the $G_{\text{tx}}\left(f - \frac{k}{T}\right) G^*_{\text{rx}}\left(f - \frac{k}{T}\right)$ factor of the Nyquist criterion is shown for several frequency shift. Based on this spectrum division, obtain the expression of $g_{\text{tx}}(t)$, $g_{\text{rx}}(t)$ and $C_g(t)$. *Hint: the inverse Fourier transform of a rectangle of height* 1 *centered in* f_0 *and of width* B *is given by*

$$f(t) = B\,\text{sinc}(Bt)\,e^{j2\pi f_0 t}.$$

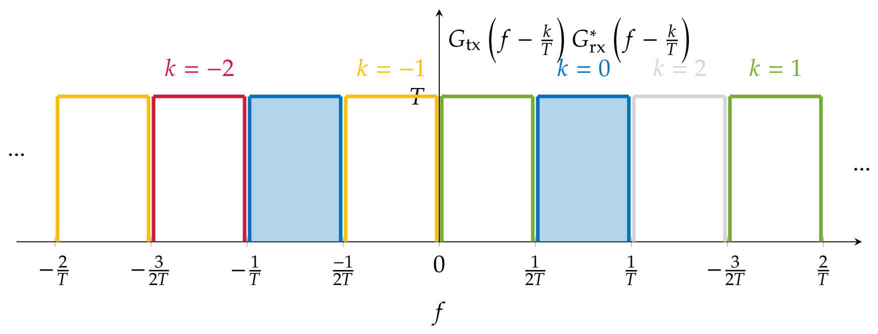


Figure B.2: Spectrum division for a pulse shaping and matched filter without DC component. The filled rectangle corresponds to $k = 0$.

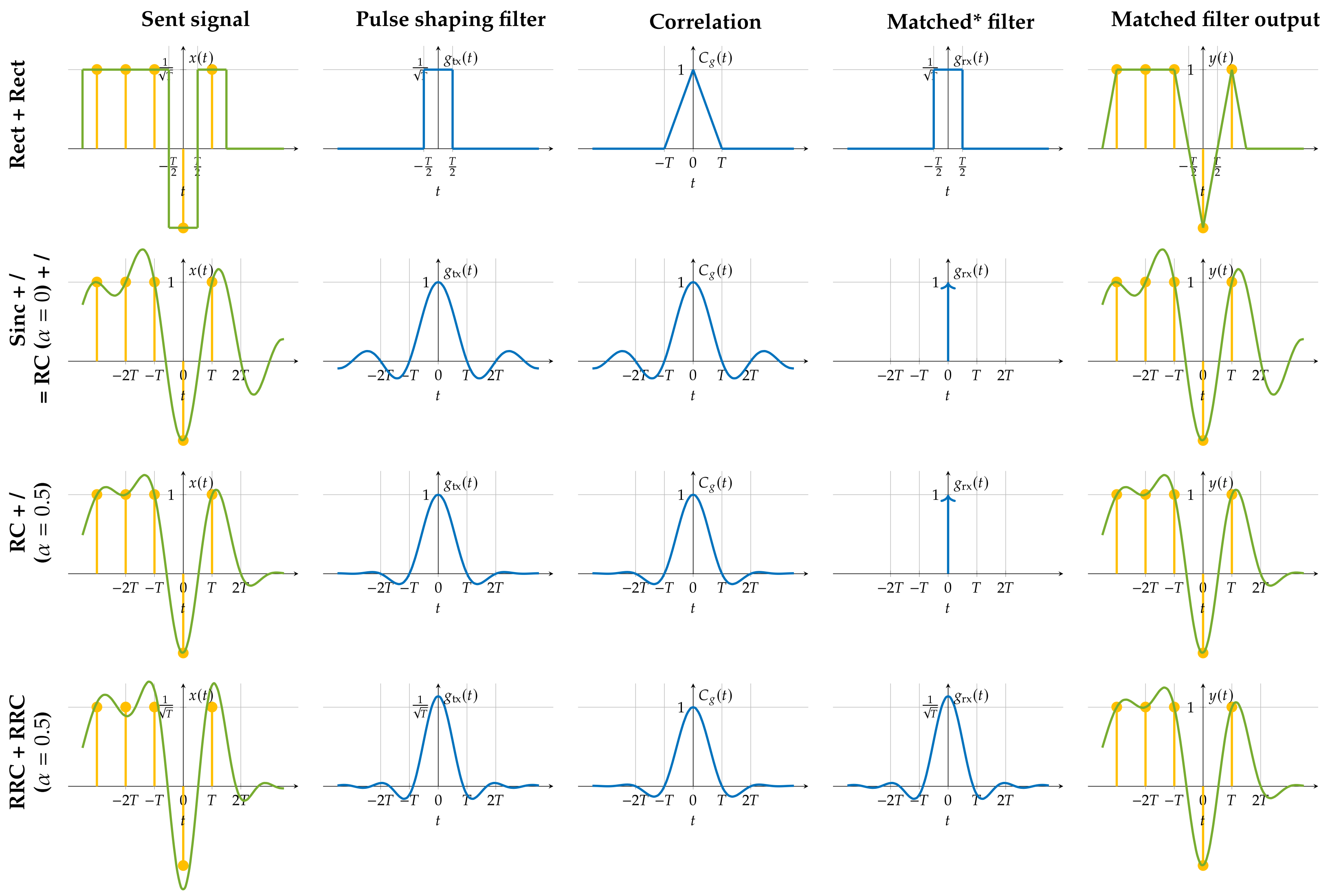


Figure B.3: This figure depicts 4 possible choices of filters, corresponding to Table B.1. All the filters respect the Nyquist criterion, as it can be observed on $y(t)$ (rightmost line). Note that $y(t)$ is noiseless, i.e. we do not consider the impact of $v(t)$. Remembering that a Raised Cosine (RC) with $\alpha = 0$ is a cardinal sine, one can compare the second and third line to conclude that increasing the roll-off factor enables to limit the side lobes level. Comparing the 3rd and 4th line, one can observe that the $C_g(t)$ and $y(t)$ are identical (this coming from the definition of a Root-Raised Cosine (RRC) filter. Finally, looking at $x(t)$ in the RRC case, one can observe that $x(nT) \neq I_n$. Indeed, the ISI is cancelled only once the matched filtering has been performed. (*) Only the first and forth lines correspond to a matched filter: the two others have no receive filter (or equivalently, $g_{rx}(t) = \delta(t)$).

Digital implementation: Note that digital implementations of the pulse shaping and matched filtering are used in practice. However, due to the excess bandwidth of rectangular and raised cosine pulse shaping filters, this is only possible when the sample rates $f_{s,\text{tx}} = {}^{1}\!/\!{}_{T_{s,\text{tx}}}$ and $f_{s,\text{rx}} = {}^{1}\!/\!{}_{T_{s,\text{rx}}}$ are increased to avoid aliasing. Therefore, we assume a sampling period $T_{s,\text{tx}} = T/L$ at the output of the transmitter and $T_{s,\text{rx}} = T/M$ at the input of the receiver, where L and M are the oversampling factors.

At the transmitter, considering the sampling of $x(t)$ at a rate $f_{s,\text{tx}}$, we have

$$\begin{aligned} x[n] &= x(nT_{s,\text{tx}}) \\ &= \sum_{m=-\infty}^{\infty} I_m \, g_{\text{tx}}(nT_{s,\text{tx}} - mT) \\ &= \sum_{m=-\infty}^{\infty} I_m \, g_{\text{tx}}[n - mL], \end{aligned}$$

where $g_{\text{tx}}[n] \triangleq g_{\text{tx}}(nT_{s,\text{tx}})$. According to the above equation, the pulse shaping can be executed digitally by upsampling the symbol stream by a factor L (insertion of $L-1$ zeros between every symbol), and convolving the upsampled symbol stream with $g_{\text{tx}}[n]$.

At the receiver, with the sampling of the received signal at a rate $f_{s,\text{rx}}$, the filtered signal becomes

$$\begin{aligned} y[n] &= y(nT) \\ &= \int_{-\infty}^{\infty} r(\tau) g_{\text{rx}}(nT - \tau) \mathrm{d}\tau \\ &\overset{(a)}{=} \sum_{m=-\infty}^{\infty} r(mT_{s,\text{rx}}) g_{\text{rx}}(nT - mT_{s,\text{rx}}) \, T_{s,\text{rx}} \\ &= \sum_{m=-\infty}^{\infty} r[m] g_{\text{rx}}[nM - m], \end{aligned}$$

where $g_{\text{rx}}[n] = T_{s,\text{rx}} \, g_{\text{rx}}(nT_{s,\text{rx}})$, and equality (a) being valid only for band-limited signals sampled above the Nyquist rate, such that there is no aliasing. The last equation shows that the MF can be executed digitally by convolving the received signal with $g_{\text{rx}}[n]$, and downsampling the output by a factor M (keep one sample every M samples). However, the downsampling operation can be executed later in the receiver chain to improve the synchronisation and equalisation performance (see next labs).

Decision

At the receiver side, once the matched filter and the sampling operation have taken place, it remains to decide which symbol have been received. To that aim, the receiver based himself on the decision variable

$$\tilde{y}[n] = I_n + \tilde{v}[n] = I_n + \frac{1}{E_g} \int_{-\infty}^{\infty} w(\tau) g_{\text{rx}}(nT - \tau) \mathrm{d}\tau,$$

which has been build from the receive signal

$$r(t) = x(t) + w(t) = \sum_{m=-\infty}^{\infty} I_m g_{\text{tx}}(t - mT) + w(t)$$

Supposing that N symbols have been transmitted, without any a priori information about the transmitted sequence, an optimal estimator is the *maximum likelihood estimator* (MLE):

$$\hat{\mathbf{I}} = \arg\max_{\mathbf{I} \in C^N} f(\mathbf{y} \mid \mathbf{I}),$$

where C^N is the set of all symbol sequences of length N, and $f(\mathbf{y}\,|\,\mathbf{I})$ is the probability to receive the sequence $\mathbf{y}$ given a transmitted sequence $\mathbf{I}$, being called the *likelihood function*. In an AWGN channel, the transmission at two different time instants are independent, and therefore the above can be simplified to a symbol by symbol decision. The decision rule is to select the closest symbol according to the euclidean distance:

$$\hat{I}_n = \arg\min_{I \in C} |y[n] - I|^2.$$

Signal to noise ratio and bit error rate

For this section, we restrict ourselves to the case of the RRC filters (as dealing with non-matched filters can be more challenging). Two metrics are particularly important when analysing a communication chain:

- the symbol energy E_s which corresponds to the energy of the useful part of **the receive signal** $x(t)$, taken at the input of the receiver (prior to any filtering operation). In our case, we have

$$E_s = \frac{\sigma_I^2 E_g}{2},$$

  the $1/2$ factor coming from the fact that as we work on a complex baseband channel, the signal are in reality modulated with a cosine prior to the transmission. This 0.5 factor corresponds thus to the energy of a cosine (mean of a cosine square).

- the Signal-to-Noise Ratio (SNR) of **the decision variable** which is the ratio between the energy of the useful part and the noise power:

$$\text{SNR} = \frac{E\left[|I_n|^2\right]}{E\left[|\tilde{v}[n]|^2\right]}.$$

Question 4: Prove that for the considered transmission chain,

$$\text{SNR} = \frac{\sigma_I^2 E_g}{2N_0} = \frac{E_s}{N_0}.$$

When the constellation, the pulse shaping filter and the matched filter are normalised, we have that

$$\text{SNR} = \frac{1}{2N_0},$$

this enabling to adapt the noise power to a target SNR. Note that to compare fairly communication systems, one should compare them for the same receive energy per bit E_b. The link between E_s and E_b depends on the constellation (or more precisely on the number of bits per symbol) as $E_s = N_b E_b$.

At this point, we would like to obtain an analytical expression of the symbol error probability. For a BPSK constellation, a symbol error occurs each time the decision variable $\tilde{y}[n]$ cross the imaginary axis:

$$p_{e,\text{BPSK}} = \frac{1}{2}\,\mathbb{P}\left(\text{Re}\{\tilde{y}[n]\} < 0 \,\middle|\, I = d\right) + \frac{1}{2}\,\mathbb{P}\left(\text{Re}\{\tilde{y}[n]\} > 0 \,\middle|\, I = -d\right) = \mathbb{P}\left(\text{Re}\{\tilde{v}[n]\} > d\right).$$

In order to compute this last probability, one needs to know the distribution of $\tilde{v}[n]$. It can be proven that it is a white noise which has a complex normal distribution of variance $\frac{2N_0}{E_g}$, or said otherwise whose real and imaginary parts are both normally distributed with variance $\frac{N_0}{E_g}$. Finally, it remains to link the symbol error rate to the Bit Error Rate (BER) which also depends on the constellation size. If a Gray mapping is employed, one can assume that each symbol error leads only to one bit error, and that therefore BER $= p_e/N_b$.

Question 5: Derive the analytical expressions of the symbol error probability and BER for a BPSK modulation, in terms of SNR, E_s/N_0 and E_b/N_0.
Hint: the erfc *function is defined as*

$$\text{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-t^2}\, \mathrm{d}t.$$

With a QPSK constellation, the symbol error probability is approximately equal to the sum of the symbol error probability of the in-phase and in-quadrature branches. However, the variance of the constellation is different compared to a BPSK constellation for the same minimum distance $2d$ between two symbols. Moreover, considering that most of the committed errors happen between two direct neighbours, thanks to the Gray coding, the BER is approximately equal to half the symbol error probability.

Question 6: Derive the analytical expressions of the symbol error probability and BER for a QPSK constellations, in terms of SNR, E_s/N_0 and E_b/N_0.

Eye Diagram

The eye diagram is a tool enabling to visualise the received signal, in order to evaluate the combined effects of the noise and ISI, but also to extract some informations related to the synchronisation (for example, the best time instant to sample the signal). This diagram is obtained from the in-phase or in-quadrature component of the signal by cutting it into slices of some symbol periods (typically two), and by overlaying all the slices. Figure B.4 shows a typical eye diagram obtained with a BPSK constellation.

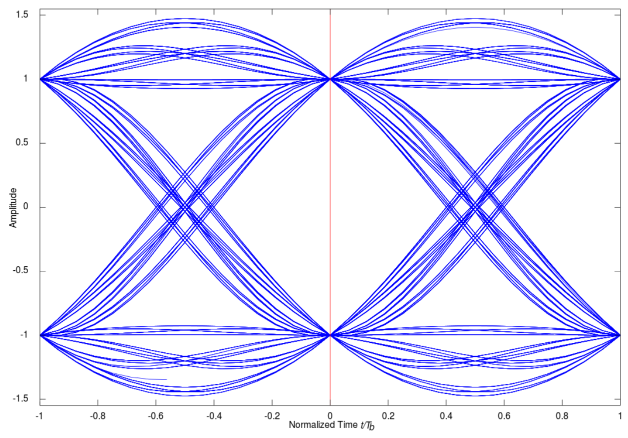


Figure B.4: Eye diagram [Wikipedia - Eye diagram].

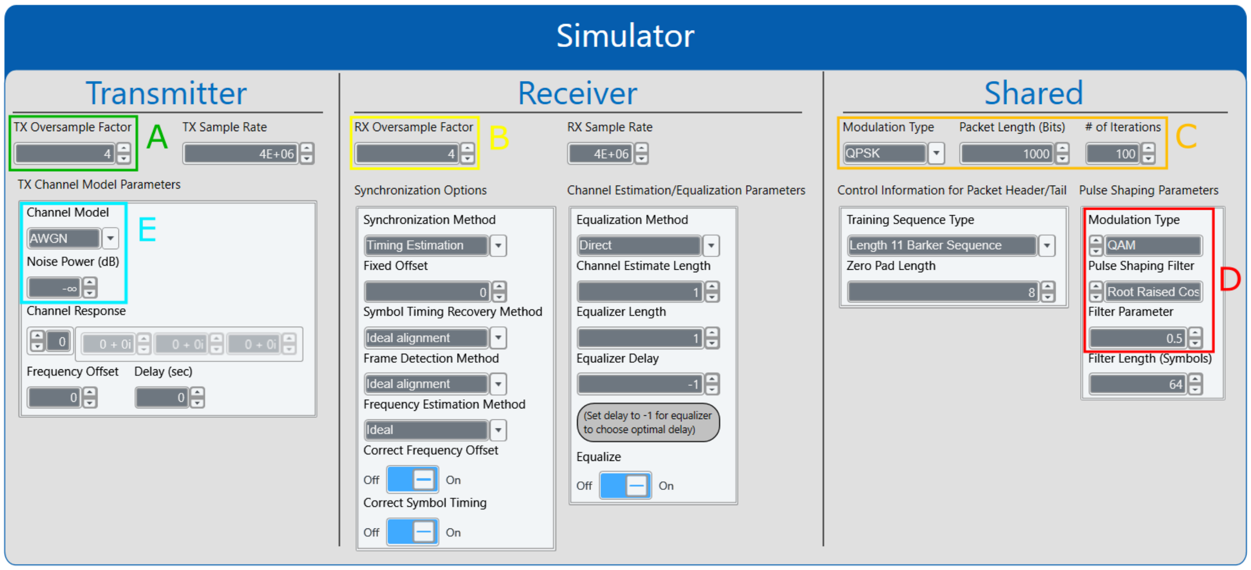
2: Lab instructions

For this laboratory, a complete communication chain is provided. However, you will have to implement your own blocks (VIs) for the symbol mapping and the decision, as well as for the pulse shaping and matched filters.

Complete communication chain

The provided communication chain comes from [Digital Communications - Physical Layer Exploration Lab Using The NI USRP Platform]. The chain is currently functional and enables to simulate a complete digital transmission chain but also to use USRPs as transmitters and receivers. This explains the complexity of the provided chain.

Nevertheless, the main VI is named *simulator.gvi*. In the *Panel* tab, you find several parameters associated to the digital transmission, such as the oversampling factor, the modulation type or the pulse shaping filter. Start by checking that the simulation runs properly, by clicking on *Run*. You should obtain a perfect QPSK constellation with an average Bit-Error Rate (BER) of 0. This signifies that the transmission happened without any error.



Simulator

Transmitter

TX Oversample Factor: 4
TX Sample Rate: 4E+06
TX Channel Model Parameters
Channel Model: AWGN
Noise Power (dB): -∞
Channel Response: 0 | 0 + 0i | 0 + 0i | 0 + 0i
Frequency Offset: 0
Delay (sec): 0

Receiver

RX Oversample Factor: 4
RX Sample Rate: 4E+06
Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Ideal alignment
Frame Detection Method: Ideal alignment
Frequency Estimation Method: Ideal
Correct Frequency Offset: Off / On
Correct Symbol Timing: Off / On
Channel Estimation/Equalization Parameters
Equalization Method: Direct
Channel Estimate Length: 1
Equalizer Length: 1
Equalizer Delay: -1
(Set delay to -1 for equalizer to choose optimal delay)
Equalize: Off / On

Shared

Modulation Type: QPSK
Packet Length (Bits): 1000
# of Iterations: 100
Control Information for Packet Header/Tail
Training Sequence Type: Length 11 Barker Sequence
Zero Pad Length: 8
Pulse Shaping Parameters
Modulation Type: QAM
Pulse Shaping Filter: Root Raised Cos
Filter Parameter: 0.5
Filter Length (Symbols): 64

Figure B.5: Useful simulation parameters.

Figure B.5 gathers different useful parameters, namely:

A. The oversampling factor of the emitter L (named *TX Oversample Factor*) ;

B. The oversampling factor of the receiver M (named *RX Oversample Factor*) ;

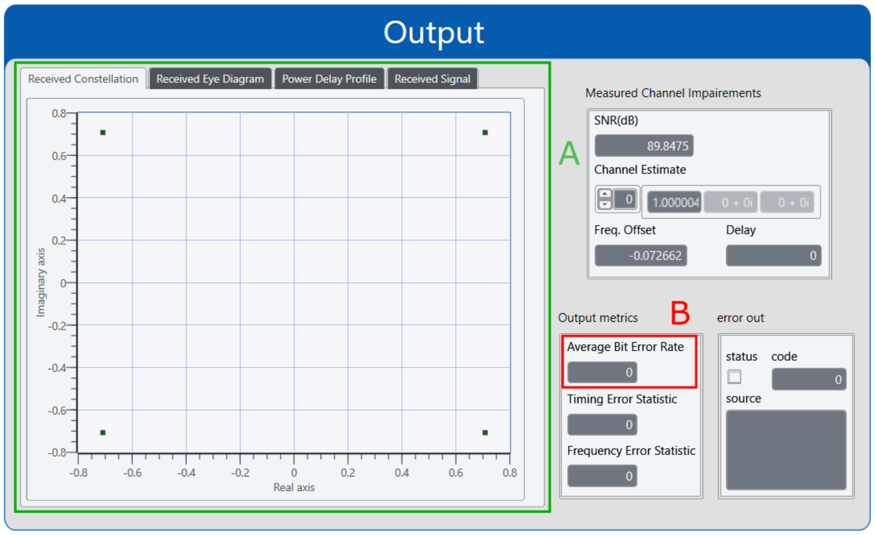
C. The modulation type (BPSK or QPSK) as well as the number of bits sent per packet and the number of times the simulation is repeated. This enables to get meaningful averages ;

D. The pulse shaping filter type (named *Pulse Shaping Filter*. Here, a *Root Raised Cosine* is selected), and the roll-off factor α (named *Filter Parameter*). The filters correspond to those of Table B.1 (and the *Gaussian* one should not be used);

E. The canal type and the noise power. Use the AWGN model and set the noise power according to the desired SNR, knowing that the constellation, the pulse shaping filter and the matched filters are normalised. For example, to study a SNR of 20dB, the parameter *Noise Power (dB)* should be set to -20dB.

Other parameters are associated to the synchronization and the equalization and will be useful in the following laboratories.

In Figure B.6, you can observe the received constellation or the received eye diagram (in A), as well as the average BER, averaged on all iterations (in B). Note that the value *SNR(dB)* in the cluster *Measured Channel Impairments* does not exactly correspond to the chosen SNR value with the parameter *Noise Power (dB)*, as this is an estimated value. You should therefore use the value that you set with *Noise Power (dB)*.



Measured Channel Impairements

SNR(dB) 89.8475

Channel Estimate 0 1.000004 0 + 0i 0 + 0i

Freq. Offset -0.072662

Delay 0

Output metrics

Average Bit Error Rate 0

Timing Error Statistic 0

Frequency Error Statistic 0

error out

status code 0

source

Figure B.6: Example of simulation output.

Furthermore, going in the *Diagram* tab associated to the file *simulator.gvi*, you will notice that the simulator is calling the VIs *top_tx.gvi* and *top_rx.gvi*, giving them the necessary simulation parameters. Note that if you do not want to run a simulation but instead perform a digital transmission using the USRPs, you have to directly use the blocks *top_tx.gvi* and *top_rx.gvi* giving them the IP addresses of the USRPs to use (one as emitter, one as receiver). However, for this laboratory, you do not have to do so as we will only use simulation results.

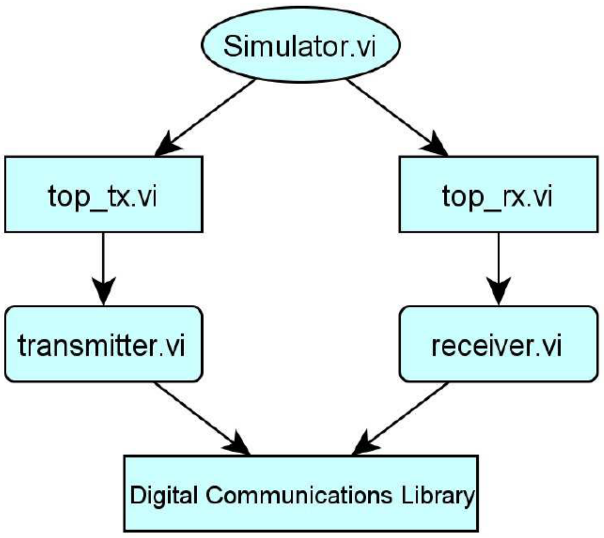


Figure B.7: Hierarchy of simulator.

You are now encouraged to go deeper in the different VIs used in *top_tx.gvi* and *top_rx.gvi*, namely the VIs *transmitter.gvi* and *receiver.gvi*. Th hierachy of the provided VIs is given in Figure B.7.

Question 7: Compare the transmission chains of the VIs *transmitter.gvi* and *receiver.gvi* with the complete digital transmission chain presented in A.4 and A.5.

Several blocks in the transmitter and the receiver are used only for the synchronization. For this preliminary laboratory, you should not go deeper in the understanding of these blocks as it will be the topic of the following laboratories. The blocks that will be studied during this lab are: :

- *modulate.gvi* (named *MOD*) used in *transmitter.gvi* ;
- *pulse_shaping.gvi* used in *transmitter.gvi* ;
- *matched_filter.gvi* used in *receiver.gvi* ;
- *decode.gvi* used in *receiver.gvi*.

Notice that these blocks are protected: you are unable to open and look at their diagrams. Indeed, the first goal of this lab is to implement these blocks (see the VIs named *student_*.gvi* in the tab *Files*) and to substitute them instead of the ones that are currently provided. Of course, **the simulation must also run properly with your blocks instead of the LabVIEW blocks that are already provided.**

3: Symbol mapping and decision

In a first step, implement the symbol mapping and the decision, basing yourself on the above theory.

student_modulate.gvi Complete the VI *student_modulate.gvi* and substitute it to the VI *modulate.gvi* (named *MOD*) in *transmitter.gvi*. Its inputs and outputs are given in Table Table B.2.

student_modulate.gvi - Transforms a bit sequence in symbols, using the selected modulation.			
Inputs	*Input Bit Stream*	U8 1D array	Bit sequence to transform into symbols, taking 0 or 1 values.
	Modulation Parameters In	Cluster	Modulation parameters in input. The *Modulation Type* parameter gives the chosen constellation: either BPSK or QPSK.
	Error In	Error cluster	Contains information regarding a possible error in the previous blocks. This can be direclty connected to *Error Out*.
Outputs	*Output Symbols*	CDB 1D array	Symbols associated to the input bits, using the appropriate modulation.
	Modulation Parameters Out	Cluster	Modulation parameters in output.
	Error Out	Error cluster	Contains information regarding a possible error in the previous blocks or in this block. This can be direclty connected to *Error In*.

Table B.2: Inputs and outputs of the block *student_modulate.gvi*.

Hints:

a. Use a *Case structure* based on the *Modulation Type* parameter in order to choose the operations to apply on the input bits based on the selected modulation;

b. The block *Decimate 1D Array* can be useful to implement the QPSK constellation;

c. Don't forget to use normalize constellation;

d. Use probes to debug your implementation;

e. It is possible to easily implement the BPSK constellation based on a linear transformation (for example): what should be the values of the parameters m and p such that, for all bit $b \in \{0, 1\}$, the associated symbol value $s \in \{-1, 1\}$ is given by $s = m\, b + p$? This transformation can also be used with the QPSK constellation.

student_decode.gvi Next, implement the symbol decoding (i.e. the decision rule) in the VI *student_decode.gvi* and substitute it to the VI *decode.gvi* in *receiver.gvi*. Its inputs and outputs are given in Table B.3.

student_decode.gvi - Performs the detection using a maximum likelihood estimator for a given symbol sequence			
Inputs	*Input Symbols*	CDB 1D array	Symbols to decode.
	Modulation Parameters In	Cluster	Modulation parameters in input. The *Modulation Type* parameter gives the chosen constellation.
	Error In	Error cluster	Contains information regarding a possible error in the previous blocks. This can be directly connected to *Error Out*.
Outputs	*Estimated Bit Sequence*	U8 1D array	Sequence of decoded bits.
	Modulation Parameters Out	Cluster	Modulation parameters in output.
	Error Out	Error cluster	Contains information regarding a possible error in the previous blocks or in this block. This can be directly connected to *Error In*.

Table B.3: Inputs and outputs of the block *student_decode.gvi*.

Hints:

a. Use again a *Case structure* based on the *Modulation Type* parameter to choose the applied operations based on the selected modulation;

b. The block *Interleave 1D Arrays* enables to merge two 1D arrays outputting successively one element from each 1D array. This block can be useful for the QPSK constellation;

c. The decision rule can be implemented using comparison blocks and/or a *Select* block.

4: Pulse shaping and matched filters

In a second step, implement the pulse shaping filter and the matched filter, as explained in the theory.

student_pulse_shaping.gvi Implement the pulse shaping filter and the oversampling in the VI *student_pulse_shaping.gvi* and substitute it to the VI *pulse_shaping.gvi* in *transmitter.gvi*. This block takes the symbol stream, oversample it and then convolve the oversampled stream with the filter. Its inputs and outputs are given in Table B.4.

student_pulse_shaping.gvi - Oversample the input and apply the pulse shaping filter.			
Inputs	*Input*	CDB 1D array	Input symbols.
	Modulation Parameters In	Cluster	Modulation parameters in input. The useful parameters are the *TX Oversample Factor* and the *Pulse Shaping Parameters: Modulation Type, Pulse Shaping Filter*, *Filter Parameter* and *Filter Length*.
	Error In	Error cluster	Contains information regarding a possible error in the previous blocks. This can be direclty connected to *Error Out*.
Outputs	*Output*	CDB 1D array	Oversampled input convolved with the pulse shaping filter.
	Filter Coefficients	DBL 1D array	Pulse shaping filter coefficients.
	Modulation Parameters Out	Cluster	Modulation parameters in output.
	Error Out	Error cluster	Contains information regarding a possible error in the previous blocks or in this block. This can be direclty connected to *Error In*.

Table B.4: Inputs and outputs of the block *student_pulse_shaping.gvi*.

Hints:

a. Use the block *Generate Filters Coefficients* (with the same documation than the LabVIEW VI *MT Generate Filter Coefficients*) giving it the appropriate parameters (gathered in the cluster *Modulation Parameters In*). Its outputs gives the *Pulse Shaping Filter Coefficients* that should be convolved with the input symbols that have been previously oversampled. Thus, you do not need to implement (B.1) and (B.2);

b. For the oversampling, use the VI *Upsample.gvi*;

c. For the convolution, use the VI named *Convolution.gvi*.

student_matched_filtering.gvi Finally, implement the matched filter (**without downsampling!**) in the VI *student_matched_filtering.gvi* and substitute it to the VI *matched_filtering.gvi* in *receiver.gvi*. Its inputs and outputs are given in Table B.5.

Hints:

a. You will have to use again the *Generate Filters Coefficients* block giving it the adequate parameters (that are gathered in the *Modulation Parameters In* cluster). This time, use the *Matched Filter Coefficients* output and convolve it with the received symbols;

b. For the convolution, use the VI named *Convolution.gvi*;

c. There is no downsampling to apply. Indeed, this operation is performed later in the reception chain, to benefit from more samples for synchronization operations.

5: Questions

At the end of this laboratory, you should be able to answer the following questions:

student_matched_filtering.gvi - Apply the matched filters corresponding to the given parameters.			
Inputs	*Input Complex Waveform*	Waveform	Received signal in basedband.
	Modulation Parameters In	Cluster	Modulation parameters in input. The useful parameters are *RX Oversample Factor* and *Pulse Shaping Parameters: Modulation Type, Pulse Shaping Filter*, *Filter Parameter* and *Filter Length*.
	Error In	Error cluster	Contains information regarding a possible error in the previous blocks. This can be direclty connected to *Error Out*.
Outputs	*Output Complex Waveform*	Waveform	Filtered signal by the matched filter.
	Modulation Parameters Out	Cluster	Modulation parameters in output.
	Error Out	Error cluster	Contains information regarding a possible error in the previous blocks or in this block. This can be direclty connected to *Error In*.

Table B.5: Inputs and outputs of the block *student_matched_filtering.gvi*.

Symbol mapping and decision

Question 8: What is the goal of the symbol mapping? What is its impact on the bit rate?

Question 9: Add noise by modifying the *Noise Power (dB)*. What do you observe on the received BPSK and QPSK constellations? If both constellations are normalized, which one is the most sensitive to noise?

It is also interesting to obtain BER vs SNR curves. To do so, one can run the simulation for different noise powers and then collect the obtained BERs in a graph. Yet, to facilitate this step, you have at your disposition a VI called *simulator_SNR_loop* which encapsulates the simulator and enables to directly obtain such a curve. It has the same interface than the usual simulator at the exception that the noise cannot be chosen. Instead, the SNR vector must be specified in the *BER vs SNR* part of the panel. In order to be able to compare the curves to the theoretical ones, you should then export the data to another program (such as Python or Matlab). There are (at least) two ways to do so:

- In the *"Panel"* tab, right-click on the graph and select *"Capture Data"*. The data should appear in the left part of LabVIEW. Right click on the data which has just been created and select *"Export"*. This will create a CSV file from which you will be able to copy past the data points.
- In the *"Panel"* tab, right-click on the BER table and select *"Capture Data"*. The data should appear in the left part of LabVIEW. Double click on the data: a new tab should appear from which you can directly copy-paste the data. Note that you may need to reduce the zoom in order to copy-paste all the data at once. You can also export the CSV file from this data (it is usually easier than exporting a graph).

Question 10: Obtain the average BER curves for BPSK and QPSK. To do so, use SNRs going from 0 to 10 dB and for these SNRs, measure the average BER given by LabVIEW. For your results to be meaningful, you should increase the number of iterations to transmit more bits and average on more realisations (point C on Figure B.5). Make sure to use a root-raised cosine pulse shaping filter with a roll-off factor $\alpha = 0.5$. Use a logarithmic scale for the average BER and compare your simulation curves with the theoretical curve (from the theoretical expression computed above). Where is the difference between the BPSK and QPSK curves coming from and how much is it (in dB)?

Question 11: Following your observations of the BER curves and of the received BPSK and QPSK constellations, explain the pros and cons of each modulation in the case where the main goal is the bit rate or the BER.

Pulse shaping and matched filter

Question 12: Why using a pulse shaping filter and a matched filter? Can only one of them be used?

Question 13: Observed the eye diagram with a *Root Raised Cosine* (*Received Eye Diagram* tab in part A of Figure B.6), in BPSK.

1. At which instant must the signal be sampled in the best case? What are the recovered values, in BPSK and QPSK?
2. Describe the effects of the roll-off factor α (*Filter Parameter* in D on Figure B.5) on the eye diagram, varying it between 0 and 1. What is the impact on ISI when there are sampling errors? Increase the oversampling factors to 8 to take more samples into account and smooth out the obtained curve.
3. Describe the effect of the noise on the eye diagram.

Question 14: Use oversampling factors of 8, with a noise power of -20dB. Observe, compare and explain the received eye diagram with a raised cosine or a root-raised cosine as pulse shaping filters (for a same α).

Question 15: According to your knowledge and by varying the roll-off factor α of the root-raised cosine filter, what is the relationship between this parameter and the excess bandwidth, as well as with the ISI? Are the ISI increasing or decreasing with α and why? In this case, are the two following goals complementary or antagonist:

- Decrease the excess bandwidth;
- Decrease the ISI.

To answer this question more easily, you can observe the pulse shaping filter used in LabVIEW. For that, modify the VI *student_pulse_shaping.gvi* to add a graph with the filter coefficients. You can then observe the waveform in time. You can also observe its spectrum by taking the discrete Fourier transform of these coefficients and display its modulus. Vary α, what do you observe? Increase the oversampling factors to take more samples and smooth out the obtained curve.

Lab 1: Synchronisation - Symbol Timing Recovery in Narrowband Channels

1: Introduction

When analysing a communication link, the impact of the noise on the communication is of crucial interest. Such noise is usually modelled as an additive white Gaussian noise: when transmitting a signal $x(t)$, the received signal reads as $r(t) = x(t) + w(t)$ with $w(t)$ an AWGN. While this AWGN model provides excellent insights (Matched filter, Nyquist criterion, BER curves), it fails to model some impairments of the channel. In this lab (and in the lectures), we will therefore study a more complex model named the flat-fading (a.k.a. frequency-flat) channel model, which reads in a baseband representation as

$$r(t) = \beta x(t - \tilde{\tau}_d) + w(t),$$

with $\beta = ae^{j\phi}$ a complex scalar coefficient modelling the attenuation a and the phase shift ϕ induced by the channel, and $\tilde{\tau}_d$ the propagation delay. This propagation delay can be split as $\tilde{\tau}_d = \tau_d + dT$, with τ_d smaller than the symbol period T and d an integer. The correction of these impairments can be divided in three steps:

- Equalisation: estimation and correction of β. (Goal of the next lab);
- Symbol alignment: estimation and correction of τ_d;
- Frame synchronisation: estimation and correction of d (which will not be seen in this course. For those who are interested, a supplementary lab describes this step.)

This lab focuses on the symbol alignment operation. Its goal is on the one hand to understand and implement one synchronisation algorithm, and on the other hand to figure out the impacts of the pulse shaping filter, of the roll-off factor alpha and of the oversampling factor on the accuracy of the synchronisation.

2: Theoretical background

Expression of the signals and impact of the delay In a baseband representation, if $\tau_d < T$, the transmitted signal $x(t)$, the received one $r(t)$, the output of the matched filter $y(t)$ and the output of the sampler $y[n]$ can respectively be described as

$$\begin{aligned}
x(t) &= \sum_{m=-\infty}^{\infty} I_m g(t - mT), \\
r(t) &= \beta \sum_{m=-\infty}^{\infty} I_m g(t - \tau_d - mT) + w(t), \\
y(t) &= \beta \sum_{m=-\infty}^{\infty} I_m C_g(t - \tau_d - mT) + v(t), \\
y[n] &= \beta I_n C_g(\tau_d) + \underbrace{\beta \sum_{m \neq n} I_m C_g((n-m)T - \tau_d)}_{\text{ISI}} + \underbrace{v[n]}_{\text{noise}},
\end{aligned}$$

with the pulse shaping and matched filter correlation[6] defined as

$$C_g(\tau) = \int_{-\infty}^{\infty} g(t) g^*(t - \tau) \mathrm{d}t,$$

[6] Throughout this lab, we assume that the pulse shaping and receive filters are matched, real and even, meaning that $g_{\text{tx}}(t) = g_{\text{rx}}(t) \triangleq g(t)$. We also consider they fulfil the Nyquist criterion.

and with $v[n]$ an AWGN with a variance equal to

$$\sigma_v^2 = 2N_0 E_g,$$

where $E_g \triangleq C_g(0)$.

Question 1: Starting from the above signals and from the flat fading channel model, verify the expression of y and of the noise power.

From the above expressions, one can observe that the delay τ_d induces ISI as the correlation function $C_g(\tau)$ is no longer sampled at multiples of T (which are its zeros). Furthermore, the correct symbol I_n is affected by a factor $C_g(\tau_d)$ which reduces its energy.

Maximum Energy function In order to estimate τ_d, several approaches can be considered which mainly differ by the fact that some of them use training sequences (data aided type) while others not (non data aided type). In this lab, we focus on the non data aided mode. In this case, the maximum likelihood estimator[7] of the delay takes the form of

$$\hat{\tau}_d = \underset{0 \le \tau < T}{\arg\max}\, J(\tau),$$

with $J(\tau)$ being the energy function defined as

$$J(\tau) = \mathbb{E}\left[|y(nT+\tau)|^2\right].$$

This symbol alignment method is therefore called the *maximum energy method*.

Analytical expression of the energy function In order to understand why the maximum energy method works, and to figure out which parameters impact the solution, we seek to obtain a closed-form expression of $J(\tau)$. From its definition, it can be expanded as

$$J(\tau) = |\beta|^2 \sigma_I^2 \sum_{m=-\infty}^{\infty} \left|C_g(mT+\tau-\tau_d)\right|^2 + 2N_0 E_g.$$

Question 2: Derive the above expression from the definition of $y(t)$ and $J(\tau)$.

Question 3: What will be the impact of the flat fading channel coefficient β on the maximum energy solution $\hat{\tau}_d$?

In the above equation, we can observe that the following function plays a central role:

$$\tilde{J}(\theta) = \sum_{m=-\infty}^{\infty} \left|C_g(mT+\theta)\right|^2.$$

After some lengthy but interesting developments which are presented in Section 4, we can obtain that for a unit-energy RRC filter with roll-off factor α,

$$\tilde{J}(\theta) = 1 - \frac{\alpha}{2} + \frac{\alpha}{2}\cos^2\left(\frac{\pi\theta}{T}\right).$$

The above expression, represented in Figure 1.1 for different values of the roll-off α, gives the value of the energy function in function of the roll-off factor and the timing error θ. Except when $\alpha = 0$, one can observe that the energy function is indeed maximized when the synchronization is perfect. We can moreover clearly see the impact of the roll-off factor on such a maximum energy function.

[7]For those interested in knowing how to obtain such an estimator, wait for the *LELEC2880 - Communication and Estimation Theory* course.

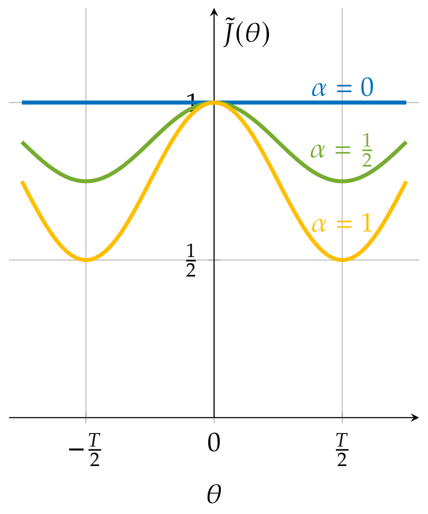


Figure 1.1: Energy function

Rectangular pulse shaping In order to grasp the impact of the pulse-shaping filter on the symbol alignment, we can also study the case of rectangular pulse shaping (which also respects the Nyquist criterion by the way). The pulse shaping filter and the associated correlation function are represented in Figure 1.2.

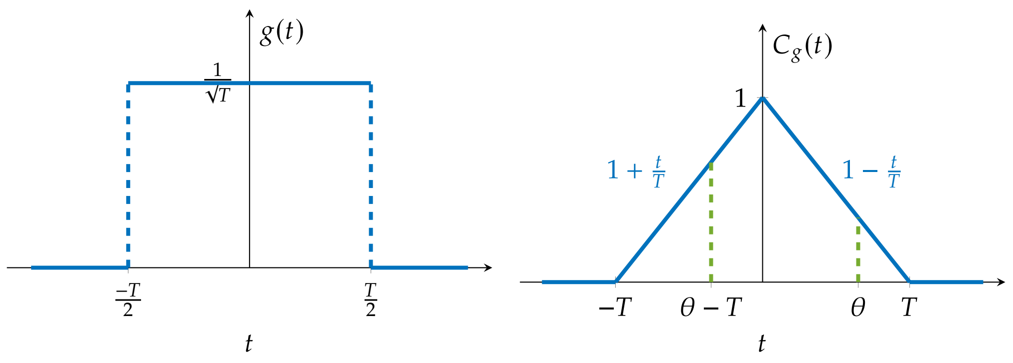


Figure 1.2: Pulse shaping rectangular function and the associated correlation function

Question 4: From Figure 1.2, prove that $\tilde{J}(\tau)$ in case of a rectangular pulse shaping is given by

$$\tilde{J}(\theta) = 2\left(\frac{\theta}{T}\right)^2 - 2\left|\frac{\theta}{T}\right| + 1.$$

Draw it on Figure 1.1 and compare it to the case of RRC filters.
Hint: the development is far more easier than for RRCs and should only take a few lines. You can assume $\theta \geq 0$. Moreover, it can be seen on Figure 1.2 that only two terms of the summation of $\tilde{J}(\theta)$ are required, the others being zero.

Question 5: Considering a noisy signal $y(t)$, which pulse shaping will lead to the more robust estimation of τ_d?

From continuous to discrete All the above developments make use of the expectation operator, and consider the continuous signal $y(t)$. However, as the LabVIEW communication chain is digital, we do not have access to such a signal. We have instead at our disposition the oversampled signal $\tilde{y}[n] = y\left(\frac{T}{M}n\right)$ where M is the oversampling factor. Therefore, we will only be able to estimate delays which are multiples of the oversample period, i.e. such that $\tau = k\frac{T}{M}$. Assuming the signals are ergodic (and that the expectation can be taken along time instead of along realisations), we approximate the energy function as

$$J_{\text{approx}}[k] = \frac{1}{P}\sum_{p=0}^{P-1}\left|y\left(pT + k\frac{T}{M}\right)\right|^2 = \frac{1}{P}\sum_{p=0}^{P-1}|\tilde{y}[pM+k]|^2\,,$$

where P depends on the length of $\tilde{y}[n]$. Choosing a large P usually gives better performances as the noise is averaged on more samples. With this definition, the maximum energy solution reads as

$$\hat{k} = \underset{k=0,\ldots,M-1}{\arg\max}\, J_{\text{approx}}[k].$$

Once $\hat{k}$ is obtained, the oversampled signal must be "re-aligned" before being downsampled.

3: Lab Instructions

1. Implement the maximum energy alignment method based on the provided template, and based on the specifications described in Table 1.1. **Insert your block inside *symbol_timing.gvi***, which can be found through *simulator.gvi* →*top_rx.gvi*→ *receiver.gvi* → *synchronize.gvi* → *symbol_timing.gvi*. (Note that several synchronisation methods exist, but that we focus in this lab on the *Timing Estimation* method).

student_align_max_energy.gvi - compute and correct timing offset by maximising $J_{\text{approx}}[k]$.			
Inputs	*Input complex waveform*	Waveform	Received sequence after matched filtering (i.e. $\tilde{y}$).
	Modulation Parameters In	Cluster	Modulation parameters in input.
	Error In	Error cluster	Contains information regarding a possible error in the previous blocks. This can be directly connected to *Error Out*.
Outputs	*Aligned complex waveform*	Waveform	Received sequence after symbol timing recovery.
	Modulation Parameters Out	Cluster	Modulation parameters in output.
	Alignment Offset	I32 (Integer)	Computed discrete offset $\hat{k}$.
	Error In	Error cluster	Contains information regarding a possible error in the previous blocks or in this block. This can be directly connected to *Error In*.

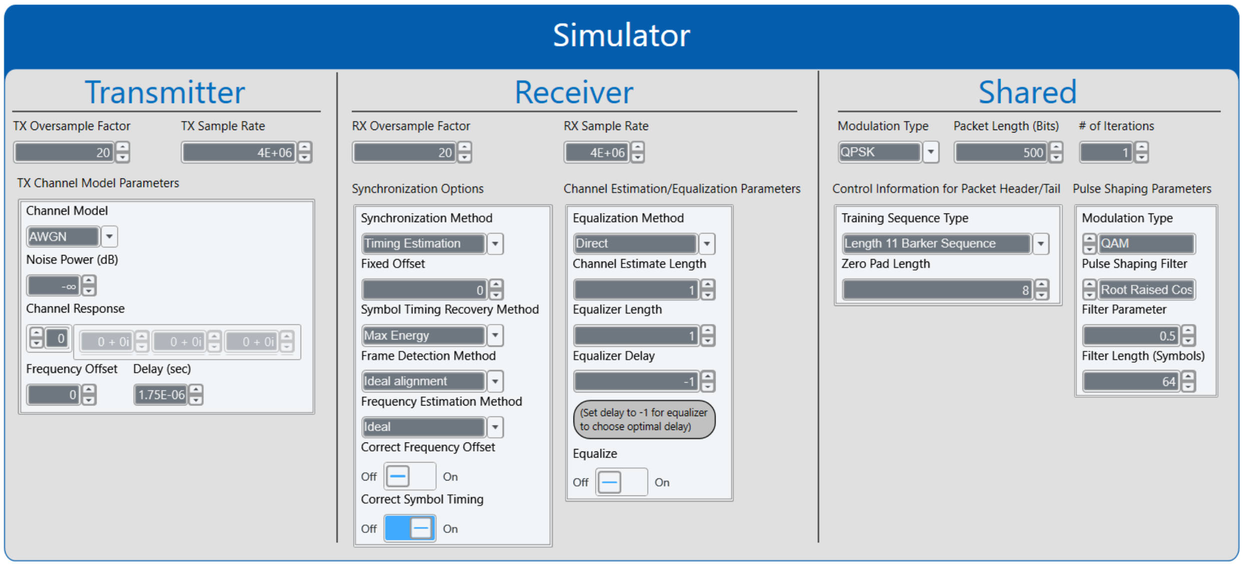
Table 1.1: Inputs and outputs of the block *student_align_max_energy.gvi*.

Hints:

a. In order to take 1 symbol out of M of an array, you can use the block *Decimate (single shot).gvi* which takes as input the array and the decimating factor and which returns the decimated array, starting with the first element and up to the end of the input array. In order start at another index, you should first remove the corresponding first elements of the array thanks to the *Array Subset.gvi* block;

b. In order to add all array elements, use the VI *Add Array Elements.gvi*;

c. The *argmax* operation is implemented in the *Array Max and Min.gvi* block.


Simulator

Transmitter

TX Oversample Factor: 20
TX Sample Rate: 4E+06

TX Channel Model Parameters
Channel Model: AWGN
Noise Power (dB): -∞
Channel Response: 0 | 0 + 0i | 0 + 0i | 0 + 0i
Frequency Offset: 0
Delay (sec): 1.75E-06

Receiver

RX Oversample Factor: 20
RX Sample Rate: 4E+06

Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Max Energy
Frame Detection Method: Ideal alignment
Frequency Estimation Method: Ideal
Correct Frequency Offset: [✓ Off] On
Correct Symbol Timing: Off [✓ On]

Channel Estimation/Equalization Parameters
Equalization Method: Direct
Channel Estimate Length: 1
Equalizer Length: 1
Equalizer Delay: -1
(Set delay to -1 for equalizer to choose optimal delay)
Equalize: [✓ Off] On

Shared

Modulation Type: QPSK
Packet Length (Bits): 500
# of Iterations: 1

Control Information for Packet Header/Tail
Training Sequence Type: Length 11 Barker Sequence
Zero Pad Length: 8

Pulse Shaping Parameters
Modulation Type: QAM
Pulse Shaping Filter: Root Raised Cos
Filter Parameter: 0.5
Filter Length (Symbols): 64

Figure 1.3: Default simulator settings for synchronisation. Pay a particular attention to the field *Symbol Timing Recovery Method* and to the oversampling factors.

2. Test your implementation of the maximum energy method **with the simulator parameters described in Figure 1.3**. In the output field of the simulator, several measures of the delay are present:

   - *Measured channel impairments* $\longrightarrow$ *Delay*: estimation of the delay $\hat{\tau}_d$.
   - *Timing Error Statistic:* Mean square normalised synchronisation error defined as

   $$\epsilon = \mathbb{E}\left[\left|\frac{\hat{\tau}_d - \tau_d}{T}\right|^2\right].$$

     The expectation is taken on the different iterations. In the following, we will monitor this timing error statistic.

   In any case, you should keep in mind that the delay estimation comes from the symbol alignment (this lab) but also from the frame synchronization, and that an erroneous estimation of the delay can come from both parts. This is why by default, we set the frame detection method to *Ideal alignment*.

3. In the *simulator.gvi* block, deactivate the symbol timing operation. Observe the impact of a delay (for instance for $\tau_d/T = 0, 0.25, 0.5, 0.75, 1$) on the constellation and on the eye diagram, in the absence of noise. *Hint: the Tips section (p10) explains how the symbol period is linked to the sample rates and oversampling factors.*

Question 6: Discuss the impact of a synchronisation error on the constellations and on the eye diagrams.

4. Set again the delay to 1.75μs. In your block *align_MaxEnergy.gvi*, probe the signal $J_{\text{approx}}[k]$ and compare it to the theoretic curves for various roll-off factors $\alpha \in [0\,,\,1]$ (*Filter Parameter* in the LabVIEW panel). Repeat this operation for a raised cosine pulse shaping filter. Do it also for a rectangular pulse shaping filter. Remember that for the rectangular pulse shaping filter there is no α parameter. *Hint: To compare the different filters, it might be a good idea to scale (i.e. multiply by a factor) the different energy curves in post processing such that they all have the same maximum, as in the LabVIEW implementation this maximum is impacted by, among others, the filter length, the number of symbols, etc.*

Question 7: How does the raised cosine curve compare to the root-raised one. Why?

Question 8: How does $J_{\text{approx}}[k]$ evolve when varying α? And when using a rectangular filter?

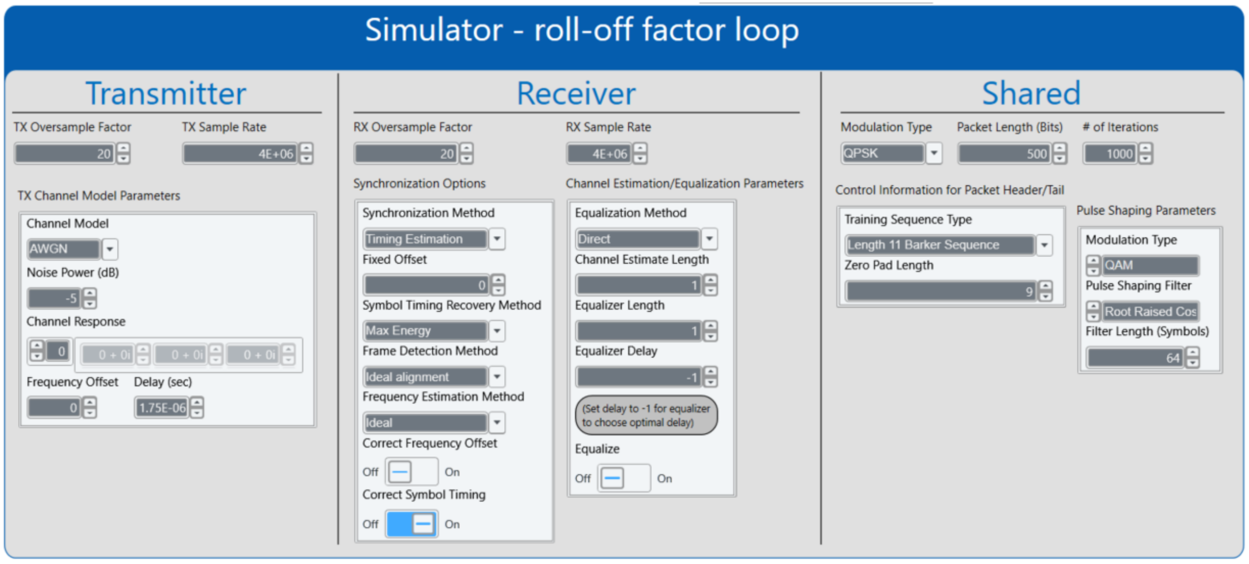
Question 9: **(Bonus)** What happens when $\alpha = 0$? Is this predicted by the analytical results? Why? *(Hint: try to modify the filter length parameter)*

5. Set $\alpha = 1$ and the noise power to -10dB. Probe again $J_{\text{approx}}[k]$ for the three different pulse shaping filters (root-raised cosine, raised cosine, rectangular).

Question 10: One of the three filters is not optimal with respect to the SNR. Which one? Why?

Question 11: How is $J_{\text{approx}}[k]$ impacted by the noise for the different filters? *Hint: We do not expect a discussion on the variation of the minimum/maximum/scale of the energy functions, but rather a discussion on the general behaviour of the curves.*

6. You have been provided with a VI *simulator_alpha_loop.gvi* which computes the error statistics for various $\alpha \in [0\,,\,1]$ in the presence of noise[8]. It has the same interface than the usual simulator at the exception that the roll-off factor cannot be chosen. Instead, the roll-off factor vector must be specified in the *Timing vs alpha* part of the panel. The data of the graph can then be exported to other programs (such as Python or Matlab) in at least two different ways:



Simulator - roll-off factor loop

Transmitter

TX Oversample Factor: 20
TX Sample Rate: 4E+06
TX Channel Model Parameters
Channel Model: AWGN
Noise Power (dB): -5
Channel Response: 0 | 0 + 0i | 0 + 0i | 0 + 0i
Frequency Offset: 0
Delay (sec): 1.75E-06

Receiver

RX Oversample Factor: 20
RX Sample Rate: 4E+06
Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Max Energy
Frame Detection Method: Ideal alignment
Frequency Estimation Method: Ideal
Correct Frequency Offset: [✓ Off] On
Correct Symbol Timing: Off [✓ On]
Channel Estimation/Equalization Parameters
Equalization Method: Direct
Channel Estimate Length: 1
Equalizer Length: 1
Equalizer Delay: -1
(Set delay to -1 for equalizer to choose optimal delay)
Equalize: [✓ Off] On

Shared

Modulation Type: QPSK
Packet Length (Bits): 500
# of Iterations: 1000
Control Information for Packet Header/Tail
Training Sequence Type: Length 11 Barker Sequence
Zero Pad Length: 9
Pulse Shaping Parameters
Modulation Type: QAM
Pulse Shaping Filter: Root Raised Cos
Filter Length (Symbols): 64

Figure 1.4: Default parameters of the roll-off factor loop.

- In the *"Panel"* tab, right-click on the graph and select *"Capture Data"*. The data should appear in the left part of LabVIEW. Right click on the data which has just been created and select *"Export"*. This will create a CSV file from which you will be able to copy past the data points.
- In the *"Panel"* tab, right-click on the BER table and select *"Capture Data"*. The data should appear in the left part of LabVIEW. Double click on the data: a new tab should appear from which

[8]Note that a similar loop is provided to obtain BER-SNR curves, in the *simulator_SNR_loop.gvi* block.

you can directly copy-paste the data. Note that you may need to reduce the zoom in order to copy-paste all the data at once. You can also export the CSV file from this data (it is usually easier than exporting a graph).

With the parameters of Figure 1.4, obtain such curve for a root-raised cosine filter and a raised cosine filter, and for a noise power of -5dB and 0dB. Compare the curves to the error statistics when the pulse shaping is rectangular.

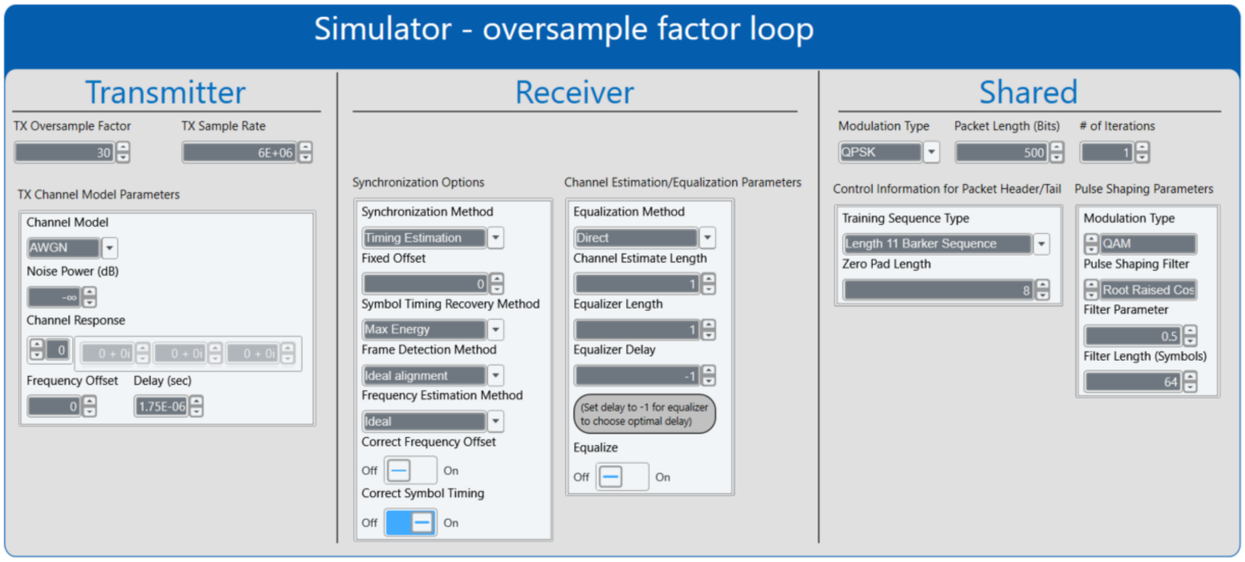
Question 12: How does ϵ evolve with α? How can you explain it based on the theoretic curves?

Question 13: How does the raised cosine curves compare to the RRC ones? Why?

Question 14: How does the root-raised cosine error statistics compare to the one of the rectangular pulse shaping? Why?

---

7. You have also at your disposition the file *simulator_oversample_loop.gvi* which obtains the evolution of the error statistics in function of the receiver oversample factor. To have a fair comparison, the symbol period T is kept constant and the RX sample rate is therefore adapted in function of the oversample factor. Obtain such a curve for $M = 2, 4, \ldots, 120$, with a RRC filter with $\alpha = 1$, with $T = 5\mu$s, $\tau_d = 0.35T$ and a noise power of $-\infty$dB. Note that as we consider a noiseless communication chain, no averaging is needed. The detailed simulator parameters are described in Figure 1.5 (Note that the *TX Oversample factor* and the *TX Sample Rate* have been modified).


Simulator - oversample factor loop

Transmitter
TX Oversample Factor: 30
TX Sample Rate: 6E+06
TX Channel Model Parameters
Channel Model: AWGN
Noise Power (dB): -∞
Channel Response: 0 | 0 + 0i | 0 + 0i | 0 + 0i
Frequency Offset: 0
Delay (sec): 1.75E-06

Receiver
Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Max Energy
Frame Detection Method: Ideal alignment
Frequency Estimation Method: Ideal
Correct Frequency Offset: Off
Correct Symbol Timing: On
Channel Estimation/Equalization Parameters
Equalization Method: Direct
Channel Estimate Length: 1
Equalizer Length: 1
Equalizer Delay: -1
(Set delay to -1 for equalizer to choose optimal delay)
Equalize: Off

Shared
Modulation Type: QPSK
Packet Length (Bits): 500
# of Iterations: 1
Control Information for Packet Header/Tail
Training Sequence Type: Length 11 Barker Sequence
Zero Pad Length: 8
Pulse Shaping Parameters
Modulation Type: QAM
Pulse Shaping Filter: Root Raised Cos
Filter Parameter: 0.5
Filter Length (Symbols): 64

Figure 1.5: Default parameters of the oversample factor loop.

Question 15: How can you explain the behaviour of the curve?

4: Appendix - Expression of the energy function for a root-raised cosine filter

In the below development, we prove that

$$\tilde{J}(\theta) = 1 - \frac{\alpha}{2} + \frac{\alpha}{2}\cos^2\left(\frac{\pi\theta}{T}\right),$$

when $g(t)$ is a root-raised cosine filter with roll-off factor α by 1) using the Parseval theorem, 2) from Fourier transform properties, expressing $\tilde{J}(\theta)$ in function of the spectrum of $C_g(\tau)$, 3) from the expression of the raised cosine spectrum, obtaining the final expression.

Defining $\mathscr{C}_g^\theta\left(e^{j\Omega}\right)$ as the DTFT of $C_g^\theta[m] \triangleq C_g(mT+\theta)$, the Parseval theorem can be used to give

$$\tilde{J}(\theta) = \sum_{m=-\infty}^{\infty}\left|C_g(mT+\theta)\right|^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi}\left|\mathscr{C}_g^\theta\left(e^{j\Omega}\right)\right|^2 \mathrm{d}\Omega.$$

In order to obtain an expression of $\mathscr{C}_g^\theta\left(e^{j\Omega}\right)$, one can notice that $C_g^\theta[m]$ is a sampled version of $C_g^\theta(t) \triangleq C_g(t+\theta)$, which is a shifted version of $C_g(t)$. Hence, using the Fourier transform properties and the sampling theorem, this gives

$$\begin{array}{ccccc} C_g(t) & \overset{\text{shift } \theta}{\longrightarrow} & C_g^\theta(t) = C_g(t+\theta) & \overset{\text{sampling}}{\longrightarrow} & C_g^\theta[m] = C_g^\theta(mT), \\ \mathscr{C}_g(\omega) & \overset{\text{modulation } e^{j\omega\theta}}{\longrightarrow} & \mathscr{C}_g^\theta(\omega) = \mathscr{C}_g(\omega)e^{j\omega\theta} & \overset{\text{repetition}}{\longrightarrow} & \mathscr{C}_g^\theta\left(e^{j\Omega}\right) = \frac{1}{T}\sum_{k=-\infty}^{\infty}\mathscr{C}_g^\theta\left(\frac{\Omega-2\pi k}{T}\right), \end{array}$$

this leading to

$$\mathscr{C}_g^\theta\left(e^{j\Omega}\right) = \frac{e^{\frac{j\Omega\theta}{T}}}{T}\sum_{k=-\infty}^{\infty}\mathscr{C}_g\left(\frac{\Omega-2\pi k}{T}\right)e^{-\frac{j2\pi k\theta}{T}}.$$

Therefore, expressing the Parseval identity with normalised frequency $F = \Omega/2\pi$, we obtain

$$\tilde{J}(\theta) = \int_{-0.5}^{0.5}\left|\sum_{k=-\infty}^{\infty}\frac{1}{T}\mathscr{C}_g(F-k)\,e^{-\frac{j2\pi k\theta}{T}}\right|^2 \mathrm{d}F.$$

The above expression is particularly interesting as the expression of $\mathscr{C}_g(F)$ is simple in the frequency domain (if you remember well the LELEC1360-Telecommunications course, the whole family of RRC filters is defined in the frequency domain in order to satisfy the Nyquist criterion). If the filter $g(t)$ is a root-raised cosine, then its correlation is a raised cosine whose spectrum $\mathscr{C}_g(F)$ (as well as the one of its shifted versions) is represented in Figure 1.6 . The equation of such a frequency response is

$$\mathscr{C}_g(F) = \begin{cases} T & |F| \leq \frac{1-\alpha}{2}, \\ TS(F) & \frac{1-\alpha}{2} \leq |F| \leq \frac{1+\alpha}{2}, \\ 0 & \text{otherwise,} \end{cases}$$

with

$$S(F) \triangleq \frac{1}{2}\left(1 + \cos\left(\frac{\pi}{\alpha}\left(|F| - \frac{1-\alpha}{2}\right)\right)\right).$$

From the figure and from the integration domain, one can notice that the only terms of the sum that will impact the integral are those corresponding to $k = -1, 0, 1$. We obtain thus

$$\begin{aligned}\tilde{J}(\theta) &= \int_{-0.5}^{\frac{-1+\alpha}{2}}\left|S(F+1)e^{\frac{j2\pi\theta}{T}} + S(F)\right|^2 \mathrm{d}F + \int_{\frac{-1+\alpha}{2}}^{\frac{1-\alpha}{2}}\left|\frac{1}{T}T\right|^2 \mathrm{d}F + \int_{\frac{1-\alpha}{2}}^{0.5}\left|S(F) + S(F-1)e^{\frac{-j2\pi\theta}{T}}\right|^2 \mathrm{d}F, \\ &= \int_0^{\frac{\alpha}{2}}\left|S\left(F+\frac{1}{2}\right)e^{\frac{j\pi\theta}{T}} + S\left(F-\frac{1}{2}\right)e^{\frac{-j\pi\theta}{T}}\right|^2 \mathrm{d}F + 1 - \alpha + \int_{\frac{-\alpha}{2}}^{0}\left|S\left(F+\frac{1}{2}\right)e^{\frac{j\pi\theta}{T}} + S\left(F-\frac{1}{2}\right)e^{\frac{-j\pi\theta}{T}}\right|^2 \mathrm{d}F, \\ &= \int_{\frac{-\alpha}{2}}^{\frac{\alpha}{2}}\left|S\left(F+\frac{1}{2}\right)e^{\frac{j\pi\theta}{T}} + S\left(F-\frac{1}{2}\right)e^{\frac{-j\pi\theta}{T}}\right|^2 \mathrm{d}F + 1 - \alpha.\end{aligned}$$

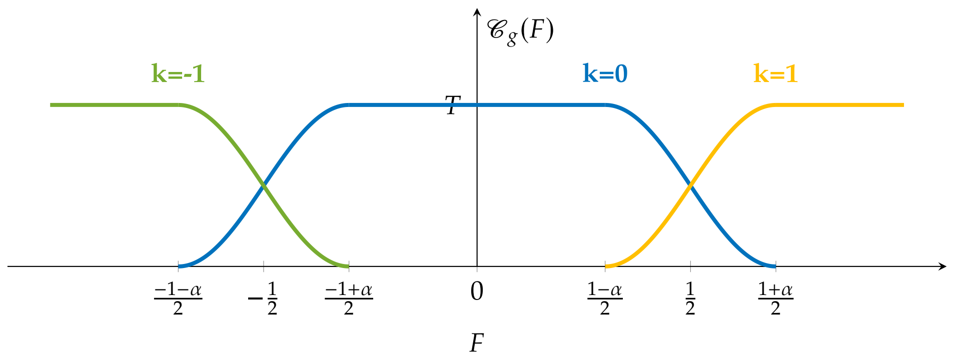


Figure 1.6: Spectrum of a raised cosine (i.e. spectrum of the correlation between two root-raised cosines)

From the definition of $S(F)$ and trigonometric identities, we have that

$$S\left(F+\frac{1}{2}\right)=\frac{1}{2}-\frac{1}{2}\sin\left(\frac{\pi F}{\alpha}\right), \qquad S\left(F-\frac{1}{2}\right)=\frac{1}{2}+\frac{1}{2}\sin\left(\frac{\pi F}{\alpha}\right),$$

this leading to

$$\begin{aligned}
\tilde{J}(\theta) &= \int_{\frac{-\alpha}{2}}^{\frac{\alpha}{2}} \left| \frac{e^{\frac{j\pi\theta}{T}}+e^{\frac{-j\pi\theta}{T}}}{2} + \frac{e^{\frac{-j\pi\theta}{T}}-e^{\frac{j\pi\theta}{T}}}{2}\sin\left(\frac{\pi F}{\alpha}\right)\right|^2 \mathrm{d}F + 1-\alpha,\\
&= \int_{\frac{-\alpha}{2}}^{\frac{\alpha}{2}} \left|\cos\left(\frac{\pi\theta}{T}\right)-\sin\left(\frac{\pi\theta}{T}\right)\sin\left(\frac{\pi F}{\alpha}\right)\right|^2 \mathrm{d}F + 1-\alpha,\\
&= \cos^2\left(\frac{\pi\theta}{T}\right)\int_{\frac{-\alpha}{2}}^{\frac{\alpha}{2}}\mathrm{d}F + \sin^2\left(\frac{\pi\theta}{T}\right)\int_{\frac{-\alpha}{2}}^{\frac{\alpha}{2}}\sin^2\left(\frac{\pi F}{\alpha}\right)\mathrm{d}F + \sin\left(\frac{2\pi\theta}{T}\right)\int_{\frac{-\alpha}{2}}^{\frac{\alpha}{2}}\sin\left(\frac{\pi F}{\alpha}\right)\mathrm{d}F + 1-\alpha,\\
&= \alpha\cos^2\left(\frac{\pi\theta}{T}\right) + \frac{\alpha}{2}\sin^2\left(\frac{\pi\theta}{T}\right) + 0 + 1 - \alpha,\\
&= 1-\frac{\alpha}{2}+\frac{\alpha}{2}\cos^2\left(\frac{\pi\theta}{T}\right),
\end{aligned}$$

which is the expression we were seaking.

Lab 2: Channel characterisation, estimation and equalisation

Part 1 - Channel characterisation

1: Introduction

When several propagation paths exist, the signal processed at the receiver is composed of multiple copies of the transmitted signal. Depending on the propagation delays of these paths, the performance of the receiver are more or less degraded.

The first part of this lab focuses on multipath channels and more precisely on their impact on the constellation, and on the link between the system bandwidth and the so-called channel frequency selectivity.

2: Theoretical Background: From flat to frequency selective channel

Channel modelling

Until now, we have considered a simple channel model named the flat fading channel, which reads in a complex baseband representation as

$$c(t) = ae^{j\phi}\delta(t-\tau),$$

where a is an attenuation, ϕ a phase shift and τ a delay. With this channel model, the composite channel is given by

$$h(t) = (g_{\text{tx}} * c * g_{\text{rx}})(t) = ae^{j\phi}\int_{t'} g_{\text{tx}}(t'-\tau)g^*_{\text{rx}}(t'-t)\mathrm{d}t = ae^{j\phi}C_g(t-\tau),$$

and the receiver must process the signal

$$y(t) = \sum_m I[m]h(t-mT) + v(t),$$

After sampling, denoting $h[n] = h(nT)$ what is usually called *channel taps*, it becomes

$$y[n] = \sum_m I[m]h[n-m] + v[n] = h[0]I[n] + \underbrace{\sum_{m\neq n} h[m]I[n-m]}_{\text{ISI}} + v[n].$$

Depending on the value of the delay τ, you have studied in Lab 1 the impact of the induced ISI on the performance of the receiver and you have applied a maximum energy-based timing recovery method to correct this delay and minimise the ISI.

However, in practice, the transmitted signal encounters different propagation effects including reflections, scattering, diffraction and path-loss. Consequently, multiple propagation paths should be considered between the transmitter and the receiver, each with its own attenuation, phase shift and delay (*multipath channel model*):

$$c(t) = \sum_i a_i e^{j\phi_i}\delta(t-\tau_i).$$

In that case, the composite channel is given by

$$h(t) = \sum_i a_i e^{j\phi_i}\int_{t'} g_{\text{tx}}(t'-\tau_i)g^*_{\text{rx}}(t'-t)\mathrm{d}t' = \sum_i a_i e^{j\phi_i}C_g(t-\tau_i). \tag{2.1}$$

Again, the receiver experiences ISI. If the path delay differences[9] $\Delta\tau_{ij} = \tau_i - \tau_j$ are not significant compared to the symbol period T, the timing recovery method helps to minimise this distortion efficiently. However, if it is not the case, the symbol synchronisation technique of Lab 1 cannot correct anymore the induced ISI owing to the presence of significantly different delays. In the first case, we remain in a *frequency flat fading* scenario. In the second case, we are in a *frequency selective fading* scenario.

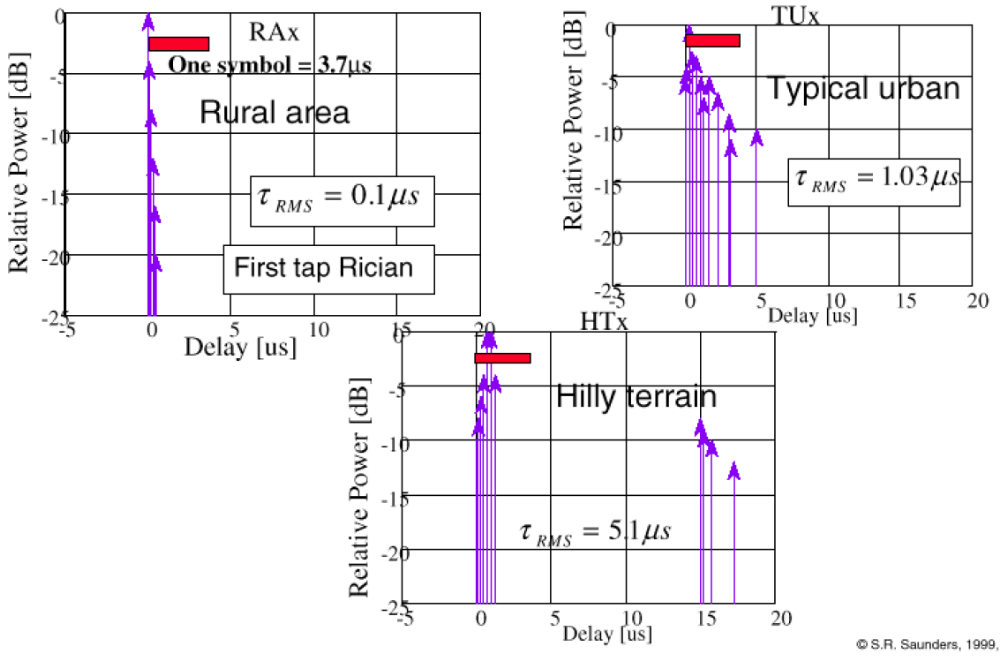


Figure 2.1: Power and delay of channel paths in different scenarios, compared to the symbol period. The rural area experiences flat fading, while the typical urban area and the hilly terrain experience frequency selective fading.

Note that the frequency selectivity of the channel depends on the relation between the path delay differences and the symbol period. Therefore, it depends on the environment topology, but also on the transmission system. Since an increase (resp. decrease) of the symbol period, or equivalently a decrease (resp. increase) of the system bandwidth, leads to an higher probability for a scenario to be in flat (resp. frequency selective) fading, the system used in that case is also named a *narrowband (resp. wideband) system*.

Owing to the physical nature of the channel, it is assumed that the channel taps are modelled as a causal Finite Impulse Response (FIR) filter. Moreover, the signal energy decays as a function of the distance, and there are energy losses at each reflection. Finally, the weakest path contributions fall below the noise threshold and aren't detected at the receiver. Therefore, we assume that

$$h[n] = 0 \;\forall\, n < 0 \quad \& \quad h[n] = 0 \;\forall\, n \geq L,$$

where L is the channel length. Note that the composite channel coefficients $h[n]$ are unknown at the receiver and must generally be estimated (see Part 2).

Question 1: Bonus - What is the relation between the the DFT of the channel taps and the spectrum of $h(t)$?

[9]A metric named *channel delay spread* helps to evaluate the amplitude of these path delay differences, weighted by the power of the paths.

Power Delay Profile

In order to evaluate the environment in which the transmission occurs, the *power delay profile* helps to estimate the power of each channel tap. It is defined as

$$\mathcal{P}[n] = \mathbb{E}\left[|h[n]|^2\right],$$

where $\mathbb{E}$ is the expectation operator. This measure is meaningful if the channel is *wide sense stationary*, i.e. the channel distribution does not change over time. In our case, we will estimate the power delay profile using an estimation of the channel taps $\hat{h}$. Moreover, thanks to *ergodicity*, the expectation operator can be replaced by a time average. Therefore, with P estimates $\hat{h}_p$ of the channel taps, an estimate of the power delay profile is given by

$$\hat{\mathcal{P}}[n] = \frac{1}{P}\sum_p |\hat{h}_p[n]|^2.$$

Question 2: Suppose that you are transmitting symbols with symbol period T over a two-path channel with impulse response

$$c(t) = \delta(t-T) + 0.5\,\delta(t-T-\tau),$$

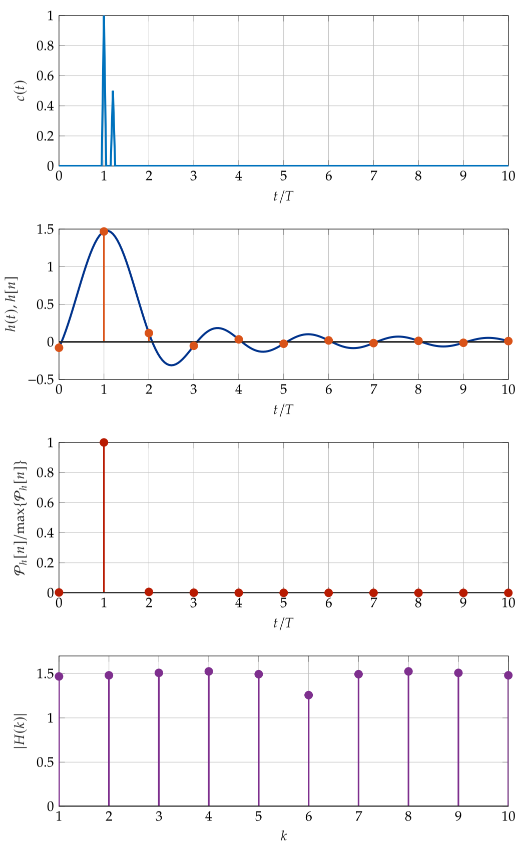
You are transmitting symbols using a sinc pulse shaping filter[a] with bandwidth $1/T$:

$$\begin{aligned} g_{\text{tx}}(t) &= \text{sinc}\left(\frac{t}{T}\right) = \frac{\sin\left(\pi\frac{t}{T}\right)}{\pi\frac{t}{T}}, \\ g_{\text{rx}}(t) &= \delta(t). \end{aligned}$$

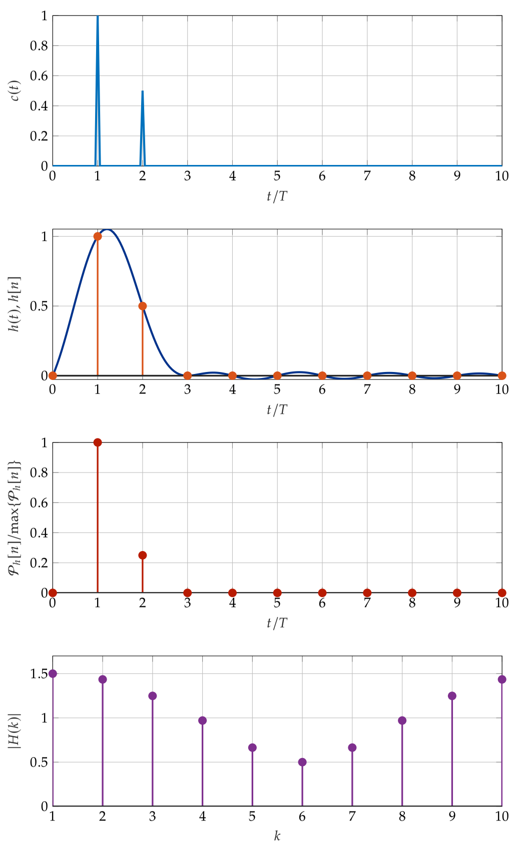
1. What are the impulse response and frequency response of the composite channel $h = g_{\text{tx}} * c * g_{\text{rx}}$ in continuous time?
   Hint:
   $$x(t) = B\,\text{sinc}(Bt) \quad \overset{\text{FT}}{\longleftrightarrow} \quad X(f) = \begin{cases} 1 & \text{if } |f| \leq \dfrac{B}{2}, \\ 0 & \text{otherwise..} \end{cases}$$
2. Let us consider $\tau/T = \{0.2, 1, 1.5\}$ (either the delay remains constant but several systems with different symbol periods are employed, or the delay changes but the system stays the same). The following graphs of Figure 2.2 illustrates
   - the channel impulse response;
   - the composite channel impulse response in continuous and discrete time;
   - the (normalised) power delay profile of the composite channel;
   - the 10-taps DFT of the discrete time composite channel.

   Observe how the channel $c(t)$ impacts the composite channel $h(t)$ and $h[n]$, the PDP and the frequency taps in the different cases.
3. The channel is (strictly) flat if the spectrum of the channel taps is constant, and (strictly) frequency selective otherwise. How the strict frequency selectivity of the channel is linked to the number of channel taps?
4. The *coherence bandwidth* is the bandwidth for which the channel frequency response is supposed to be flat (variation of maximum 3dB). The following graphs in Figure 2.3 illustrates the composite channel frequency response in normal scale and log scale, for $\tau/T = \{0.2, 1, 1.5\}$, with τ fixed. Evaluate the coherence bandwidth. For which system the considered bandwidth is smaller than the coherence bandwidth (*flat fading scenario*) or larger than the coherence bandwidth (*frequency selective fading scenario*)?

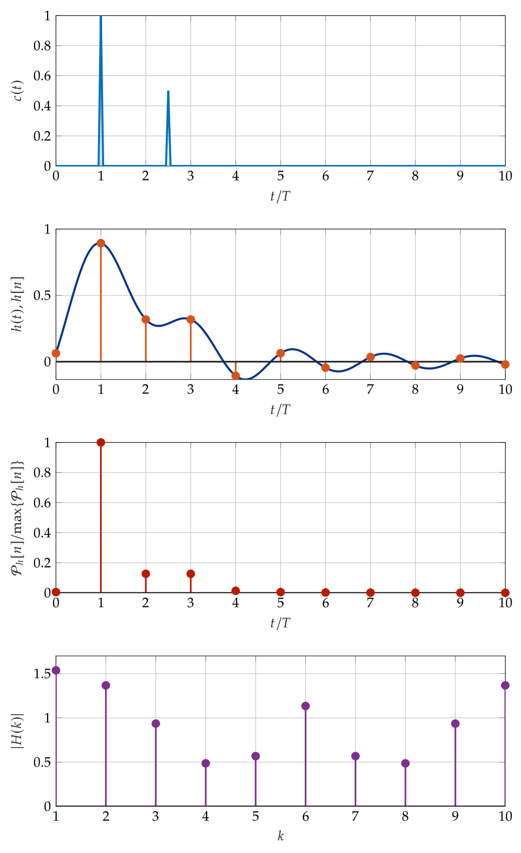
[a]Note that there is no matched filter with a sinc pulse shaping filter to fulfill the Nyquist criterion.



(a)



(b)



(c)

Figure 2.2: Channel and composite channel analysis in frequency flat and frequency selective scenarios.

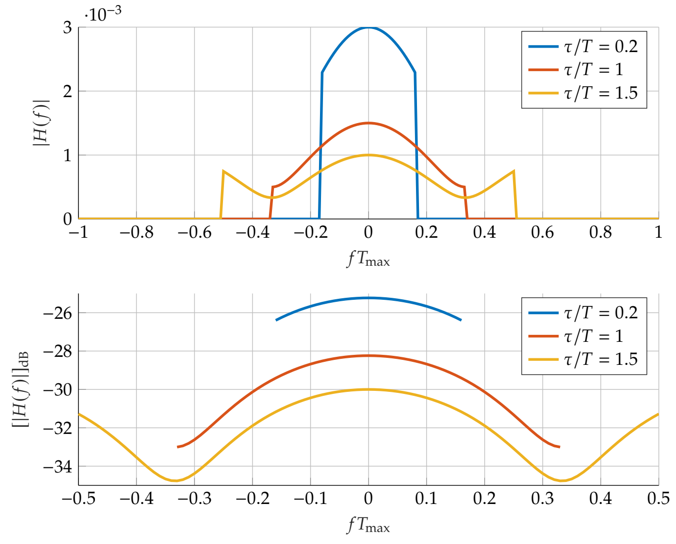


Figure 2.3: Composite channel frequency response (τ fixed).

3: Lab Instructions

1. First, let us analyse the impact of multipath on the received constellation. The default simulator settings are given in Figure 2.4. Note that it is important to set the channel model to ISI, otherwise LabVIEW will not take into account the channel coefficients. Set a noise power of $-\infty$dB and turn off the equalisation. Observed the received constellation when $\mathbf{h} = [1 \quad 0.5]$, $\mathbf{h} = [1 \quad 0.35 + 0.35j]$ and $\mathbf{h} = [0.86 + 0.5j \quad 0.35 + 0.35j]$. *Hint: Verify that your channel model is set to ISI.*

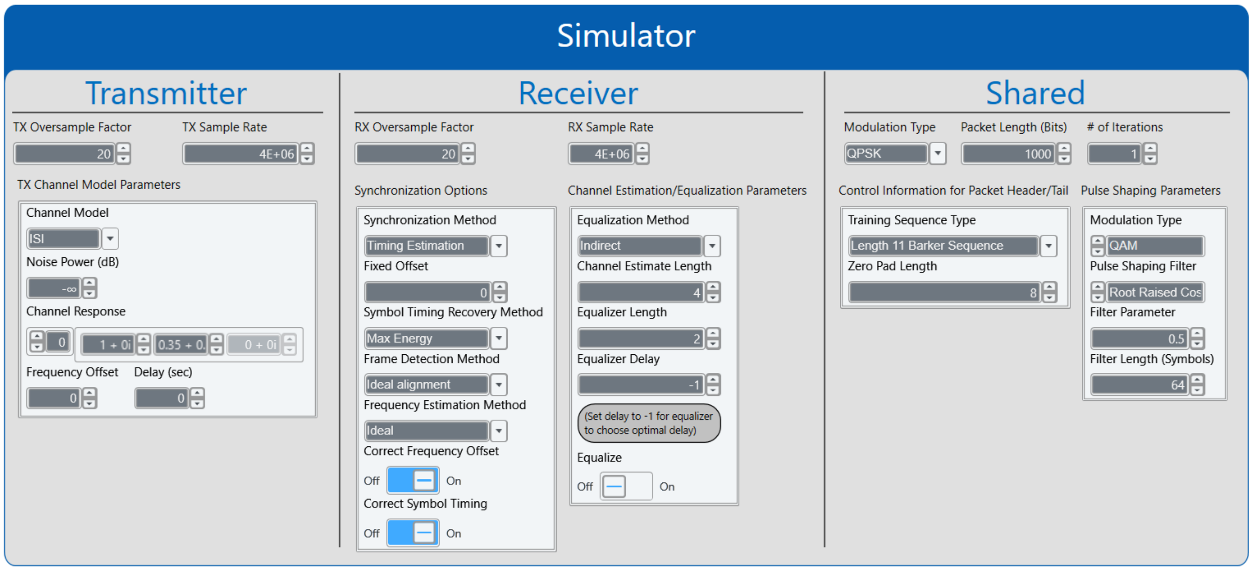
Question 3: What happens to the constellation? How can you explain it? *Hint: The different channels can also be written as* $\mathbf{h} = [1e^{j0} \quad 0.5e^{j0}]$, $\mathbf{h} = [1e^{j0} \quad 0.5e^{j\frac{\pi}{4}}]$, $\mathbf{h} = [1e^{j\frac{\pi}{6}} \quad 0.5e^{j\frac{\pi}{4}}]$.

---

2. Now, let us observe the received constellation over a real channel using the USRPs. In order to do that, set back the channel model to AWGN. A communication through the USRPs will be established directly thanks to the *top_tx.gvi* and *top_rx.gvi* blocks. First, choose a pair of USRPs and set up the IP address and the carrier frequency following the informations given by the teaching assistants. Ensure that each pair of USRPs is used by **only one group at a time** to avoid corrupted transmissions. Then, set the other parameters as shown in Figure 2.5 and 2.6. To transmit a packet, launch the *top_tx.gvi* block first, and *top_rx.gvi* block after.

Question 4: Let us consider a narrowband and wideband system, respectively with bandwidths of 250kHz and 5MHz. Observe the obtained constellations for several successive receptions. Ignoring cases where the transmission has failed (owing to a synchronisation error for example), what can you observe? Does it fit your expectations?
Hint: to obtain the required bandwidths, set the sample rate at 1Msps (resp. 20Msps) with an oversampling factor of 4 at the emitter and receiver. Don't forget to adapt the capture time depending on the packet duration!

Question 5: Describe the power delay profile and the frequency responses obtained for each channel. Is the channel frequency flat or frequency selective ?

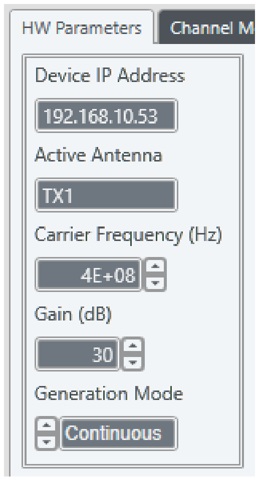

Simulator

Transmitter
TX Oversample Factor: 20
TX Sample Rate: 4E+06
TX Channel Model Parameters
Channel Model: ISI
Noise Power (dB): -∞
Channel Response: 0 | 1 + 0i | 0.35 + 0i | 0 + 0i
Frequency Offset: 0
Delay (sec): 0

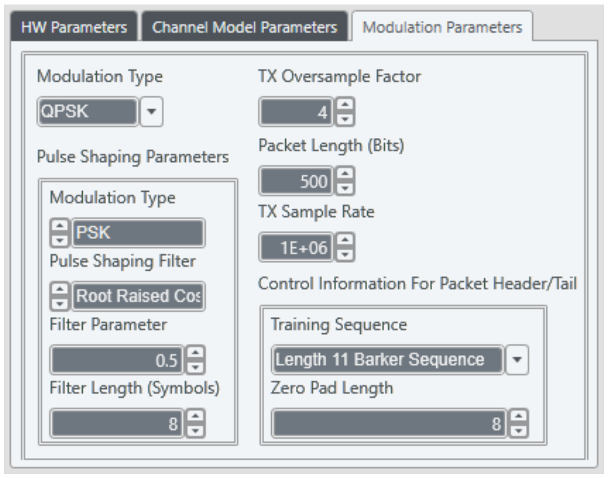
Receiver
RX Oversample Factor: 20
RX Sample Rate: 4E+06
Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Max Energy
Frame Detection Method: Ideal alignment
Frequency Estimation Method: Ideal
Correct Frequency Offset: Off / On
Correct Symbol Timing: Off / On
Channel Estimation/Equalization Parameters
Equalization Method: Indirect
Channel Estimate Length: 4
Equalizer Length: 2
Equalizer Delay: -1
(Set delay to -1 for equalizer to choose optimal delay)
Equalize: Off / On

Shared
Modulation Type: QPSK
Packet Length (Bits): 1000
# of Iterations: 1
Control Information for Packet Header/Tail
Training Sequence Type: Length 11 Barker Sequence
Zero Pad Length: 8
Pulse Shaping Parameters
Modulation Type: QAM
Pulse Shaping Filter: Root Raised Cos
Filter Parameter: 0.5
Filter Length (Symbols): 64

Figure 2.4: Default simulator settings for equalisation.

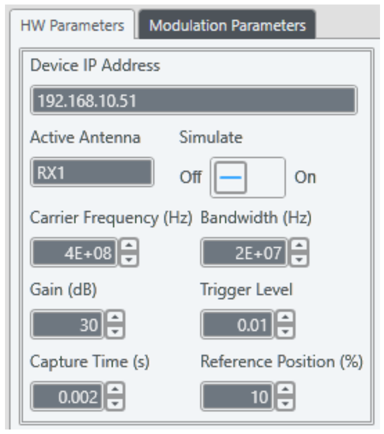

HW Parameters | Channel M
Device IP Address: 192.168.10.53
Active Antenna: TX1
Carrier Frequency (Hz): 4E+08
Gain (dB): 30
Generation Mode: Continuous

(a) HW Parameters


HW Parameters | Channel Model Parameters | Modulation Parameters
Modulation Type: QPSK
Pulse Shaping Parameters
Modulation Type: PSK
Pulse Shaping Filter: Root Raised Cos
Filter Parameter: 0.5
Filter Length (Symbols): 8
TX Oversample Factor: 4
Packet Length (Bits): 500
TX Sample Rate: 1E+06
Control Information For Packet Header/Tail
Training Sequence: Length 11 Barker Sequence
Zero Pad Length: 8

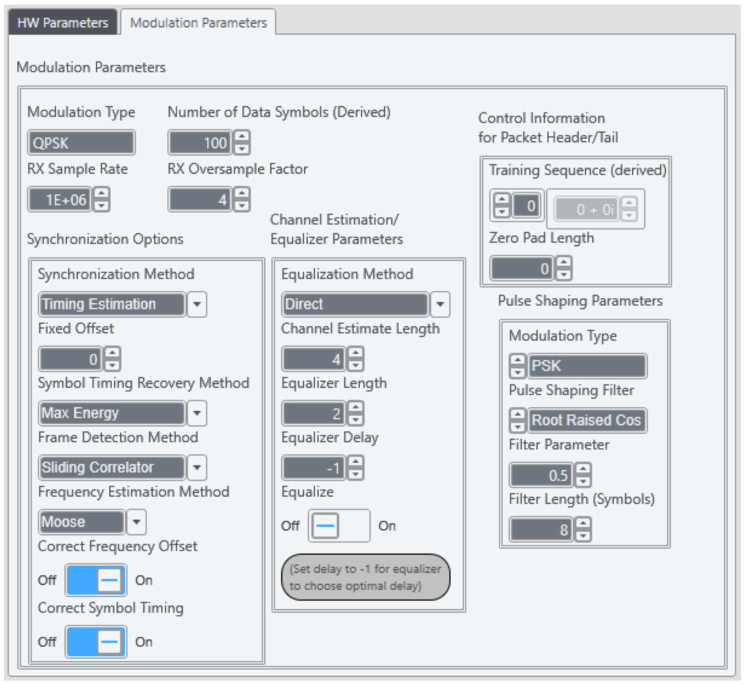
(b) Modulation Parameters

Figure 2.5: Hardware and modulation parameters for *top_tx.gvi*.



Device IP Address: 192.168.10.51
Active Antenna: RX1
Simulate: Off / On
Carrier Frequency (Hz): 4E+08
Bandwidth (Hz): 2E+07
Gain (dB): 30
Trigger Level: 0.01
Capture Time (s): 0.002
Reference Position (%): 10

(a) HW Parameters



Modulation Parameters
Modulation Type: QPSK
Number of Data Symbols (Derived): 100
RX Sample Rate: 1E+06
RX Oversample Factor: 4
Control Information for Packet Header/Tail
Training Sequence (derived): 0, 0 + 0i
Zero Pad Length: 0
Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Max Energy
Frame Detection Method: Sliding Correlator
Frequency Estimation Method: Moose
Correct Frequency Offset: Off / On
Correct Symbol Timing: Off / On
Channel Estimation/ Equalizer Parameters
Equalization Method: Direct
Channel Estimate Length: 4
Equalizer Length: 2
Equalizer Delay: -1
Equalize: Off / On
(Set delay to -1 for equalizer to choose optimal delay)
Pulse Shaping Parameters
Modulation Type: PSK
Pulse Shaping Filter: Root Raised Cos
Filter Parameter: 0.5
Filter Length (Symbols): 8

(b) Modulation Parameters

Figure 2.6: Hardware and modulation parameters for *top_rx.gvi*.

Part 2 - Channel Estimation and Equalization

4: Introduction

In the first part of this lab, you have seen that depending on the system parameters, the channel might be frequency flat or frequency selective. From this first part, you should also know that a frequency selective channel will induce ISI, and therefore will strongly deteriorate the performance of the communication chain.

This second part focuses thus on the equalisation operation, which tries to counteract the distorsion of the channel. Its goal is on the one hand to understand how one can estimate the channel. On the other hand, we will study the implementation of the well-known Wiener filter which has been seen during the lectures. This filter will moreover be compared to the zero-forcing equaliser.

5: Theoretical Background

In order to counteract the channel effect, we remind the reader that we have at our disposition the output of the matched filter which reads as

$$y[n] = \sum_{m=-\infty}^{\infty} I[m]h[n-m] + v[n],$$

with

$$v[n] = \int_{-\infty}^{\infty} w(t) g_{\mathrm{rx}}^{*}(t-nT)\mathrm{d}t,$$

which is white if the filters respect the Nyquist criterion:

$$\mathbb{E}\left[v[k]v^{*}[l]\right] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathbb{E}\left[w(t)w^{*}(t^{'})\right] g_{\mathrm{rx}}^{*}(t-lT)g_{\mathrm{rx}}(t^{'}-kT)\mathrm{d}t\mathrm{d}t^{'} = 2N_0\,\delta[k-l],$$

assuming $E_g = 1$. Based on this discrete output, the goal is to design a filter $\mathbf{w} = \{w[k]\}_{k=-K_1}^{K_2}$ which will estimate the sent symbols as follows:

$$\hat{I}[n] = \sum_{k=-K_1}^{K_2} w[k]y[n-k].$$

Different type of equalizers can be designed. Two of them will be covered in this lab: 1) the Wiener equaliser 2) the zero-forcing equaliser. The goal will be to compare these two methods in terms of accuracy and complexity. Throughout this lab, we assume that we have an accurate estimation of the channel coefficients $\mathbf{h}$ and of the noise PSD $2N_0$. Methods enabling an estimation of these quantities are presented in the lab appendix.

Wiener equaliser The equaliser is designed in order to minimize the mean square error, i.e.

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \mathbb{E}\left[|e[n]|^2\right],$$

with $e[n] = I[n] - \hat{I}[n]$. During the lectures, you have learned how to derive the optimum coefficients of this filter. **You need to pay attention that in the lecture slides, the Wiener filtering is applied prior to the matched filter, and that h is the convolution between the pulse shaping and the channel, without the matched filter. This is more general in theory because no assumption is made on the operations made at the reception. However, in our case, we only have an estimation on the whole composite channel, and this is why we consider the output of the matched filter for the Wiener filtering.** In this lab, we will start from the orthogonality principle which reads as

$$\begin{aligned} \mathbb{E}\left[e[n]y^*[n-l]\right] &= 0 \qquad \forall l = -K_1, \ldots, K_2, \\ \iff \mathbb{E}\left[I[n]y^*[n-l]\right] &= \mathbb{E}\left[\hat{I}[n]y^*[n-l]\right] \qquad \forall l = -K_1, \ldots, K_2. \end{aligned}$$

Defining the auto-correlation of the channel as

$$f[p] \triangleq \sum_{m=-\infty}^{\infty} h^*[m]h[m+p],$$

and expanding the above equations, the Wiener criterion boils down to

$$\sigma_I^2 h^*[-l] = \sum_{k=-K_1}^{K_2} w[k]\left(\sigma_I^2 f^*[k-l] + 2N_0\delta[l-k]\right), \qquad \forall l = -K_1 \leq l \leq K_2,$$

which, using the conjugate symmetry property of $f[p]$, can equivalently be written in a matrix form as

$$\underbrace{\sigma_I^2 \begin{bmatrix} h^*[K_1] \\ h^*[K_1-1] \\ \vdots \\ h^*[-K_2] \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} \sigma_I^2 f[0] + 2N_0 & \sigma_I^2 f^*[1] & \ldots & \sigma_I^2 f^*[K_1+K_2] \\ \sigma_I^2 f[1] & \sigma_I^2 f[0] + 2N_0 & \ldots & \sigma_I^2 f^*[K_1+K_2-1] \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_I^2 f[K_1+K_2] & \sigma_I^2 f[K_1+K_2-1] & \ldots & \sigma_I^2 f[0] + 2N_0 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} w[-K_1] \\ w[-K_1+1] \\ \vdots \\ w[K_2] \end{bmatrix}}_{\mathbf{w}}.$$

Question 6: Building on the lecture developments, obtain this final matrix equation starting from the orthogonality principle.

Question 7: Simplify the above matrix equation when $h[k] = \delta[k]$, i.e. when the channel is AWGN.

At this point, we are able to obtain the Wiener filter if we know the composite discrete channel $\mathbf{h}$, the received noise power $2N_0$ and the transmit symbol power σ_I^2. You can observe that one does not need to know the physical channel $c(t)$ but that the estimation of $\mathbf{h}$ is sufficient. While the noise power can be estimated during the channel estimation, the transmit symbol power σ_I^2 can be assumed equal to 1. Another big advantage of the Wiener method is that we can obtain a closed-form expression of the mean square error:

$$\begin{aligned}\mathbb{E}\left[|e[n]|^2\right] &= \sigma_I^2 - \sum_{k=-K_1}^{K_2} w[k]\sigma_I^2 h[-k], \\ &= \sigma_I^2 - \mathbf{b}^H\mathbf{w}.\end{aligned}$$

Monitoring this MSE will enable us to choose the best possible K_1, K_2 for a given filter length. Note that the above expression is in theory purely real as $\mathbf{w} = \mathbf{A}^{-1}\mathbf{b}$. However, in practice, it will be necessary to take the real part of the above expression due to numerical errors.

Question 8: (Bonus) Building on the lecture developments, derive this expression of the MMSE.

Zero-forcing equaliser The Wiener equaliser, while being relatively simple in its final form, is demanding in terms of analytical developments. Another approach is the zero-forcing one which designs the equaliser by neglecting the impact of the noise. In this case, the noise-free received signal can be written as

$$y^{\text{nf}}[n] = (I * h)[n].$$

The goal is thus to find a filter $w[k]$ s.t. $(h * w)[k] \simeq \delta[k]$, hence leading to

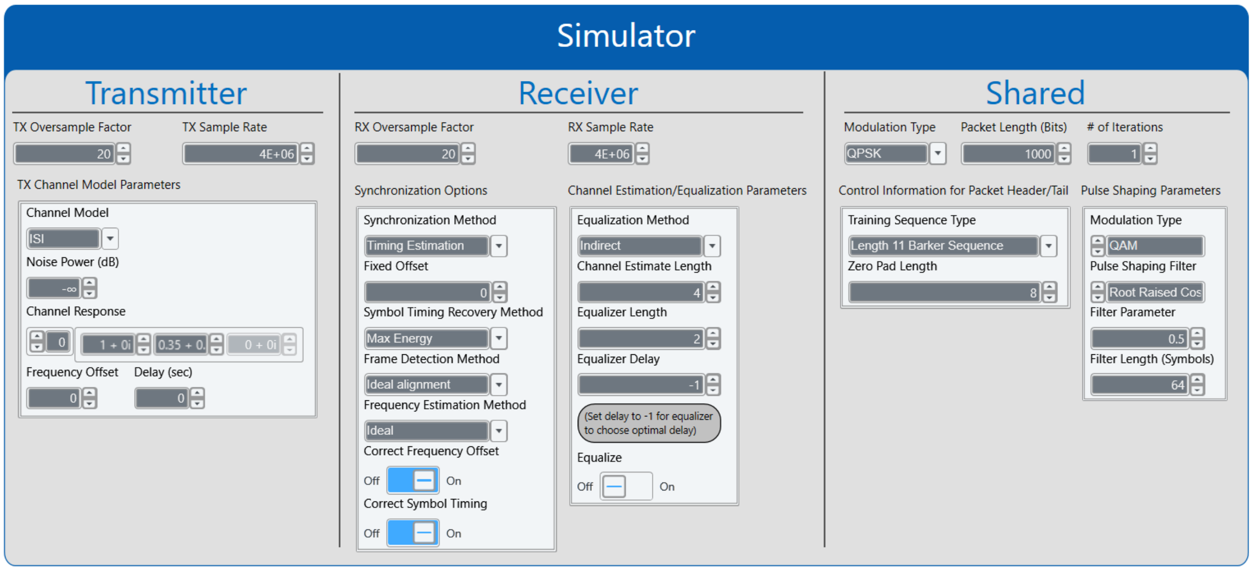
$$\left(y^{\text{nf}} * w\right)[n] = (I * h * w)[n] \simeq I[n].$$

Setting apart the index for which w and h are 0's, we obtain

$$\sum_{k=-K_1}^{K_2} h[l-k]w[k] \simeq \delta[l], \qquad l = -K_1, \dots, L-1+K_2,$$

which can equivalently be rewritten as

$$\underbrace{\begin{bmatrix} h[0] & 0 & \cdots & 0 \\ h[1] & h[0] & & \\ \vdots & \vdots & & \\ h[L-1] & h[L-2] & \vdots & \vdots \\ 0 & h[L-1] & & \\ & 0 & \ddots & \\ & & 0 & \\ & & h[0] & 0 \\ \vdots & \vdots & h[1] & h[0] \\ & & \vdots & \vdots \\ & & h[L-1] & h[L-2] \\ 0 & \cdots & 0 & h[L-1] \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} w[-K_1] \\ \vdots \\ w[K_2] \end{bmatrix}}_{\mathbf{w}} = \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\mathbf{e}} \begin{matrix} \\ \\ \\ \longleftarrow l = 0. \\ \\ \\ \\ \end{matrix}$$


Simulator

Transmitter

TX Oversample Factor: 20
TX Sample Rate: 4E+06
TX Channel Model Parameters
Channel Model: ISI
Noise Power (dB): -∞
Channel Response: 0 | 1 + 0i | 0.35 + 0. | 0 + 0i
Frequency Offset: 0
Delay (sec): 0

Receiver

RX Oversample Factor: 20
RX Sample Rate: 4E+06
Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Max Energy
Frame Detection Method: Ideal alignment
Frequency Estimation Method: Ideal
Correct Frequency Offset: Off / On
Correct Symbol Timing: Off / On
Channel Estimation/Equalization Parameters
Equalization Method: Indirect
Channel Estimate Length: 4
Equalizer Length: 2
Equalizer Delay: -1
(Set delay to -1 for equalizer to choose optimal delay)
Equalize: Off / On

Shared

Modulation Type: QPSK
Packet Length (Bits): 1000
# of Iterations: 1
Control Information for Packet Header/Tail
Training Sequence Type: Length 11 Barker Sequence
Zero Pad Length: 8
Pulse Shaping Parameters
Modulation Type: QAM
Pulse Shaping Filter: Root Raised Cos
Filter Parameter: 0.5
Filter Length (Symbols): 64

Figure 2.7: Default simulator settings for equalisation.

Except in very particular cases, the above equation cannot be solved exactly. Hence, we can however consider its minimum square solution, i.e.

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \|\mathbf{H}\mathbf{w} - \mathbf{e}\|_2^2 ,$$

which can be proven to be equal to

$$\hat{\mathbf{w}} = \left(\mathbf{H}^H\mathbf{H}\right)^{-1} \mathbf{H}^H\mathbf{e}.$$

Again, one can provide a measure on how good the equalization is by monitoring the minimum square error achieved which is given by $\|\mathbf{H}\hat{\mathbf{w}} - \mathbf{e}\|_2^2$.

Question 9: In both cases (Wiener or ZF), we are able to monitor a MSE which measures the quality of the equalization. Can we compare both MSEs? Why?

Question 10: In the above matrix equation, what is the size of the matrix $\mathbf{H}$? With regards to this size, explain one can say that we need an infinite impulse response to counteract the channel effect?

6: Lab instructions

The default simulator settings are given in Figure 2.7. Note that it is important to set the channel model to ISI, otherwise LabVIEW will not take into account the channel coefficients.

1. Implement the Wiener equalizer, whose specification are given in Table 2.1, its template being already given. Note that in the LabVIEW project, instead of specifying K_1, K_2, one specifies the filter length L_f and the filter delay n_d (i.e. the index of the filter center). The link between these two descriptions is given by

$$K_1 = n_d,$$
$$K_2 = L_f - n_d - 1.$$

Note that by setting $n_d = -1$, all possible delays are tested for a given filter length. Then, the delay giving the smallest MSE is selected. Check that your equalization works by setting the channel to an AWGN one and comparing your filter to your answer of Question 7.

student_Wiener_Equalizer.gvi - computes the coefficients of the Wiener filter and the associated MSE.			
Inputs	*Modulation Parameters In*	Cluster	Transmission chain parameters.
	Channel Estimate	CDB 1D array	Estimation of the channel taps.
	Equalizer length	I32	Equalizer length L.
	Equalizer Delay	I32	Equalizer delay (i.e. index of the filter center).
	var_symb	DBL	Variance of the symbols.
	var_noise	DBL	Variance of the noise.
Outputs	*Modulation Parameters Out*	Cluster	Transmission chain parameters.
	Wiener filter coeff	CBD 1D array	Coefficients of the Wiener filter.
	MSE	DBL	MSE of the Wiener filter.

Table 2.1: Wiener filter computation

Hints:

a. You have been provided with the block *compute_channel_cross_correlation.gvi*, which, based on the channel estimate $\mathbf{h}$, the equaliser length and delay, computes the channel cross correlation $f[p]$ as $[f[0], \ldots, f[K_1 + K_2]]$. This block also outputs the formatted channel coefficients as $[h[-K_2], \ldots, h[K_1]]$.

b. The matrix $\mathbf{A}$ is of the Toeplitz type, meaning that each line and column is a shifted version of the previous one. It can be efficiently build thanks to the block *Create Special Matrix.gvi* which works on the basis of the first row and column. In also requires a code specific to each matrix type which can found in the documentation.

c. The block *Solve Linear Equations.gvi* efficiently solve equation systems.

d. The operation $\mathbf{b}^H\mathbf{w}$ is the usual scalar product between vectors $\mathbf{b}$ and $\mathbf{w}$, which can be handled by the block *Dot Product.gvi*. Since this block output type is CDB, don't forget to make the conversion in DBL, by taking the real part for example.

2. Letting $\mathbf{h} = [1 \quad 0.35 + 0.35j]$, observe the filter coefficients you obtain for different noise powers (e.g. $-\infty$dB, -10dB, 0dB), with $K_1 = K_2 = 5$ (i.e. $L_f = 11$, $n_d = 5$), and *Channel estimate length*= 2. You have also been provided with the ZF equalizer (named *indirect* in the LabVIEW project). Also obtain the corresponding filter coefficients.

Question 11: Compare the coefficients of the different filters. (You can focus on the real part only).

3. You have been provided with the file *simulator_SNR_loop_equalisers.gvi*, which computes the BER for a range of SNRs. Obtain such curves for $\mathbf{h} = [1 \qquad 0.35 + 0.35j]$, $L_f = 11$, and for $n_d = -1$. Compare the performance of the two filters (Wiener and ZF). Compare also their performance without equalization, when the channel is AWGN.

Question 12: Comment the BER curves. What happens at high SNR? How can you explain it?

Question 13: The final Wiener equation, which reads as

$$\sigma_I^2 h^*[-l] = \sum_{k=-K_1}^{K_2} w[k] \left(\sigma_I^2 f^*[k-l] + 2N_0 \delta[l-k]\right), \qquad \forall l = -K_1 \leq l \leq K_2,$$

can be written in a concise way as

$$\sigma_I^2 \mathbf{h}^\dagger = \sigma_I^2 \mathbf{w} * \mathbf{h} * \mathbf{h}^\dagger + \sigma_N^2 \mathbf{w},$$

with $\mathbf{h}^\dagger$ which is the flipped complex conjugate of the channel vector. Based on this equation, explain why the Wiener equalizer and the ZF ones are identical at high SNR.

7: Appendix

Channel Estimation

We consider a training sequence of length N_t, denoted by $\{t[n]\}_{n=0}^{N_t-1}$. The output of the MF when this transmit sequence is sent on a channel of length L, reads as

$$y[n] = \sum_{l=0}^{L-1} h[l] s[n-l] + v[n],$$

with $s[m] = t[m]$ if $0 \leq m \leq N_t-1$ while being unknown otherwise. Restricting ourselves to $L-1 \leq n \leq N_t-1$ in order to only depend on known sent symbols, the above equation can equivalently be rewritten in a matrix form as

$$\underbrace{\begin{bmatrix} y[L-1] \\ y[L] \\ \vdots \\ y[N_t-1] \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} t[L-1] & \dots & t[0] \\ t[L] & \dots & t[1] \\ \vdots & \ddots & \vdots \\ t[N_t-1] & \dots & t[N_t-L] \end{bmatrix}}_{\mathbf{T}} \underbrace{\begin{bmatrix} h[0] \\ \vdots \\ h[L-1] \end{bmatrix}}_{\mathbf{h}} + \underbrace{\begin{bmatrix} v[L-1] \\ \vdots \\ v[N_t-1] \end{bmatrix}}_{\mathbf{v}},$$

where $\mathbf{T}$ is a Toeplitz matrix known as the training matrix of size $(N_t - L - 1) \times L$. Good training sequences are usually much longer than the channel length, and such that the Training matrix is full rank. From this matrix equation, we are interested in estimating the channel vector $\mathbf{h}$. If the noise vector $\mathbf{v}$ is AWGN, then the maximum likelihood estimation boils down to the minimum square error method, i.e.

$$\hat{\mathbf{h}} = \arg\min_{\mathbf{h}} \|\mathbf{y} - \mathbf{T}\mathbf{h}\|_2^2,$$

whose solution can be obtained from matrix theoretic results, and reads as

$$\hat{\mathbf{h}} = \left(\mathbf{T}^H \mathbf{T}\right)^{-1} \mathbf{T}^H \mathbf{y}.$$

It should be noted that this $\hat{\mathbf{h}}$ is the estimation of the composite channel, and not an estimation[10] of $c(t)$.

One of the advantage of the above channel estimation method is that it enables to estimate the noise power. Indeed, as shown below, letting $J \triangleq \left\|\mathbf{y} - \mathbf{T}\hat{\mathbf{h}}\right\|_2^2$ be the residual MSE which occurs when performing the channel estimation, the noise power σ_v^2 can be estimated as

$$\hat{\sigma_v^2} = \frac{J}{N_t - 2L - 1} = \frac{\left\|\mathbf{y} - \mathbf{T}\hat{\mathbf{h}}\right\|_2^2}{N_t - 2L - 1}.$$

[10]Estimation of $c(t)$ is much more complex and out of the scope of this course.

Moreover, if the filters are normalised such that $E_g = 1$, we have that $2N_0 = \sigma_v^2$.

Noise power estimation

The minimum square error which occur when estimating the channel can be expressed as

$$\begin{aligned}
J &= \left\|\mathbf{y} - \mathbf{T}\hat{\mathbf{h}}\right\|_2^2, \\
&= \left\|\mathbf{v} + \mathbf{T}\left(\mathbf{h} - \hat{\mathbf{h}}\right)\right\|_2^2, \\
&= \left\|\mathbf{v} - \mathbf{T}\left(\mathbf{T}^H\mathbf{T}\right)^{-1}\mathbf{T}^H\mathbf{v}\right\|_2^2, \\
&= \mathbf{v}^H\left(\mathbf{I}_{N_t-L-1} - \mathbf{T}\left(\mathbf{T}^H\mathbf{T}\right)^{-1}\mathbf{T}^H\right)^2\mathbf{v}, \\
&= \mathbf{v}^H\underbrace{\left(\mathbf{I}_{N_t-L-1} - \mathbf{T}\left(\mathbf{T}^H\mathbf{T}\right)^{-1}\mathbf{T}^H\right)}_{\mathbf{T}}\mathbf{v}, \\
&= \sum_i\sum_j \tilde{T}_{ij}v_i^*v_j.
\end{aligned}$$

Note that the matrix $\mathbf{T}$ is such that $\mathbf{T}^2 = \mathbf{T}$ (a.k.a. the idempotency property). The above expression gives the minimum square error given a noise realisation. The mimimum mean square error is obtained by averaging on the noise, giving thus

$$\begin{aligned}
\mathbb{E}[J] &= \sigma_v^2\sum_i \tilde{T}_{ii}, \\
&= \sigma_v^2\mathrm{Tr}\left\{\tilde{\mathbf{T}}\right\}, \\
&= \sigma_v^2\left(\mathrm{Tr}\left\{\mathbf{I}_{N_t-L-1}\right\} - \mathrm{Tr}\left\{\mathbf{T}\left(\mathbf{T}^H\mathbf{T}\right)^{-1}\mathbf{T}^H\right\}\right), \\
&= \sigma_v^2\left(\mathrm{Tr}\left\{\mathbf{I}_{N_t-L-1}\right\} - \mathrm{Tr}\left\{\mathbf{T}^H\mathbf{T}\left(\mathbf{T}^H\mathbf{T}\right)^{-1}\right\}\right), \\
&= \sigma_v^2\left(\mathrm{Tr}\left\{\mathbf{I}_{N_t-L-1}\right\} - \mathrm{Tr}\left\{\mathbf{I}_L\right\}\right), \\
&= \sigma_v^2\left(N_t - L - 1 - L\right) = \sigma_v^2\left(N_t - 2L - 1\right),
\end{aligned}$$

where we have used i) the fact that if the filters respect the Nyquist criterion, $v[n]$ is a white noise, ii) the cyclic property of the trace (i.e. the fact the $\mathrm{Tr}\{\mathbf{ABC}\} = \mathrm{Tr}\{\mathbf{CAB}\} = \mathrm{Tr}\{\mathbf{BCA}\}$). Thanks to the above equation, we will therefore be able to estimate σ_v^2 as

$$\hat{\sigma_v^2} = \frac{J}{N_t - 2L - 1} = \frac{\left\|\mathbf{y} - \mathbf{T}\hat{\mathbf{h}}\right\|_2^2}{N_t - 2L - 1}.$$

Lab 3: OFDM Modulation and Frequency Domain Equalisation

1: Introduction

The previous lab has shown that frequency selective systems lead to severe degradation of the BER. Even when performing equalisation, it is difficult (one could even say that it is impossible) to achieve a perfect equalisation. Hopefully, a modulation technique called Orthogonal Frequency Division Multiplexing (OFDM) enables an extremely easy equalisation. This property and its other advantages result in the fact that this technique has been widely adopted in last generations cellular networks and in several commercial wireless systems.

The goal of this lab is to understand and implement an OFDM communication chain, as well as frequency-domain equalisation. Moreover, the sensitivity of OFDM systems to frequency offset will be analysed.

2: Theoretical Background

In the previous labs, the considered transmission chain performs a *single carrier transmission*, i.e. symbols are transmitted through a single bandwidth located around a unique carrier frequency, and the orthogonality between the symbols is maintained through *Time Division Multiplexing* (TDM). With the OFDM modulation, the symbols s obtained after the symbol mapping are allocated to different orthogonal carrier frequencies called *subcarriers* (*multi carrier transmission*). Therefore, symbols are transmitted by blocks, and the orthogonality between the symbols is maintained through *Frequency Division Multiplexing* (FDM). Such a chain is presented in Figure 3.1 and analysed in the following.

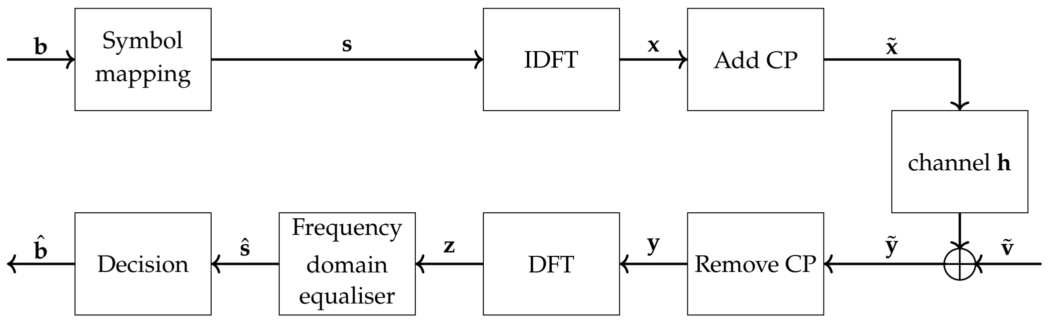


Figure 3.1: Communication chain considered for the OFDM. The vectors $\mathbf{x}$, $\mathbf{v}$ and $\mathbf{y}$ are obtained by removing the Cyclic Prefixes (CP) part of $\tilde{\mathbf{x}}$, $\tilde{\mathbf{v}}$ and $\tilde{\mathbf{y}}$. Note that we have considered the equivalent discrete channel h, which comes from the convolution between the physical channel $c(t)$, the pulse shaping filter and the matched filter.

Denoting by N the number of available subcarriers, the symbols transmitted in the time domain for a given block i (called *OFDM symbol*) are given by

$$x_i[n] = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} s_i[k] e^{j\frac{2\pi}{N}kn}, \qquad n = 0, ..., N-1,$$

where x_i has been computed using the following definition of the Discrete Fourier Transform (DFT) and Inverse Discrete Fourier Transform (IDFT):

$$\begin{aligned} \mathrm{DFT}_{\sqrt{N}}\{x[n]\}[k] &= \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn}, \\ \mathrm{IDFT}_{\sqrt{N}}\{X[k]\}[n] &= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi}{N}kn}. \end{aligned}$$

This formulation (with a $\sqrt{N}$ factor on both sides) enables to keep the same signal energy in both domains (See appendix). Note that N is usually chosen to be a power of 2 to increase the efficiency of the Fast Fourier Transform (FFT) algorithm. However, most FFT implementations (as the one of Labview) consider the following definitions:

$$\begin{aligned}\text{DFT}\{x[n]\}[k] &= \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn},\\ \text{IDFT}\{X[k]\}[n] &= \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{j\frac{2\pi}{N}kn}.\end{aligned}$$

Question 1: What is the subcarrier spacing Δf in continuous time, i.e. the bandwidth dedicated to one specific subcarrier, depending on the number of subcarriers N and the symbol period T?

In order to simplify the equalisation at the receiver, a *cyclic prefix* of length L_c is introduced at the start of each block: the last L_c symbols of each OFDM symbol are copied at the start of the symbol. Therefore, in practice, for N available subcarriers, N symbols s in the frequency domain are mapped to $N + L_c$ symbols $\tilde{x}$ in the time domain:

$$\tilde{x}_i[n] = \frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} s_i[k]e^{j\frac{2\pi}{N}k(n-L_c)}, \qquad n = 0, ..., N + L_c - 1.$$

You can verify that $\tilde{x}_i[n] = \tilde{x}_i[n + N]$, $n = 0, ..., L_c - 1$ and that $\tilde{x}_i[n + L_c] = x_i[n]$, $n = 0, ..., N - 1$. The presence of cyclic prefixes ensure that there is no interference between adjacent OFDM symbols, and it helps to convert the linear convolution into a circular convolution. The length of the cyclic prefix L_c should be the same or exceed the order of the channel response. Therefore, for a L-taps frequency selective channel impulse response,

$$L - 1 \leq L_c < N.$$

Question 2: What are the cyclic prefix duration T_{CP} and the OFDM symbol durations T_{OFDM} depending on the parameters defined previously?

In practice, some subcarriers remain unused. For example, the zero frequency is commonly unused owing to RF distortion at DC. Additionally, subcarriers at the edges of the total bandwidth are also unused to avoid interference with signals in adjacent frequency bands (*guard bands*). In the following, we denote by K the number of unused subcarriers, called null tones.

Question 3: The introduction of cyclic prefixes and null tones has an impact on the achievable rate, i.e. the number of bits transmitted during one second, of the system. Assuming that the input symbols are modulated using a M-QAM constellation, what is the effective rate of the OFDM transmission? Compare it to the rate of a single carrier transmission using the same constellation.

Specific synchronisation techniques have been designed for the OFDM modulation. However, the techniques and the packet structure studied in the previous labs are still applicable with OFDM.

Assuming a perfect synchronisation, after the matched filtering and downsampling, thanks to the introduction of cyclic prefixes, the received signal for the OFDM symbol i is given by

$$\tilde{y}_i[n] = \sum_{l=0}^{L-1} h[l]\tilde{x}_i[n-l] + \tilde{v}_i[n], \qquad n = 0, ..., N + L_c - 1, \tag{3.1}$$

where, h is the frequency selective channel impulse response, L is the number of channel taps, and v is an AWGN. If the cyclic prefix is discarded,

$$\begin{aligned}
y_i[n] &= \tilde{y}_i[n+L_c] \\
&= \sum_{l=0}^{L-1} h[l]\tilde{x}_i[n+L_c-l] + \tilde{v}_i[n+L_c] \\
&= \frac{1}{\sqrt{N}} \sum_{l=0}^{L-1} h[l] \sum_{k=0}^{N-1} s_i[k] e^{j\frac{2\pi}{N}k(n-l)} + v_i[n] \\
&= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \lambda[k] s_i[k] e^{j\frac{2\pi}{N}kn} + v_i[n] \\
&= \mathrm{IDFT}_{\sqrt{N}}\left\{\lambda[k]s_i[k]\right\}[n] + v_i[n],
\end{aligned}$$

where $v_i[n] = \tilde{v}_i[n+L_c]$ and $\lambda[k]$ is the conventional DFT of the zero-padded channel taps:

$$\lambda[k] = \mathrm{DFT}\left\{h[n]\right\}[k] = \sum_{l=0}^{L-1} h[l] e^{-j\frac{2\pi}{N}kl}.$$

Question 4: In the above equations (development of $y_i[n]$), spot where the assumption $L_c > L-1$ is used.

Finally, assuming perfect CSI, the transmitted symbols in the frequency domain are recovered using a DFT and frequency domain equalisation (zero-forcing). Denoting $\nu_i[k] = \mathrm{DFT}_{\sqrt{N}}\left\{v_i[n]\right\}[k]$,

$$z_i[k] = \mathrm{DFT}_{\sqrt{N}}\left\{y_i[n]\right\}[k] = \lambda[k]s_i[k] + \nu_i[k] \quad \overset{\text{FEQ}}{\Rightarrow} \quad \hat{s}_i[k] = \frac{z_i[k]}{\lambda[k]} = s_i[k] + \frac{\nu_i[k]}{\lambda[k]}. \tag{3.2}$$

The above equation shows that the OFDM modulation divides a frequency selective channel into multiple “flat fading"-like subchannels. The transmitted symbols are recovered independently of each other, and each symbol is affected by the attenuation relative to its subcarrier. Therefore, the framework is well suited for adaptive modulation and power control techniques (adapting the modulation and the power for each subcarrier).

Question 5: Based on the DFT of the channel taps, show that, with an OFDM system, the channel is necessarily (strictly) flat if $L = 1$, and (strictly) frequency selective if $L > 1$.

In summary, the important parameters of an OFDM transmission are the following:

- the number of subcarriers N;
- the cyclic prefix length L_c;
- the number of null tones K;
- the passband bandwidth $B \leftrightarrow$ the symbol period T, the relation between these parameters being defined by the pulse shaping filter.

3: Sensitivity to Carrier Frequency Offset

Among the synchronisation impairments, the timing recovery has been handled in the first lab. However, we have always assumed that the carrier frequency of the transmitter and receiver are perfectly synchronised. Small carrier frequency differences lead to a distortion known as *carrier frequency offset*. These differences can be caused by a wrong synchronisation between the oscillators of the transmitter and receiver, or because of the Doppler effect[11] introduced by the paths of the channel. Figure 3.2 illustrates the introduction of the carrier frequency offset f_0 into the complex baseband transmission chain.

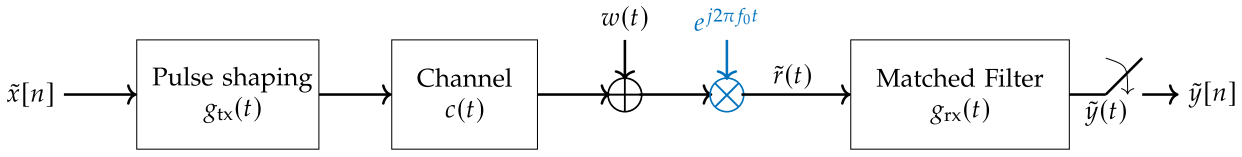


Figure 3.2: Introduction of the carrier frequency offset.

In discrete complex baseband representation, the impact of the carrier frequency offset can be interpreted as a rotation of the received signal:

$$\tilde{y}[n] \approx e^{j2\pi f_0 T n} \sum_{l=0}^{L-1} h[l]\tilde{x}[n-l] + \tilde{v}[n],$$

where $h(t) = (g_{\text{tx}} * c * g_{\text{rx}})(t)$, $h[n] = h(nT)$ and $\tilde{v}[n]$ is an AWGN.

Question 6: *Bonus*: Demonstrate the above equation.
Hint: since the frequency offset is generally small compared to the matched filtering bandwidth, you can assume that $e^{-j2\pi f_0 t} g_{rx}(t) \approx g_{rx}(t)$ (small shift of the spectrum).

The OFDM modulation is especially sensitive to frequency offset. In particular, a frequency offset can be seen as a shift of the signal in the frequency domain, leading to what is called *inter-carrier interference* (ICI). The subcarrier spacing determines the sensitivity to carrier frequency offset. Taking into account the cyclic prefixes, denoting $\epsilon = f_0 T N = f_0/\Delta f$, for the OFDM symbol i,

$$\tilde{y}_i[n] = e^{j2\pi\epsilon\left(1+\frac{L_c}{N}\right)i} e^{j\frac{2\pi}{N}\epsilon n} \sum_{l=0}^{L-1} h[l]\tilde{x}_i[n-l] + \tilde{v}_i[n], \qquad n = 0, \dots, N+L_c-1. \tag{3.3}$$

It remains to analyse how this frequency offset impacts the received symbols $\hat{s}_i[l]$. To that aim, similarly to the developments of (3.1)-(3.2), it can be shown that

$$\begin{aligned} y_i[n] &= \tilde{y}_i[n+L_c], \\ &= \frac{e^{j\frac{2\pi}{N}\epsilon L_c}}{\sqrt{N}} e^{j2\pi\epsilon\left(1+\frac{L_c}{N}\right)i} \sum_{k=0}^{N-1} \lambda[k] s_i[k] e^{j\frac{2\pi}{N}(k+\epsilon)n} + v_i[n], \end{aligned}$$

with $v_i[n] = \tilde{v}_i[n+L_c]$. Then, using a DFT to recover the transmitted symbols in the frequency domain, denoting $\nu_i[k] = \text{DFT}_{\sqrt{N}}\{v_i[n]\}[k]$,

$$\begin{aligned} z_i[l] &= \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} y_i[n] e^{-j\frac{2\pi}{N}ln}, \\ &= \frac{e^{j\frac{2\pi}{N}\epsilon L_c}}{N} e^{j2\pi\epsilon\left(1+\frac{L_c}{N}\right)i} \sum_{k=0}^{N-1} \lambda[k] s_i[k] \sum_{n=0}^{N-1} e^{j\frac{2\pi}{N}(k+\epsilon-l)n} + \nu_i[l]. \end{aligned}$$

[11]In order to consider a multipath channel model with Doppler effect, a time-variant channel model is required. If the path i experiences a Doppler effect with Doppler frequency $f_{\text{D}i}$, Equation (2.1) is extended as

$$h(t,\tau) = \sum_i a_i e^{j\phi_i} e^{j2\pi f_{\text{D}i} t} \delta(\tau - \tau_i).$$

For simplicity, this model will not be introduced in these labs.

Finally,

$$\hat{s}_i[l] = e^{j\frac{2\pi}{N}\epsilon L_c + j\pi\frac{N-1}{N}\epsilon}\, e^{j2\pi\epsilon\left(1+\frac{L_c}{N}\right)i} \left(s_i[l]\,\Gamma_N(0,\epsilon) + \underbrace{\sum_{k\neq l} \frac{\lambda[k]}{\lambda[l]} s_i[k]\,\Gamma_N(k-l,\epsilon)}_{\text{ICI}} \right) + \frac{\nu_i[l]}{\lambda[l]}, \tag{3.4}$$

where the Gamma function is defined as

$$\Gamma_N(n,\epsilon) = \frac{1}{N} e^{-j\pi\frac{n}{N}} \frac{\sin(\pi\epsilon)}{\sin\left(\frac{\pi}{N}(n+\epsilon)\right)},$$

and represented in Figure 3.3.

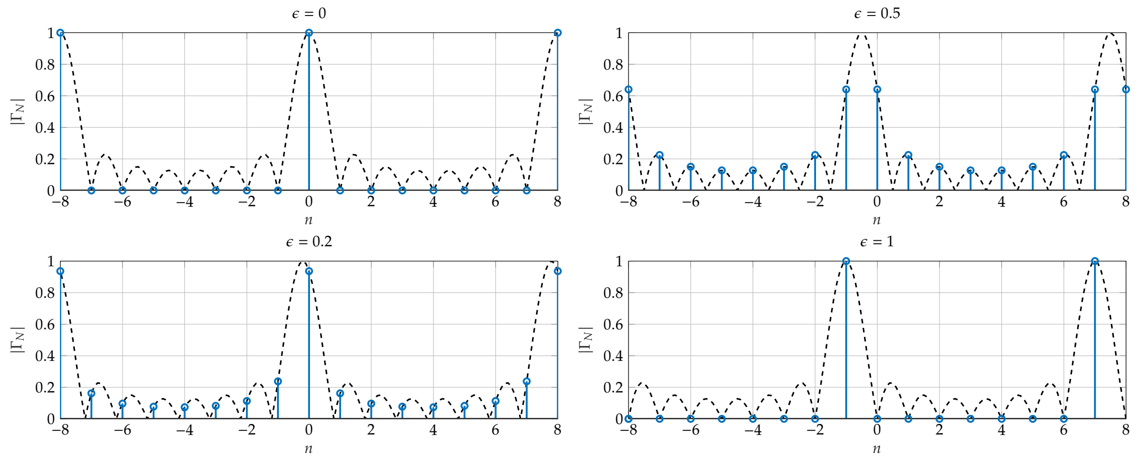


Figure 3.3: Impact of the carrier frequency offset on the frequency sampling, with $N = 8$.

Question 7: Based on (3.4) and Figure 3.3, link the following 4 constellation impairments to their analytical explanation:

- Rotation of the whole constellation;
- Rotation of successive OFDM symbols;
- Inter-Carrier Interference (ICI);
- When ϵ is large, shift of the transmitted symbols.

Question 8: *Bonus*: Starting from (3.3), obtain (3.4) which analyses the impact of a frequency offset on the received symbols.
Hint:

$$\sum_{n=0}^{N-1} e^{j\frac{2\pi}{N}kn} = \frac{1-e^{j2\pi k}}{1-e^{j\frac{2\pi}{N}k}} = e^{j\pi\frac{N-1}{N}k} \frac{\sin(\pi k)}{\sin\left(\frac{\pi}{N}k\right)}.$$

Question 9: Discuss the common points and differences between ISI introduced by a symbol timing error in the case of a single carrier transmission, and ICI introduced by a carrier frequency offset in an OFDM transmission. What is the impact of each impairment on the received signal constellation?

4: Lab Instructions

In this lab, you will implement three blocks:

- the OFDM modulator, described in Table 3.1;
- the OFDM demodulator, described in Table 3.2;
- the frequency domain equaliser which performs the frequency domain equalisation (Equation (3.2)), described in Table 3.3.

Numerous blocks described in Table 3.4 are provided to help you.

Figure 3.4 shows the block diagrams of the modulator and demodulator. In order to process symbols blocks efficiently, the *Serial to Parallel* generates parallel symbol streams from a serial symbol stream (1D array to a 2D array). Each row of the generated 2D array at the output of the block correspond to a single OFDM symbols, and the number of rows corresponds to the number of OFDM symbols. In LabVIEW, you can extract each block (one line of the 2D array) independently thanks to the *auto-indexing* feature of the For loops. All the parameters relative to the OFDM modulation you will need can be found unbundled from the cluster *OFDM Parameters in*. The other parameters relative to the transmission chain are bundled in the cluster *Modulation Parameters In*.

Here are some additional tips for the implementation:

- Be sure to use the *Inverse Complex FFT* block, and not the *Inverse FFT* block.
- When the number of bits is not a multiple of $N - K$, zeros are introduced in the last OFDM symbol. Don't forget to remove these zeros at the end from *Demodulated Symbols* at the output of *student_OFDM_demodulator.gvi*;
- The block *OFDM_FEQ.gviprotected* sets to zero the channels taps in the frequency domain at the indices corresponding to null tones.

Don't forget to connect your blocks into *OFDM_transmitter.gvi* and *OFDM_receiver.gvi*!

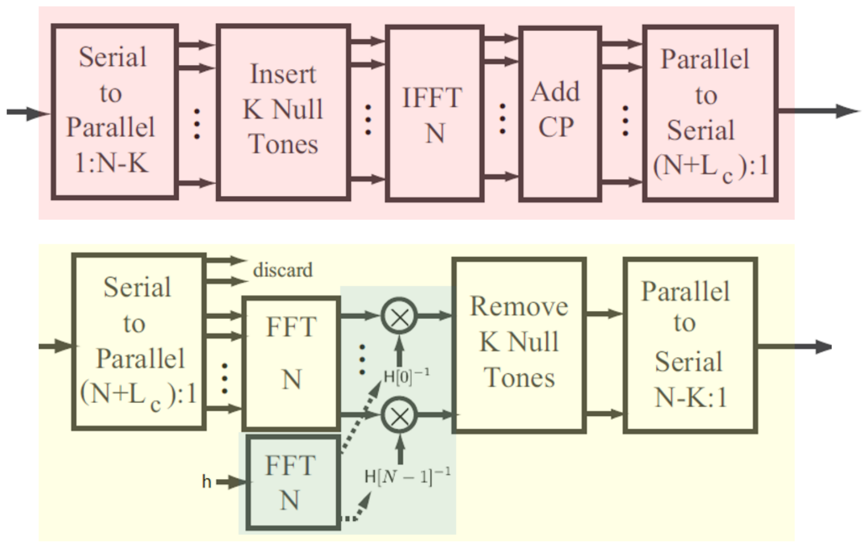


Figure 3.4: OFDM Modulator (red) and Demodulator (yellow) with Frequency domain Equalisation (blue).

student_OFDM_modulator.gvi - implements the OFDM modulator of Figure 3.4.			
Inputs	*OFDM Parameters In*	Cluster	OFDM related parameters.
	Input Symbols	CDB 1D array	Input symbol stream.
	error in (no error)	Error	Input LabVIEW error.
Outputs	*OFDM Parameters Out*	Cluster	OFDM related parameters.
	Output Symbols	CDB 1D array	Output symbol stream.
	error out	Error	Output LabVIEW error.

Table 3.1: OFDM modulator specifications.

student_OFDM_demodulator.gvi - implements the OFDM demodulator of Figure 3.4.			
Inputs	*OFDM Parameters In*	Cluster	OFDM related parameters.
	Modulation Parameters In	Cluster	Transmission chain parameters.
	Received Samples	CDB 1D array	Input sample stream.
	Channel Estimate	CDB 1D array	Estimation of the channel taps.
	Error In (no error)	Error	Input LabVIEW error.
Outputs	*OFDM Parameters Out*	Cluster	OFDM related parameters.
	Modulation Parameters Out	Cluster	Transmission chain parameters.
	Demodulated Symbols	CDB 1D array	Output symbol stream.
	FD Channel Estimate	CDB 1D array	Estimation of the channel taps in the frequency domain.
	Error Out	Error	Output LabVIEW error.

Table 3.2: OFDM demodulator specifications.

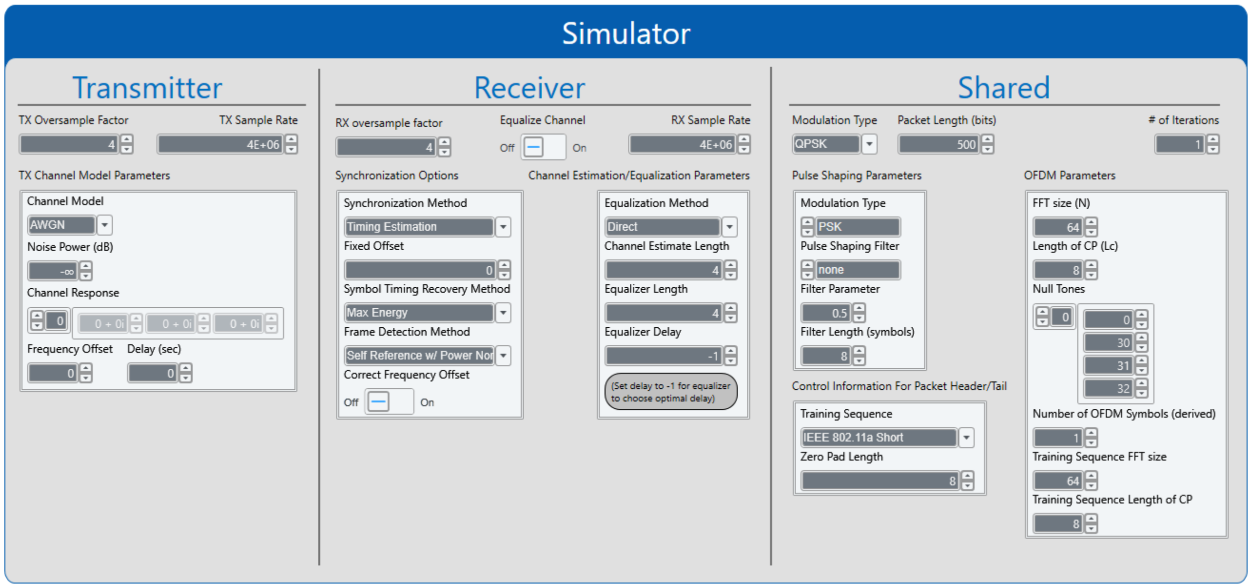
student_OFDM_FEQ.gvi - implements the frequency domain equalisation (Equation (3.2)).			
Inputs	*OFDM Parameters In*	Cluster	OFDM related parameters.
	Modulation Parameters In	Cluster	Transmission chain parameters.
	Input	CDB 2D array	Parallel non equalised input symbol stream.
	Channel Estimate	CDB 1D array	Estimation of the channel taps.
	Error In (no error)	Error	Input LabVIEW error.
Outputs	*OFDM Parameters Out*	Cluster	OFDM related parameters.
	Modulation Parameters Out	Cluster	Transmission chain parameters.
	Equalised Output	CDB 2D array	Parallel equalised output symbol stream.
	FD Channel Estimate	CDB 1D array	Estimation of the channel taps in the frequency domain.
	Error Out	Error	Output LabVIEW error.

Table 3.3: OFDM Frequency domain equaliser specifications.

Name of the VI	Description
S2P.gvi	Converts a serial input stream (1D array) into a parallel output stream (2D array). The number of columns is specified by the parameter *Block Size*.
P2S.gvi	Converts a parallel input stream (2D array) into a serial output stream (1D array).
OFDM_insert_null_tones.gvi	Insert columns of zeros into a 2D input array at indices specified through the 1D array *Null Tones*.
OFDM_remove_null_tones.gvi	Remove columns of a 2D input array at indices specified through the 1D array *Null Tones*.
OFDM_add_CP.gvi	Prepends the last columns of a 2D input array to the start of the array. The number of columns is specified by the parameter *Length of CP (Lc)*.
OFDM_remove_CP.gvi	Remove the first columns of a 2D input array. The number of columns is specified by the parameter *Length of CP (Lc)*.

Table 3.4: Provided VIs.

The default parameters of the simulator are the following:


Simulator

Transmitter

TX Oversample Factor: 4
TX Sample Rate: 4E+06
TX Channel Model Parameters
Channel Model: AWGN
Noise Power (dB): -∞
Channel Response: 0 | 0 + 0i | 0 + 0i | 0 + 0i
Frequency Offset: 0
Delay (sec): 0

Receiver

RX oversample factor: 4
Equalize Channel: Off
RX Sample Rate: 4E+06
Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Max Energy
Frame Detection Method: Self Reference w/ Power Nor
Correct Frequency Offset: Off
Channel Estimation/Equalization Parameters
Equalization Method: Direct
Channel Estimate Length: 4
Equalizer Length: 4
Equalizer Delay: -1
(Set delay to -1 for equalizer to choose optimal delay)

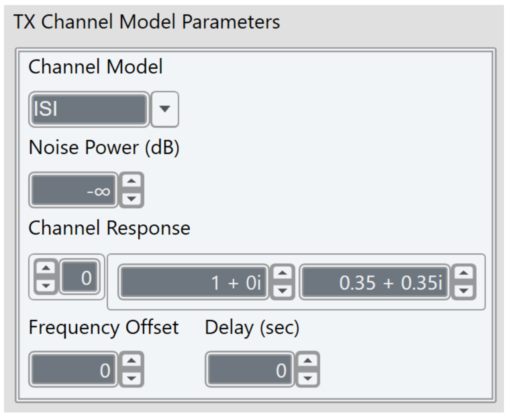
Shared

Modulation Type: QPSK
Packet Length (bits): 500
# of Iterations: 1
Pulse Shaping Parameters
Modulation Type: PSK
Pulse Shaping Filter: none
Filter Parameter: 0.5
Filter Length (symbols): 8
Control Information For Packet Header/Tail
Training Sequence: IEEE 802.11a Short
Zero Pad Length: 8
OFDM Parameters
FFT size (N): 64
Length of CP (Lc): 8
Null Tones: 0 | 0, 30, 31, 32
Number of OFDM Symbols (derived): 1
Training Sequence FFT size: 64
Training Sequence Length of CP: 8

Figure 3.5: Default simulator settings for OFDM.

1. Verify the behaviour of your blocks in the default case. Then, increase the noise power to -20dB and -10dB. If everything is working well, suppress the noise and introduce a 2-tap channel impulse response $\mathbf{c} = [1 \quad 0.35 + 0.35j]$ as shown in Figure 3.6. Don't forget to ensure that the parameter *Channel Model* is set to *ISI*. Verify your implementation of the frequency domain equaliser by setting the parameter *Equalize Channel* to *On*[12].

[12]Don't forget to implement this feature in the OFDM demodulator!


TX Channel Model Parameters

Channel Model: ISI

Noise Power (dB): -∞

Channel Response: 0 | 1 + 0i | 0.35 + 0.35i

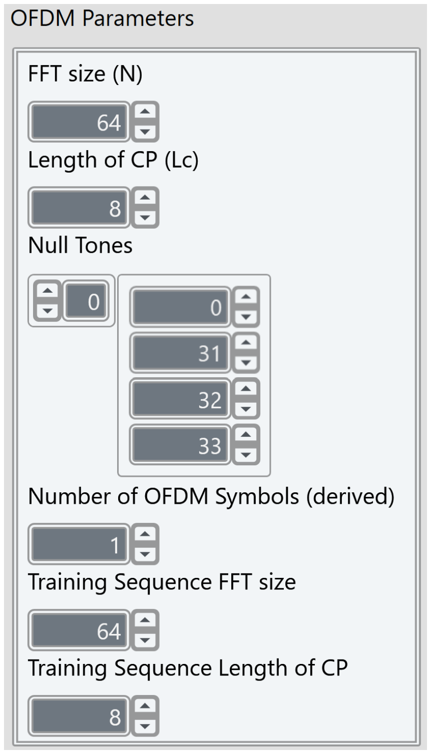
Frequency Offset: 0

Delay (sec): 0

Figure 3.6: Introduction of a frequency selective channel.

Question 10: Deactivate the frequency domain equaliser in order to study the impact of a frequency selective channel with OFDM modulation. What are the differences you can observe on the constellation compared to the results of Lab 2?

2. Let us now consider a frequency offset $f_0 = 200$ Hz with the two sets of parameters shown in Figure 3.7 for the OFDM modulation. The other parameters are set to their default value. Specifically, the channel is set to AWGN. Verify especially that the *Correct Frequency Offset* parameter is deactivated.


OFDM Parameters

FFT size (N): 64

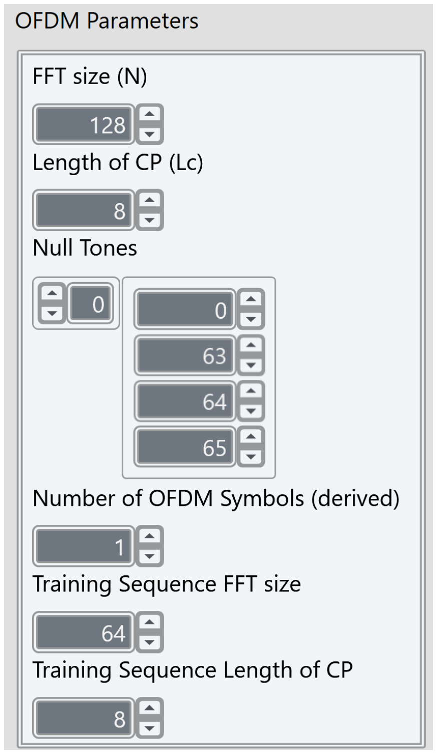
Length of CP (Lc): 8

Null Tones: 0 | 0, 31, 32, 33

Number of OFDM Symbols (derived): 1

Training Sequence FFT size: 64

Training Sequence Length of CP: 8


OFDM Parameters

FFT size (N): 128

Length of CP (Lc): 8

Null Tones: 0 | 0, 63, 64, 65

Number of OFDM Symbols (derived): 1

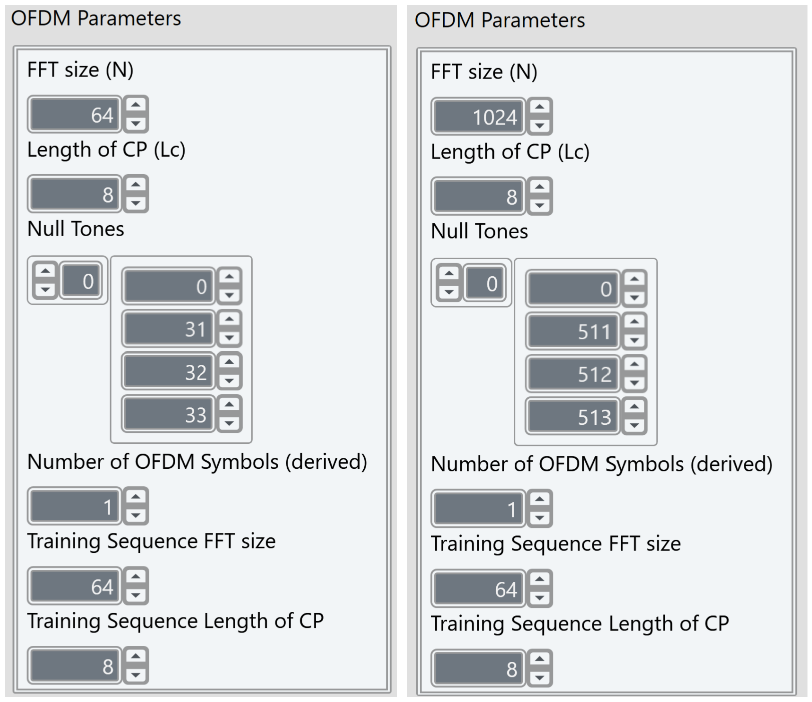
Training Sequence FFT size: 64

Training Sequence Length of CP: 8

Figure 3.7: Considered OFDM systems parameters.

Question 11: Observe the impact of the frequency offset on the received signal constellation. Does it match your analytical developments?

3. Finally, let us consider two different sets of parameters shown in Figure 3.8 for the OFDM modulation. The other parameters are set to their default value. Verify especially that the *Correct Frequency Offset* boolean is set to false. The VI file *OFDM_simulator_frequency_loop.gvi* is provided to you in order to study the impact of the frequency offset on the BER, or to observe dynamically its impact on the constellation.



OFDM Parameters
FFT size (N): 64
Length of CP (Lc): 8
Null Tones: 0 | 0, 31, 32, 33
Number of OFDM Symbols (derived): 1
Training Sequence FFT size: 64
Training Sequence Length of CP: 8

OFDM Parameters
FFT size (N): 1024
Length of CP (Lc): 8
Null Tones: 0 | 0, 511, 512, 513
Number of OFDM Symbols (derived): 1
Training Sequence FFT size: 64
Training Sequence Length of CP: 8

Figure 3.8: Considered OFDM systems parameters.

Question 12: Increase the number of transmitted bits. Evaluate the BER of the transmission for different frequency offset values. Which system is the most sensitive to frequency offset? Why?

5: Appendix - normalised (I)DFT

Let us consider $x[n] = \mathrm{IDFT}_{\sqrt{N}}\{X[k]\}[n]$ and $X[k] = \mathrm{DFT}_{\sqrt{N}}\{x[n]\}[k]$. Using the triangular inequality, we have

$$\sum_{k=0}^{N-1} |X[k]|^2 = \sum_{k=0}^{N-1} \left| \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn} \right|^2 \leq \sum_{k=0}^{N-1} \frac{1}{N} \sum_{n=0}^{N-1} |x[n]|^2 = \sum_{n=0}^{N-1} |x[n]|^2,$$
$$\sum_{n=0}^{N-1} |x[n]|^2 = \sum_{n=0}^{N-1} \left| \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi}{N}kn} \right|^2 \leq \sum_{n=0}^{N-1} \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 = \sum_{k=0}^{N-1} |X[k]|^2.$$

Therefore,

$$\left.\begin{array}{l} \sum_{k=0}^{N-1} |X[k]|^2 \leq \sum_{n=0}^{N-1} |x[n]|^2 \\ \sum_{n=0}^{N-1} |x[n]|^2 \leq \sum_{k=0}^{N-1} |X[k]|^2 \end{array}\right\} \Rightarrow \sum_{k=0}^{N-1} |X[k]|^2 = \sum_{n=0}^{N-1} |x[n]|^2,$$

which shows that both the signal in the time domain and frequency domain have the same energy.

Supplementary Lab: Synchronisation - Frame Synchronisation and Frequency Offset Correction

1: Introduction

In the previous labs, you have mitigated the distortion induced by a multi-path channel through equalisation. However, the equalisation methods you have studied use the knowledge of some pilot symbols (*data-aided equalisation*). Therefore, a prior knowledge about the starting index of the frame is needed. Additionally, a last channel impairment must be corrected: the frequency offset. This impairment is a consequence of a frequency mismatch between the local oscillators at the transmitter and receiver, used for the conversions from baseband to radio-frequencies and vice-versa.

The goal of this lab is to understand the impact of a frame misalignment and frequency offset on the performance of the system and to study how to handle these impairments.

2: Theoretical Background

Frame Synchronisation

Let us consider a flat fading channel with an (unknown) attenuation a and dephasage ϕ. If symbol synchronisation has already been performed, the received signal at the output of the matched filter still suffers of an unknown integer delay d and from an AWGN v:

$$y[n] = ae^{j\phi}x[n-d] + v[n].$$

These received samples are complex Gaussian random variables whose conditional PDF is given by

$$f(y[n]|a,\phi,d,x[n]) = \frac{1}{\pi\sigma_v^2}\ \exp\left(-\frac{\left|y[n]-ae^{j\phi}x[n-d]\right|^2}{\sigma_v^2}\right),$$

and with independent noise realisations, the likelihood function for N samples is given by

$$f(\mathbf{y}|a,\phi,d,\mathbf{x}) = \left(\frac{1}{\pi\sigma_v^2}\right)^N\ \exp\left(-\frac{1}{\sigma_v^2}\sum_{n=0}^{N-1}\left|y[n]-ae^{j\phi}x[n-d]\right|^2\right).$$

The maximum likelihood estimator of a, ϕ and d is obtained by maximising this likelihood function or its associated log-likelihood function. Therefore, at the end, the estimates of the three parameters are obtained by solving the following problem:

$$\begin{aligned}\hat{a},\hat{\phi},\hat{d} &= \arg\min_{a,\phi,d}\ \sum_{n=0}^{N-1}\left|y[n]-ae^{j\phi}x[n-d]\right|^2 \\ &= \arg\min_{a,\phi,d}\ a^2\sum_{n=0}^{N-1}|x[n-d]|^2 - 2a\,\Re\left\{e^{-j\phi}\sum_{n=0}^{N-1}y[n]x^*[n-d]\right\}.\end{aligned}$$

First, one can observe there is only one term depending on the phase. Since $|z| \geq |\Re\{z\}|$ for every complex number z, the estimate of ϕ can already be computed as

$$\hat{\phi} = \arg\left\{\sum_{n=0}^{N-1}y[n]x^*[n-d]\right\},$$

and with this optimal phase, the problem to solve becomes

$$\hat{a},\hat{d} = \arg\min_{a,d}\ a^2\sum_{n=0}^{N-1}|x[n-d]|^2 - 2a\left|\sum_{n=0}^{N-1}y[n]x^*[n-d]\right|.$$

If the number of samples is sufficiently large compared to the length of the received sequence, the first term evaluates the energy of the transmitted sequence, which does not depend on the delay d. In conclusion, if the phase of the received signal is unknown, the estimate of the delay is obtained by computing the magnitude of the correlation between the received signal and the transmitted sequence:

$$\hat{d} = \arg\max_{d} \left| \sum_{n=0}^{N-1} y[n]x^*[n-d] \right|.$$

In practice, as for the previous lab, a preamble is inserted at the start of the frame, composed of a known symbol sequence of length N_t. Therefore, assuming that the transmitted data is different from the training sequence, the correlation of the received sequence with the preamble s is the metric maximised to estimate the delay d:

$$\hat{d} = \arg\max_{d} \left| \sum_{n=0}^{N_t-1} y[n+d]s^*[n] \right|. \tag{C.1}$$

Note that, in a frequency selective channel, the observed peak in the auto-correlation is spread because of the multi-path components of the channel. Therefore, correlation-based frame detection becomes less accurate, and the maximum may shift from some indices. However, this can be handled through equalisation.

In order to obtain the best estimations, specific preamble sequences with powerful properties have been designed, as the Frank sequences, Zadoff-Chu sequences, Gold sequences or Golay sequences. In this Lab, Barker codes will be introduced. These sequences are detailed in Table C.1. They have been used for example in the IEEE 802.11 and IEEE 802.11b protocols.

Length	Barker sequences
2	[01],[00]
3	[001]
4	[0010],[0001]
5	[00010]
7	[0001101]
11	[00011101101]
13	[0000011001010]

Table C.1: Barker Sequences (with a BPSK mapping, $0 \rightarrow 1$ and $1 \rightarrow -1$). The complete set of known Barker sequences is the closure of this set under reversal and negation.

The Barker sequences benefit from a good aperiodic autocorrelation property. Additionally, it has also a good rank property for the channel estimation (usually, training sequences with strong autocorrelation properties can also provide sufficient rank properties). A Barker sequence of length N_t is a sequence of values ± 1 such that

$$\left| \sum_{i=0}^{N_t-k-1} s[i]s[i+k] \right| \leq 1 \quad \forall k \neq 0. \tag{C.2}$$

The autocorrelations of some Barker codes are illustrated in Figure C.1.

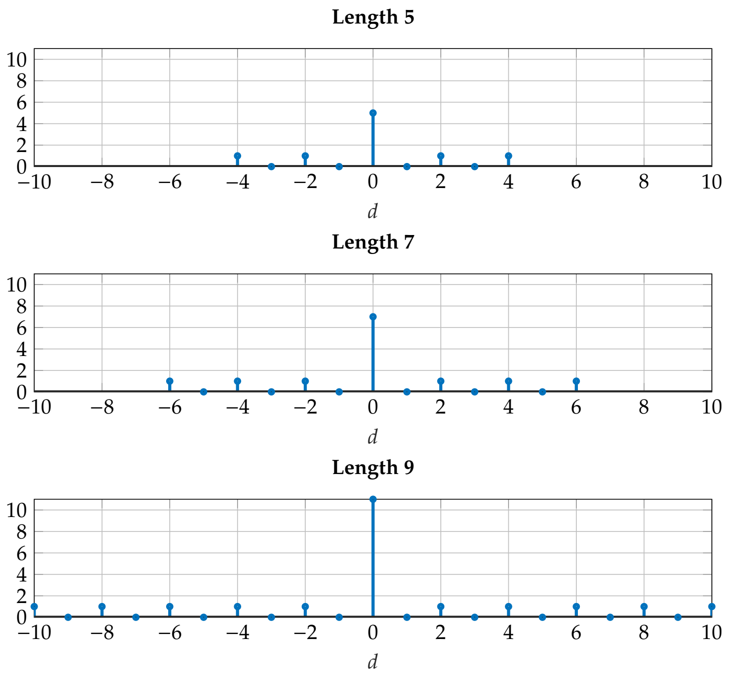


Figure C.1: Absolute value of the autocorrelation for Barker sequences of length 5, 7 and 11.

Note that the sequences provided in Table C.1 are limited in length (maximum 13). To obtain longer training sequences, multiple Barker sequences can be concatenated together. This will generate a higher main peak, but also multiple smaller peaks. However, for frequency offset estimation, the concatenation of multiple copies of the same training sequence is useful, as it will be described after. Another approach is to relax Equation (C.2), allowing sequences with higher cross-correlation values, or delete the binary restriction and use complex values, as it is done in other families of training sequences.

Frequency Offset Estimation and Correction

As presented in Lab 3, small carrier frequency differences lead to a distortion known as *carrier frequency offset*. These differences can be caused by a wrong synchronisation between the oscillators of the transmitter and receiver, or because of the Doppler effect[13] affecting the paths of the channel. Figure C.2 illustrates the introduction of the carrier frequency offset f_0 into the complex baseband transmission chain.

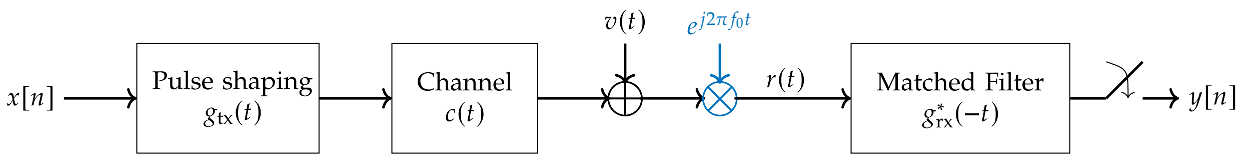

$x[n]$
Pulse shaping $g_{\text{tx}}(t)$
Channel $c(t)$
$v(t)$
$e^{j2\pi f_0 t}$
$r(t)$
Matched Filter $g^*_{\text{rx}}(-t)$
$y[n]$

Figure C.2: Introduction of the carrier frequency offset.

In discrete complex baseband representation, the impact of the carrier frequency offset can be interpreted

[13]In order to consider a multipath channel model with Doppler effect, a time-variant channel model is required. If the path i experiences a Doppler effect with Doppler frequency $f_{\text{D}i}$, the channel model in complex baseband representation is extended as

$$c(t,\tau) = \sum_i \gamma_i e^{j\phi_i} e^{j2\pi f_{\text{D}i} t}\, \delta(\tau - \tau_i).$$

For sake of simplicity, this model will not be introduced in these labs.

as a rotation of the received signal:

$$y[n] \approx e^{j2\pi f_0 T n} \sum_m h[m] x[n-m] + v[n], \tag{C.3}$$

where $h(t) = g_{\text{tx}}(t) * c(t) * g_{\text{rx}}(t)$, $h[n] = h(nT)$ and $w[n]$ is an AWGN.

Question 1: *Bonus*: Demonstrate Equation (C.3).
Hint: since the frequency offset is generally small compared to the matched filtering bandwidth, you can assume that $e^{j2\pi f_0 t} g_{rx}(t) \approx g_{rx}(t)$ (small shift of the spectrum).

In order to estimate f_0, Moose proposed a data-aided approach based on periodic training sequences. Let us consider a 2-periodic preamble sequence s. Denoting $N_p = N_t/2$,

$$s[n+N_p] = s[n] \quad \forall n = 0, ..., N_p - 1.$$

With a frequency selective channel of length L, the $L-1$ first elements of the sequence are corrupted by previous samples (noise realisations). Therefore, for $L \leq n \leq N_p - 1$,

$$\begin{aligned} y[n+N_p] &= e^{j2\pi f_0 T (n+N_p)} \sum_m h[m] s[n+N_p-m] + v[n+N_p] \\ &= e^{j2\pi f_0 T N_p}\, e^{j2\pi f_0 T n} \sum_m h[m] s[n-m] + v[n+N_p] \\ &\approx e^{j2\pi f_0 T N_p}\, y[n]. \end{aligned} \tag{C.4}$$

Note that this does not depend on the channel coefficients which are unknown!

In order to estimate f_0 using this property, a modified least square problem can be formulated:

$$\hat{\alpha} = a\, e^{j\pi \hat{f}_0 T N_p} = \arg\min_{\alpha} \sum_{l=L}^{N_p-1} \left| y[l+N_p] - \alpha y[l] \right|^2.$$

The solution of this problem is given by

$$\hat{\alpha} = \frac{\sum_{l=L}^{N_p-1} y[l+N_p] y^*[l]}{\sum_{l=L}^{N_p-1} |y[l]|^2}.$$

and therefore the estimate of the frequency offset is obtained:

$$\hat{f}_0 = \frac{\arg\{\hat{\alpha}\}}{\pi T N_t} = \frac{\arg\left\{\sum_{l=L}^{N_p-1} y[l+N_p] y^*[l]\right\}}{\pi T N_t}.$$

Owing to the periodicity of the complex exponential, the Moose algorithm can estimate frequency offsets included in the following range:

$$|f_0| \leq \frac{1}{TN_t}.$$

In order to improve the estimation, a larger N_t is preferable to increase noise averaging, but it reduces the range of non ambiguous frequency offset estimations. A solution to this issue is based on multiple repetitions of short training sequences in the preamble, but this will not be considered in this Lab.

Finally, the frequency offset correction is simply performed as follows:

$$\tilde{y}[n] = e^{-j2\pi f_0 T n}\, y[n].$$

Frame Synchronisation with Frequency Offset

It was assumed previously that there was no frequency offset. However, the presence of a large frequency offsets can strongly affect the accuracy of correlation based frame detection algorithms. The auto-correlation properties of the training sequence are affected by the rotation of the received sequence. Fortunately, the periodic preamble sequence property obtained in Equation (C.4) can also be helpful to overcome this issue: instead of comparing a sequence suffering of frequency offset with the initial training sequence, the corrupted sequence can be compared to a sequence which also rotates at the same frequency! Since the frequency offset is unknown at this stage, the corrupted sequence can be correlated with itself (*self-referenced algorithm*). Based on Equation (C.4), it also follows that

$$y[n+N_p]y[n]^* \approx e^{j2\pi f_0 T N_p}|y[n]|^2.$$

Since the two sequences rotate at the same frequency, the same phase shift is experienced for every index n.

Formally, the metric $M[n]$ introduced by Schmidl & Cox is defined as

$$M[n] = \frac{\sum_{l=L}^{N_p-1} y[l+n+N_p]y^*[l+n]}{\sqrt{\sum_{l=L}^{N_p-1} \left|y[l+n+N_p]\right|^2}\sqrt{\sum_{l=L}^{N_p-1} |y[l+n]|^2}},$$

where the denominator is present for normalisation, and to avoid peaks at the start and the end of the transmitted data frame. Then, the start of the frame is estimated as

$$\hat{d} = \arg\max_n \; |M[n]|^2 = \arg\max_n \; \frac{\left|\sum_{l=L}^{N_p-1} y[l+n+N_p]y^*[l+n]\right|^2}{\sum_{l=L}^{N_p-1} \left|y[l+n+N_p]\right|^2 \sum_{l=L}^{N_p-1} |y[l+n]|^2},$$

and the frequency offset is estimated as

$$\hat{f}_0 = \frac{\arg\left\{M[\hat{d}]\right\}}{\pi T N_t} = \frac{\arg\left\{\sum_{l=L}^{N_p-1} y[l+\hat{d}+N_p]y^*[l+n]\right\}}{\pi T N_t}.$$

3: Lab Instructions

In this Lab, you will implement two blocks:

- the sliding correlator for the frame synchronisation, described in Table C.2;
- the Moose estimation and frequency offset correction described in Table C.3.

The sliding correlator should be inserted in *simulator.gvi→top_rx.gvi→receiver.gvi→synchronize.gvi→frame_detect.gvi*. The second block is inserted in the sliding correlator block.

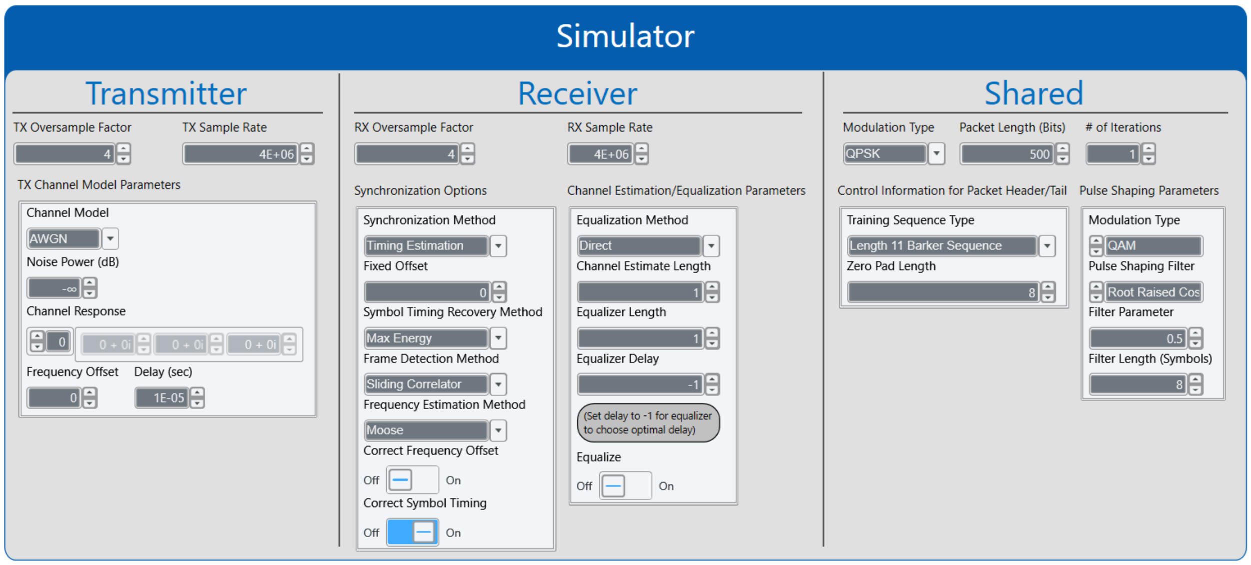
student_sliding_correlator.gvi - implements a correlation-based frame detector. The start of the frame is estimated following Equation (C.1).			
Inputs	*Input Complex Waveform*	Cluster	Cluster containing the input waveform.
	Modulation Parameters	Cluster	Modulation parameters in input.
	Error In	Error cluster	Contains information regarding a possible error in the previous blocks.
Outputs	*Output*	CDB 1D array	Sequence of symbols after frame synchronisation and frequency offset correction.
	Frame offset	I32	Estimated delay before the start of the sequence (in symbols).
	Frequency offset	DBL	Estimated frequency offset, obtained from the *student_moose.gvi* block.
	Error Out	Error cluster	Contains information regarding a possible error in the previous blocks or in this block.

Table C.2: Inputs and outputs of the block *student_sliding_correlator.gvi*.

student_moose.gvi - implements a frequency offset estimation and detection, following the Moose algorithm.			
Inputs	*Input Complex Waveform*	Cluster	Cluster containing the input waveform.
	Modulation Parameters	Cluster	Modulation parameters in input.
	Error In	Error cluster	Contains information regarding a possible error in the previous blocks. This can be directly connected to *Error out*.
Outputs	*Output*	CDB 1D array	Sequence of symbols after frame synchronisation.
	Frequency offset	DBL	Estimated frequency offset.
	Error Out	Error cluster	Contains information regarding a possible error in the previous blocks or in this block. This can be directly connected to *Error in*.

Table C.3: Inputs and outputs of the block *student_moose.gvi*.

The default simulator parameters are shown in Figure C.3. A delay of $10T$ has been introduced.



Simulator

Transmitter

TX Oversample Factor: 4
TX Sample Rate: 4E+06
TX Channel Model Parameters
Channel Model: AWGN
Noise Power (dB): -∞
Channel Response: 0 | 0 + 0i | 0 + 0i | 0 + 0i
Frequency Offset: 0
Delay (sec): 1E-05

Receiver

RX Oversample Factor: 4
RX Sample Rate: 4E+06
Synchronization Options
Synchronization Method: Timing Estimation
Fixed Offset: 0
Symbol Timing Recovery Method: Max Energy
Frame Detection Method: Sliding Correlator
Frequency Estimation Method: Moose
Correct Frequency Offset: Off On
Correct Symbol Timing: Off On
Channel Estimation/Equalization Parameters
Equalization Method: Direct
Channel Estimate Length: 1
Equalizer Length: 1
Equalizer Delay: -1
(Set delay to -1 for equalizer to choose optimal delay)
Equalize: Off On

Shared

Modulation Type: QPSK
Packet Length (Bits): 500
# of Iterations: 1
Control Information for Packet Header/Tail
Training Sequence Type: Length 11 Barker Sequence
Zero Pad Length: 8
Pulse Shaping Parameters
Modulation Type: QAM
Pulse Shaping Filter: Root Raised Cos
Filter Parameter: 0.5
Filter Length (Symbols): 8

Figure C.3: Default simulator parameters.

1. Verify the implementation of your sliding correlator based on the default configuration of the simulator.

Question 2: Explain what happens to the BER if $\hat{d} \neq d$. Why?

---

2. Now, introduce a frequency offset of 100Hz and activate the *Correct Frequency Offset* button. Verify the implementation of the Moose algorithm and the frequency offset correction. If it works, deactivate the *Correct Frequency Offset* button.

Question 3: What is the impact of a frequency offset on the constellation?

Question 4: What is the impact of the frequency offset on the correlation of the training sequence with the received preamble? Plot the cross-correlation function for different frequency offset (i.e. 0Hz, 5kHz, 10kHz and 15kHz) in a zero noise case.

---

3. Let us define the timing and frequency error statistics as

$$\epsilon_\tau = \mathbb{E}\left[\left|\frac{\hat{\tau} - \tau}{T}\right|^2\right], \qquad \epsilon_f = \mathbb{E}\left[\left|(\hat{f}_0 - f_0)T\right|^2\right],$$

where the expectation is taken on the different iterations.

Question 5: Plot a graph of the timing and frequency error statistics against a frequency offset, with both frame synchronisation methods (correlation-based and self-referenced). What can you observe? Do it fit with your expectations?
NB: timing errors also occur when the symbol timing recovery fails. However, in that case, the timing error statistics cannot be larger than 1.

Question 6: Plot a graph of the timing and frequency error statistics against SNR, with a frequency offset of 0Hz and 10kHz. What can you observe?

Question 7: Is it also critical that the periodic preamble sequence has good autocorrelation properties? Why?

Question 8: *Bonus* - What are typical values of the frequency offset you can suffer from with USRPs? Compute an average value and a standard deviation.

Question 9: *Bonus* - Evaluate the impact on the metrics of both frame detection methods of a frequency selective channel.
