MICAS 913 Project
End-to-end Deep Learning of a Communication System

Mansoor Yousefi
Télécom Paris, Palaiseau, France

Abstract

This is a guide for a simulation project on end-to-end deep learning of a communication system. The project consists of implementing a simple point-to-point communication system, a generative deep neural network representing the channel, and an auto-encoder for the end-to-end learning of the transmitter (TX) and receiver (RX). The bit error rate (BER) is plotted as a function of the signal-to-noise ratio for several system parameters, and compared with the BER of a TX and RX from the literature.

Contents

1 Introduction

We implement end-to-end deep learning of a point-to-point communication system.
The project consists of two parts. In the first part in Section 2, we prescribe the data generation model for producing data sets. This part consists of simulating a basic transmitter (TX), a receiver (RX) and a generative deep neural network representing the channel. In the second part in Section 3, we will learn the end-to-end channel (TX, RX and the physical channel) using deep learning.
You should prepare a report in which you will present the plots on the performance and complexity of the deep learning system, and answer the questions in this guide. The software is best written in Python, however MATLAB (or GNU Octave) or Julia are also accepted. MATLAB is used for demonstration in this guide due to its compact syntax. The report is written in LaTeX.
The instructor will provide hands-on help in the class, however, the project must be completed mostly at home. The part on communications is explained in detail in this guide together with pseudo code, to help students who lack background in communications, and to focus on the deep learning part.

2 Generative model

2.1 Block diagram of a point-to-point communication system

Layered architecture The *separation theorem* for point-to-point data communication implies that the source and channel can be designed separately, suggesting a layered architecture [1]. The data (*e.g.*, a camera image) is initially in the analogue domain. The analogue signal is converted to a sequence of bits by the source encoder (consisting of sampling, quantization and data compression). This digital representation has a number desirable advantages [1, Chap. 4.1.1]. After applying digital signal processing, the bits sequence is transformed back to a continuous-time signal by the channel encoder.
The channel encoder consists of channel coding, modulation (bits-to-symbols and symbols-to-signal mapping) and baseband-to-passband shift. The inverse of these layers is the channel decoder. In this lab, we only implement the part from the input of the channel encoder to the output of the channel decoder. Furthermore, we ignore channel error-correction coding and consider a baseband model. The simplified diagram is shown in Fig. 1
The foundations of the digital communication are reviewed in [1–3].

Setting up a simulation environment. Create a simulation environment as follows.

a. SSH into the school server, issuing `ssh username@ssh.enst.fr`

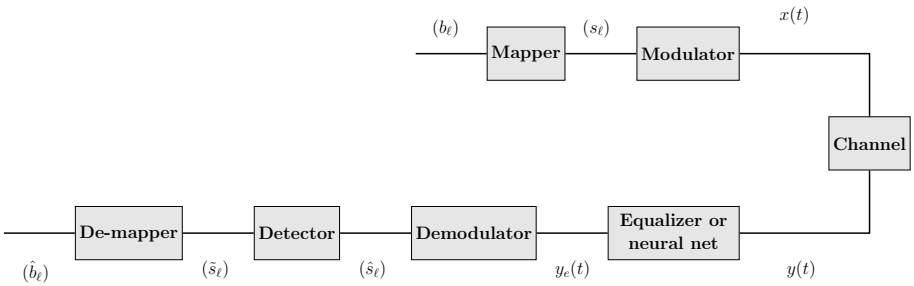


Fig. 1: Block diagram of a simple communication system.

b. Create a folder "micas913-project-name" with sub-folders "generative" and "learning" in your home directory, where "name" is the last name of the group representative. You will save all files here.

c. Create the following Matlab files (m-files): parameters.m, main.m, source.m, mod.m, channel.m, nnet_gen.m, demod.m, detector.m and ber.m.

   As noted, Python is preferred, in which case the above functions appear in a module named, *e.g.*, `generative.py`. This module contains all functions required to produce the data set. The functions on the deep learning will appear in a separate module named, *e.g.*, `predictive.py`. The deep learning mini-library is implemented in `algorithms.py` module.

2.2 Channel model.

Linear channels. Communication media are frequently modeled by the additive white Gaussian noise (AWGN) channel [1–3]

$$y(t) = h(t) * x(t) + n(t),$$

where $x(t)$ is input, $y(t)$ is output, $h(t)$ is channel impulse response, $n(t)$ is white circular symmetric Gaussian noise, and $*$ denotes convolution. All signals are functions from $\mathbb{R}$ to $\mathbb{C}$. The channel filter $h(t)$ introduces a distortion called inter-symbol interference (ISI) that must be cancelled via equalization.
The communication theory of the AWGN channels, such as the design of the optimal receiver, is well developed. As a result, we consider the optical fiber channel for which the model is rather intractable. Application of deep learning may help find good TX and RX for this channel.

The stochastic NLS equation. Let $Q(\tau, \ell) : \mathbb{R} \times \mathbb{R}^+ \mapsto \mathbb{C}$ be the complex envelope of the propagating signal as a function of time τ and distance ℓ. Pulse propagation in optical fiber is modeled by the stochastic nonlinear Schrödinger

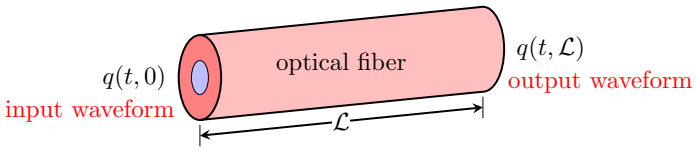


Fig. 2: Fiber channel.

(NLS) equation [4, Eq. 7]

$$\frac{\partial Q(\tau,\ell)}{\partial \ell} = -\underbrace{\frac{j\beta_2}{2}\frac{\partial^2 Q(\tau,\ell)}{\partial \tau^2}}_{\text{dispersion}} + \underbrace{j\gamma|Q(\tau,\ell)|^2 Q(\tau,\ell)}_{\text{nonlinearity}} + \underbrace{N(\tau,\ell)}_{\text{noise}}, \tag{1}$$

where β_2 is the dispersion coefficient, γ is the nonlinearity parameter and $j = \sqrt{-1}$. Further, $N(\tau,\ell)$ is bandlimited circular symmetric complex Gaussian noise with auto-correlation

$$\mathsf{E}\left\{N(\tau,\ell)N^*(\tau',\ell')\right\} = \sigma_0^2 \delta_B(\tau-\tau')\delta(\ell-\ell'),$$

where $\delta(x)$ is the Dirac Delta function, $\delta_B(x) = B\,\mathrm{sinc}(Bx)$, $\mathrm{sinc}(x) = \sin(\pi x)/(\pi x)$, B is the noise bandwidth, σ_0^2 is the in-band noise power spectral density (PSD) and E is the expected value. The transmitter is located at $\ell = 0$ and the receiver at $\ell = \mathcal{L}$.
The stochastic NLS equation (1) models the chromatic dispersion (responsible for temporal broadening), Kerr nonlinearity (responsible for spectral broadening) and noise. It is assumed that the fiber loss is perfectly compensated by distributed amplification.

Constraints. The transmitter is subject to the average power constraint

$$\lim_{T\to\infty} \frac{1}{T}\mathsf{E}\left\{\int_{-\frac{T}{2}}^{\frac{T}{2}} |Q(\tau,0)|^2 \mathrm{d}\tau\right\} \leqslant \mathcal{P}. \tag{2}$$

It is assumed that $Q(\tau,\ell)$ is bandlimited to B Hz for all ℓ, $0 \leqslant \ell \leqslant \mathcal{L}$.

Question 1. The variables Q, τ and ℓ are measured in $\sqrt{\text{Watt}}$ ($\sqrt{W}$), second (s) and meter (or kilometer km), respectively. Find out the units of β_2, γ and σ_0^2. □

Normalized NLS equation. We will find it convenient to normalize the NLS equation.

Question 2. Consider the change of variables

$$q = \frac{Q}{\sqrt{P_0}}, \qquad t = \frac{\tau}{T_0}, \qquad z = \frac{l}{\mathcal{L}}.$$

Show that if we choose

$$P_0 = \frac{2}{\gamma \mathcal{L}}, \qquad T_0 = \sqrt{\frac{|\beta_2|\mathcal{L}}{2}}, \qquad L = \mathcal{L}, \tag{3}$$

we obtain from (1)

$$j\frac{\partial q(t,z)}{\partial z} = \frac{\partial^2 q(t,z)}{\partial t^2} + 2|q(t,z)|^2 q(t,z) + n(t,z), \tag{4}$$

where $0 \leqslant z \leqslant 1$. Here, $n(t,z)$ is circular symmetric complex Gaussian noise with auto-correlation

$$\mathsf{E}\left\{n(t,z)n^*(t',z')\right\} = \sigma^2 \delta_{B_n}(t-t')\delta(z-z'), \tag{5}$$

in which

$$\boxed{\sigma^2 = \frac{\sigma_0^2 \mathcal{L}}{P_0 T_0}}, \tag{6}$$

and $B_n = BT_0$ is the normalized bandwidth. □

In what follows, we consider the normalized NLS equation (4) with noise PSD (6). The normalized model (4) contains no parameter other than σ^2 in (6) which is computed in the following question.

Signal and system parameters We consider a fiber with parameters in Tab. 1, $\mathcal{L} = 1000$ km and $B = 1$ to 10 GHz. We assume a quadrature amplitude modulation (QAM) constellation $\mathcal{C}$ with $M = 16$ points; see Fig. 3(b).

Question 3. Enter the constants in Tab. 1 in `parameters.m`
Convert the dispersion parameter D with unit ps/(nm − km) to dispersion coefficient β_2, using $\beta_2 = -\frac{\lambda_0^2}{2\pi c} D$, where $c = 3 \times 10^8$ m/s is the speed of light. Convert the loss coefficient α from unit dB/km to m^{-1}, using $\alpha = 10^{-4}\log(10) a_{\mathrm{dB}}$ in m^{-1}; see Appendix A.
Calculate the scale parameters in (3).
Calculate the noise PSD $\sigma_0^2 = n_{sp} h \alpha f_0$, where $f_0 = c/\lambda_0$ is the carrier frequency, and its normalized value σ^2 (6).
Enter the constants, scale factors, physical and normalized variables in `parameters.m` as follows.

```
% variables
L = ...       % distance
B = ...       % bandwidth

% physical constants
a_dB = ...    % power loss in dB
```

```
D = ...       % dispersion ps/(nm-km)
gama = ...    % nonlinearity coefficient
nsp = ...     % a constant factor
h = ...       % Planck constant
lambda0 = ... % center wavelength
f0 = ...      % center frequency

alpha =       % loss coefficient
beta2 = ...   % dispersion coefficient

% scale factors
L0 = ...
T0 = ...
P0 = ...

% noise PSD
sigma02 =     % physical
sigma2 =      % normalized
```

□

Question 4. Consider the peak power $P_p = 6$ mW. Obtain the normalize peak power and plot the constellation $\mathcal{C}$.

2.2.1 Transmitter

Binary source. A discrete source is a discrete-time stochastic process.

Question 5. Consider a Bernoulli stochastic process, *i.e.*, a sequence

$$b^N = (b_0, b_1, \cdots, b_{N-1}),$$

where b_i are random variables drawn independently from a Bernoulli probability distribution function (PDF)

$$p(b_i) = \begin{cases} p, & b_i = 0, \\ 1-p, & b_i = 1. \end{cases}$$

Write an m-function `b = source(N, p)` that generates b^N, and test it with $N = 1024$ and $p = 1/2$.

□

Table 1: Fiber and Noise Parameters

a_{dB}	0.2 dB per km	fiber loss
D	17 ps/(nm-km)	chromatic dispersion
γ	1.27 $\text{W}^{-1}\text{km}^{-1}$	nonlinearity parameter
n_{sp}	1	spontaneous emission factor
h	$6.626 \times 10^{-34}\text{J} \cdot \text{s}$	Planck's constant
λ_0	1.55 μm	carrier wavelength

Modulation. The modulation consists of bits-to-symbols and symbols-to-signal mapping.

Question 6. *Bits-to-symbols mapping.* There are M symbol points in the constellation. Map each symbol point to a binary sequence of length $\log M$. Can you present a mapping such that the Hamming distance between the binary sub-sequences associated with the neighbor symbol points is 1? That is the Gray mapping.
Divide b^N into $n = N/\log M$ sub-sequences, each of which with $\log(M)$ bits. Map each sub-sequence into a symbol in the constellation and obtain a sequence of complex numbers $s^n = (s_0, s_1, \cdots, s_{n-1})$, where $s_i \in \mathcal{C}$ are *symbols*.
Write an m-function `s = bit_to_symb(b, cnt)` where cnt is the constellation (vector or matrix of all symbol points).

□

It can be shown that if $p = 1/2$ then s^n is a sequence of complex random variables drawn i.i.d. from the uniform distribution over $\mathcal{C}$.
Symbols-to-signal mapping. Linear modulation consists of expanding a signal $q(t, 0)$ onto an orthonormal basis. The Nyquist-Shannon sampling theorem applied to the space of functions bandlimited to B Hz provides a suitable representation

$$q(t,0) = \sqrt{B} \sum_{\ell=\ell_1}^{\ell_2} s_\ell \operatorname{sinc}(Bt - \ell), \tag{7}$$

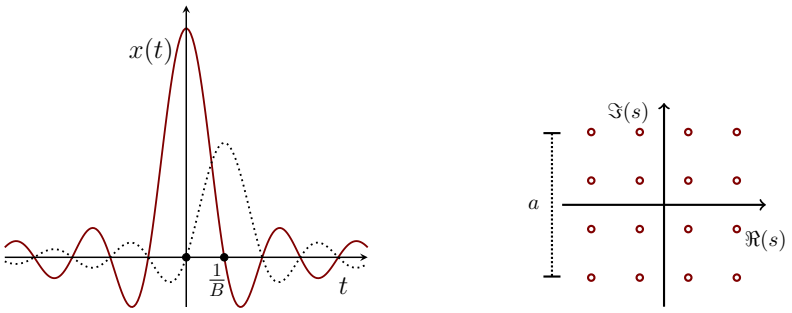
where $\ell_1 = -\lfloor n/2 \rfloor$ and $\ell_2 = \lceil n/2 \rceil - 1$.

Question 7. We write one m-function main.m for each section, in which you set up and run the project. In this section, you will gradually complete the first main.m file.
Complete the following main.m file.

```
% time mesh
T  = 400; % you have to choose this
N  = 2^10; % you have to choose this
dt = T/N;
t  = ... % should be from -T/2 to T/2 with length N

% frequency mesh
F  = 1/dt;
df = 1/T;
f  = ... % should be from -F/2 to F/2 with length N

% bits to signal
M = 16; % size of the constellation
n = 3; % number of symbols (or sinc functions); test with s=1
nb = ... ; % number of bits
p =1/2; % probability of zero
b = source(nb, p); % Bernoulli source, random bits sequence
s = bit_to_symb(b, cnt); % symbol sequence; could be inside modulator.m
```



(a) (b)

Fig. 3: (a) Modulation using sinc functions; (b) a 16-QAM constellation.

```
q0 = mod(t, s, B); % transmitted signal
```

Write an m-function for modulator `mod(t, s, B)`, implementing (7).

```
function q0t = mod(t, s, B)

Ns = len(s)  % number of symbols
l1 = ...
l2 = ...

q0t = ...

end
```

Plot $q(t, 0)$ in time and in frequency. Make sure that the basis elements have unit energy: $\|\mathrm{sinc}(x)\|^2_{L^2(\mathbb{R})} = \left\|\sqrt{B}\,\mathrm{sinc}(Bx)\right\|^2_{L^2(\mathbb{R})} = 1$.

□

Remark 1. The code in this guide should be viewed as pseudo-code. You have to complete the code, add missing variables, arguments, etc. You may have to change the parameters as well in order to obtain meaningful results, especially in places where this is indicated. □

Question 8. The time interval between two consecutive sinc functions is $1/B$. Thus, the symbol rate is $R_s = 1/B$ symbols/s. Express the bit rate R_b in bits/s in terms of R_s.

□

2.2.2 Channel model

Continuous-time model. Consider first the NLS equation (4) in the absence of nonlinearity

$$j\partial_z q(t, z) = \partial_{tt} q(t, z) + n(t, z). \tag{8}$$

Let us simplify the partial differential equation channel (8) to a more familiar form.

Question 9. We solved (8) with $n(t,z) = 0$ in the class. Define the Fourier transform with respect to t with convention:

$$\hat{q}(\omega, z) = \mathcal{F}(q)(\omega) = \int_{-\infty}^{\infty} q(t,z) e^{-j\omega t} \mathrm{d}t. \tag{9}$$

Solve (8) in the presence of noise and show that (8) is reduced to the AWGN channel

$$\hat{q}(\omega, z) = \hat{h}(\omega, z)\hat{q}(\omega, 0) + \hat{z}(\omega), \tag{10}$$

where $\hat{q}(\omega, z)$ is the Fourier transform of $q(t,z)$, and the channel transfer function is

$$\hat{h}(\omega, z) = e^{j\omega^2 z}.$$

Characterize the stochastic process $\hat{z}(\omega)$, *i.e.*, determine the PDF of $\hat{z}(\omega)$, its mean $\mathsf{E}\hat{z}(\omega)$ and the correlation function $\mathsf{E}(\hat{z}(\omega)\hat{z}^*(\omega'))$.

□

In the time domain we have

$$q(t,z) = h(t,z) * q(t,0) + z(t), \tag{11}$$

where $*$ is convolution. Determine the channel impulse response $h(t,z)$. Determine the PDF of $z(t)$, and its mean and correlation.

Discrete-time model. The continuous-time model (8), (11) or (10) should be discretized to a discrete-time model for simulation. To formulate a discrete problem, we need to discretize time and space, thereby the channel and the constraints.
The signal $q(t,z)$ can be sampled at the Nyquist rate $1/B$ to obtain a vector

$$\mathbf{q}(z) = \begin{pmatrix} q_1(z) & q_2(z) & \dots & q_N(z) \end{pmatrix}.$$

However, to discretize the integral that underlies the convolution in (11) (or the time derivative in (8)), we sample the signal at a sufficiently small simulation step size dt, which can be much less than the Nyquist rate.

Question 10. Discretize the signal power (2), by discretizing the integral into a Riemann sum, proving that, as $N \to \infty$:

$$\frac{1}{N} \sum_{i=1}^{N} \mathsf{E}|q_i(0)|^2 \leqslant \mathcal{P}. \tag{12}$$

□

Question 11. The channel can be discretized in the frequency domain by sampling (10) on a frequency mesh obtained from the time mesh.
Discretize the Fourier integrals

$$\hat{x}(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} \mathrm{d}t, \quad x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{x}(\omega) e^{j\omega t} \mathrm{d}\omega,$$

on the time-frequency mesh introduced above in the main.m file, and express these Fourier integrals in terms of the discrete Fourier transforms (DFTs)

$$\hat{x}_k = \sum_{n=0}^{N-1} x_n e^{-j\frac{2\pi nk}{N}t}, \quad x_n = \frac{1}{N} \sum_{n=0}^{N-1} \hat{x}_k e^{j\frac{2\pi nk}{N}t}.$$

Use this result to approximate $\hat{q}(\omega, z)$ in terms of $\hat{\mathbf{q}}(z) = \mathrm{DFT}(\mathbf{q}(z))$.
To summarize, write down the channel model in the frequency $\hat{\mathbf{q}}(0) \mapsto \hat{\mathbf{q}}(z)$ and in time $\mathbf{q}(0) \mapsto \mathbf{q}(z)$, and their input constraints.

□

Question 12. Write an m-function `channel(t, q0t, z, sigma2, B)` that outputs a realization of the stochastic process $q(t, z)$ given $q(t, 0)$.
You need to discretize noise $n(t, z)$ to a vector $\mathbf{n}(z)$, and (5) to a discrete autocorrelation function $\mathsf{E}(\mathbf{n}(z)\mathbf{n}^H(z))$. Hint: equate the total noise power in the continuous and discrete models.

```
function [qzt, qzf] = channel(t, q0t, z, sigma2, B)

a = sigma2*B*z % total noise power in B Hz and distance [0, z]
f = ...        % get the f vector from t vector

q0f = ...      % input in frequency
qzf = q0f.* ... % output in frequency

% add Guassian noise in frequency, with correct variance
qzf = qzf + ...

qzt = ... % back to time

end
```

□

2.2.3 Receiver

Equalization. Equalization is the task of inverting the channel transformation in the absence of noise. For the dispersive channel, this can be done in the

frequency domain

$$\begin{aligned}\hat{q}_e(\omega,\mathcal{L}) &\triangleq \hat{h}^{-1}(\omega,\mathcal{L})\hat{q}(\omega,\mathcal{L})\\ &= \hat{q}(\omega,0)+\hat{h}^{-1}(\omega,\mathcal{L})\hat{z}(\omega)\\ &\triangleq \hat{q}(\omega,0)+\hat{w}(\omega),\end{aligned}$$

where $\hat{w}(\omega)=\hat{h}^{-1}(\omega,\mathcal{L})\hat{z}(\omega)$. In the time domain

$$q_e(t,\mathcal{L}) = q(t,0)+w(t), \tag{13}$$

where, *e.g.*, $q_e(t,\mathcal{L})=\mathcal{F}^{-1}(\hat{q}_e(\omega,\mathcal{L}))$.

Question 13. Characterize the stochastic processes $w(t)$ and $\hat{w}(\omega)$, namely, state their PDFs and the statistical properties (the mean and the auto-correlation function).

□

Question 14. Write an m-function `equalize(t, qzt, z)` that outputs $q(t,0)$ given $q(t,z)$ in the absence of noise.

```
function [qzte, qzfe] = equalize(t, qzt, z)

f = ...         % get the f vector from t vector

qzf = ...       % input in frequency
qzfe = qzf.* ... % output in frequency


qzte = ... % back to time

end
```

Let us test the code so far. Add the following to the main.m file.

```
T = ...
N = ...
dt = ...
t = ...

A   = 1;
q0t = A*exp(-t.^2); % Gaussian input, for testing
q0f = ... % input in frequency
% plot below the input in t & f. You must tune T and N!
...
...

% propagation
[qzt, qzf] = channel(t, q0t, z, 0, 0); % output in t,f. Zero noise.
plot(t, q0t, ...)        % plot input output in t; tune T and N
    accordingly
```

```
plot(f, abs(q0f), ...)   % plot input output in f; tune T and N
    accordingly

% equalization and comparison
[qzte, qzfe] = equalize(t, qzt, z);     % equalized output
plot(t, q0t, ...) % plot input & equalized output in t
compare(q0t, qzte) % you should write the function compare
```

□

Demodulation. The demodulation consists of obtaining the received symbols at the output via projection

$$\hat{s}_\ell = \sqrt{B} \int_{-\infty}^{\infty} q_e(t, \mathcal{L}) \operatorname{sinc}(Bt - \ell) \mathrm{d}t. \tag{14}$$

Question 15. Write an m-function `shat = demod(t, qzte, B, Ns)` to compute $\hat{s}$ from $q_e(t, \mathcal{L})$, by implementing (14). Make sure that the basis elements have unit energy: $\|\operatorname{sinc}(x)\|^2_{L^2(\mathbb{R})} = \left\|\sqrt{B}\operatorname{sinc}(Bx)\right\|^2_{L^2(\mathbb{R})} = 1$.

```
function shat = demod(t, qzte, B, Ns)

shat = ...

end
```

□

Detection. The optimal receiver is the *maximum likelihood (ML) detection* applied to the joint conditional PDF $p(\hat{s}^n|s^n)$, *i.e.*,

$$\tilde{s}^n = \underset{s^n \in \mathcal{C}^n}{\operatorname{argmax}} \quad p(\hat{s}^n|s^n). \tag{15}$$

Question 16. Substitute (13) into (14) and show that the continuous-time model $q(t, 0) \mapsto q(t, z)$ is discretized to the discrete-time model $s^n \mapsto \hat{s}^n$, described by

$$\hat{s}_\ell = s_\ell + Z_\ell, \quad \ell = 1, 2, \ldots,$$

where $Z_\ell \overset{\text{i.i.d.}}{\sim} \mathcal{N}_{\mathbb{C}}(0, c)$. Determine c.
Conclude that the channel $s^n \mapsto \hat{s}^n$ is memoryless, *i.e.*,

$$p(\hat{s}^n|s^n) = \prod_{\ell=1}^{n} p(\hat{s}_\ell|s_\ell). \tag{16}$$

Therefore the maximization in (17) can be performed separately per component $p(\hat{s}_\ell|s_\ell)$

$$\begin{aligned}\tilde{s}_\ell &= \underset{s_\ell \in \mathcal{C}}{\operatorname{argmax}} \quad p(\hat{s}_\ell|s_\ell) \\ &= \underset{s_\ell \in \mathcal{C}}{\operatorname{argmin}} \quad |\hat{s}_\ell - s_\ell|^2 ,\end{aligned} \tag{17}$$

where we used

$$p(\hat{s}_\ell|s_\ell) = \frac{1}{2\pi c} \exp\left(-\frac{1}{2c}|\hat{s}_\ell - s_\ell|^2\right). \tag{18}$$

We thus obtain that, for the AWGN channel, the ML detector is given by the minimum distance detector (17).
Find decision regions and simplify the optimal receiver.
Write an m-function `stilde = detector(shat, cnt)` that implements the ML detection. The function takes the received-equalized symbols $\hat{s}^n$ and outputs the ML estimates $\tilde{s}^n$.

```
function stilde = detector(shat, cnt)

stilde = ...

end
```

□

Demapping. This step is symbols-to-bits mapping, that you should implement in `bhat = symb_to_bit(stilde, cnt)` where cnt is the constellation. Use a look-up table for mapping and demapping.

Question 17. Check first the end-to-end implementation when noise is zero. In this case, you should obtain $\hat{\mathbf{s}} \approx \mathbf{s}$ with high accuracy.
Complete the main.m file.

```
% modulation
nb = 64; % number of bits
M = 16; % order of modulation
ns = ... % number of symbols
b= ... % random bit sequence
s = bit_to_symb(b, cnt);
q0t = mod(t, s, B);

% propagation & equalization. Set the noise to zero for now
[qzt, qzf] = channel(t, q0t, z, sigma2, B); % output in t,f
[qzte, qzfe] = equalize(t, qzt, z);     % equalized output

% demodulation
shat = demod(t, qzte, B);
```

```
% detection
stilde = detector(shat, cnt);
bhat = symb_to_bit(stilde, cnt);
```

Do you obtain $\tilde{\mathbf{s}} = \mathbf{s}$ and $\hat{\mathbf{b}} = \mathbf{b}$ in the absence of noise?

□

Inter-symbol interference. One type of distortion in communication is *inter-symbol interference (ISI)*, explained in the class.

Question 18. Simulate the output for the input

$$q(t,0) = A_1 e^{-\frac{(t+t_0)^2}{2D^2}} + A_2 e^{-\frac{(t-t_0)^2}{2D^2}},$$

where $A_1 = 1$, $A_2 = 2$, $D = 1$ and $t_0 = 4$. Plot $q(t,0)$ and $q(t,1)$ (without equalization) in the same graph.
Does dispersion give rise to ISI in time (*i.e.*, the two parts in $q(t,0)$ that are non-overlapping at $z = 0$ overlap at $z = 1$)? Repeat for $t_0 = 7$. Is ISI completely canceled with equalization?
Does dispersion give rise to interaction between frequency components? Which domain is preferred for modulation and equalization?

□

2.2.4 Performance evaluation

In this section, we compute the BER as a function of the signal-to-noise ratio (SNR), numerically and analytically.

Back-to-back setup. Make sure that you obtain $\hat{s}^n = s^n$ and $\hat{b}^n = b^n$ in the back-to-back setup where the channel is $q(t, \mathcal{L}) = q(t,0)$. Also, make sure that you obtain zero error in the noise-free channel with equalization.

Noisy channel. We now turn to the noisy channel.

Question 19. *(Simulated BER).* Add noise to the channel and compute the bit error rate BER $= p(\hat{b}_i \neq b_i)$ (or the symbol error rate). Complete the main.m file.

```
a = 1; % constellation radius
M = 2; % the number of points in the constellation
ser = ser(s,shat); % symbol error rate
ber = ber(b, bhat); % bit error rate

% plot the constellation at TX and RX. Change the annotation
plot(real(cnt), imag(cnt), '*', real(shat), imag(shat), '.');
% plot the BER
plot(snr, ber)
```

The BER should be plotted as a function of the SNR $= \mathcal{P}/P_n$ at TX, where $\mathcal{P}$ is the signal power and P_n is the total noise power (from $z = 0$ to $z = \mathcal{L}$). To compute $\mathcal{P}$, substitute (7) into (2) and prove that the signal power equals to the constellation power, *i.e.*, $\mathcal{P} = \mathsf{E}|s_k|^2$, $\forall k$. When $p = 1/2$,

$$\mathsf{E}|s_\ell|^2 = \frac{1}{M}\sum_{s\in\mathcal{C}}|s|^2, \quad \forall \ell, \tag{19}$$

which can be computed easily for a given constellation. The SNR can thus be scaled by varying the constellation radius a.
You can choose the number of symbols $n = 10000$, run the code $N_r = 1$ time, and count the errors. It is, however, easier to choose, say, five symbols $n = 5$, and run the same code $N_r = 500$ times in parallel over 4 CPU cores. You can change n and N_r as long as the the graphs are insensitive to the values of these parameters.
The final results of this section are: (i) constellation plot; (ii) the plot of the BER as a function of SNR, for an M-QAM constellation with $M = 2, 4, 8, 16$ *in the same figure.*

□

2.3 Generative deep neural network

We now consider the NLS equation (4) with dispersion, nonlinearity and noise. We solve (4) numerically using an algorithm called *split-step Fourier method* (SSFM) [4]. The SSFM is a generative deep neural network described below [5]. Consider a model described by the partial differential equation (PDE):

$$\frac{\partial q(t,z)}{\partial z} = \underbrace{L_L(q)}_{\text{linear}} + \underbrace{L_N(q)}_{\text{nonlinear}} + \underbrace{n(t,z)}_{\text{noise}}, \tag{20}$$

where L_L and L_N are linear and nonlinear operators. In the stochastic NLS equation (4),

$$L_L(q) = -j\frac{\partial^2 q}{\partial t^2}, \quad L_N(q) = -2j|q|^2 q. \tag{21}$$

Discretize the distance $\mathcal{L}$ into a large number $M \to \infty$ of segments of length $\epsilon = \mathcal{L}/M$. Discretize the time interval $[-T/2, T/2]$ into a large number $n \to \infty$ of intervals of step size $\mu = T/n$.
Divide the PDE (20) into three parts shown in (20), and perform successively the following linear, nonlinear and noise transformations in distance.

Linear transformation. The linear part of PDE (20) with (21) is

$$\frac{\partial q(t,z)}{\partial z} = -j\frac{\partial^2 q(t,z)}{\partial t^2}. \tag{22}$$

Taking the Fourier transform of both sides of (22), we obtain

$$\frac{\partial \hat{q}(\omega,z)}{\partial z} = j\omega^2 \hat{q}(\omega,z).$$

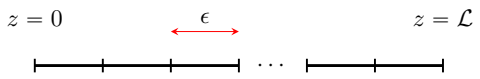


Fig. 4: Discretization of the channel in distance.

The solution is

$$\hat{q}(\omega, z) = e^{j\omega^2 z}\hat{q}(\omega, 0). \tag{23}$$

In the time domain

$$q(t, z) = \frac{1}{\sqrt{-j4\pi z}} e^{-j\frac{t^2}{4z}t^2} * q(t, 0),$$

where we used $e^{-\frac{t^2}{2\lambda}} \leftrightarrow \sqrt{2\pi\lambda}e^{-\frac{\lambda\omega^2}{2}}$.
It follows that dispersion is a simple phase change (all-pass filter) in the frequency domain. In the time domain, dispersion is a convolution, giving rise to memory and pulse broadening.
Let the input and output of the linear transformation in a given layer be $\mathbf{X} \in \mathbb{C}^n$ and $\mathbf{V} \in \mathbb{C}^n$, respectively. Discretizing (23), we obtain

$$\boxed{\mathbf{V} = W\mathbf{X}.} \tag{24}$$

Here,

$$W = D^H \Gamma D, \tag{25}$$

where D is the discrete Fourier transform (DFT) matrix of size n, and $\Gamma = \mathrm{diag}(e^{j\epsilon\omega_i^2})$ where (ω_i) is discretization of ω. It follows that linear transformation is multiplication by a (convolutional) *weight matrix* $W \in \mathbb{C}^{n\times n}$.

Remark 2. In the generative model, W is given by (25) and is not learned. □

Nonlinear transformation. The nonlinear part of PDE (20) with (21) is

$$\frac{\partial q(t, z)}{\partial z} = 2j|q(t, z)|^2 q(t, z).$$

It can be verified that the solution in the time domain is

$$q(t, z) = e^{2j|q(t,0)|^2 z} q(t, 0). \tag{26}$$

It follows that nonlinearity is a signal-dependent phase change in the time domain. In the frequency domain, nonlinearity gives rise to a convolution and spectral broadening.
Let the input and output of the nonlinear step in a given layer be $\mathbf{V} \in \mathbb{C}^n$ and $\mathbf{U} \in \mathbb{C}^n$, respectively. Discretizing (26)

input : $\mathbf{X} \in \mathbb{C}^n$, number of layers M, and parameters ϵ, P_n.
output: $\mathbf{Y} \in \mathbb{C}^n$
for $l = 1$ **to** M **do**
  $\mathbf{V} = W\mathbf{X}$, where the weight matrix W is given by (25).
  $\mathbf{U} = \sigma(\mathbf{V})$, where the activation function is given by (27).
  $\mathbf{Y} = \mathbf{V} + \mathbf{N}$, where $\mathbf{N} \sim \mathcal{N}_{\mathbb{C}}(0, P_n I_n)$
  $\mathbf{X} = \mathbf{Y}$
end
return $\mathbf{Y}$

Algorithm 1: Generative fully-connected feedforward neural net.

$$U_i = \sigma(V_i),$$

where $i = 1, 2, \ldots, n$, and $\sigma : \mathbb{C} \mapsto \mathbb{C}$ is the activation function

$$\sigma(x) = xe^{2j\epsilon|x|^2}. \tag{27}$$

Note that nonlinearity is memoryless, *i.e.*, it acts component wise.

Noise addition. This is a simple noise addition

$$\mathbf{Y} = \mathbf{U} + \mathbf{N}, \quad \mathbf{N} \sim N(0, P_n I_n),$$

where $P_n = \sigma^2 B_n \epsilon$ is the noise power.

Question 20. Write a function that generates a realization of a random vector in $\mathbf{Z} \in \mathbb{C}^n$ drawn from $\mathcal{N}_{\mathbb{C}}\big(0, \sigma^2 I_n\big)$.

```
function Z = noise(n, sigma2)

Z = ...       % you need real and imaginary components
end
```

□

Remark 3. In neural networks, one performs successive linear and nonlinear transformations, and there is no noise step. In the generative neural net there is a noise step, so that the data set comes from a probability distribution. There will not be a noise step in the neural net designed for inference. □

Question 21. Write an m-function `nnet_gen(x, nz, params)` that outputs `y` given `x`, where nz is the number of layers and `params` is a structure or class containing the required parameters.
The code is an implementation of the Algorithm (1). We first write a code for a feed-forward neural net that may not be very efficient. Later we shall discuss efficient numerical implementation in the context of the deep learning.

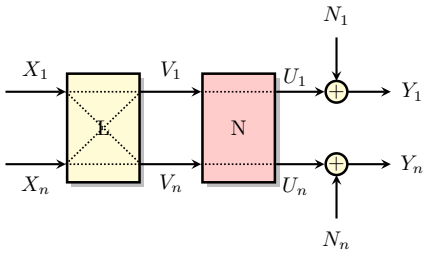


Fig. 5: One layer of the generative neural network, consisting of a linear, a nonlinear and a noise transformation.

```
function y = nnet_gen(x, nz=500, params)

% pre-compute the fixed weight matrix W
n = len(x)
f = ...                 % frequency vector
w = 2*pi*f              % angular frequency vector

z = params. ...
dz = z/nz               % epsilon
h = exp(j*w.^2*dz);     % all-pass filter
D =                     % DFT matrix of size n
W = conj(D') * ...

% Loop over the nnet layers
for k=1:nz

    % linear transformation -- multiplication by the weight matrix W
    x = W*x

    % activation function
    x = sigma(x, dz)

    % noise addition
    x = x + ...

end

y = x

end
```

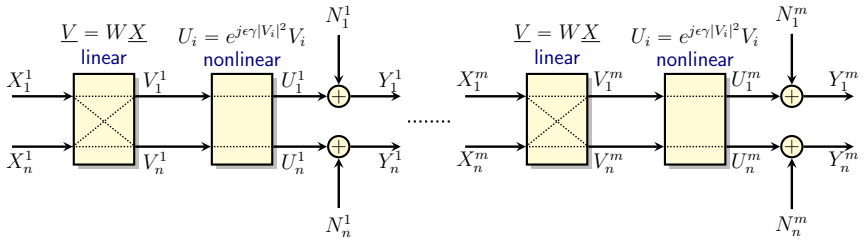


Fig. 6: The generative neural network with several layers.

3 Predictive model

TBC.

A Units

The conversion of unit for the loss coefficient α is as follows. Let α denote the loss coefficient for signal in m^{-1}, and $a_{\mathrm{dB}}(z)$ the power loss in the dimensionless unit dB at distance z. Let $P(z)$ denote the signal power at distance z.
The power decays 0.2 dB every 1 km, *i.e.*, $a_{\mathrm{dB}}(1\mathrm{km}) = 0.2$ dB. We have

$$10\log_{10}\left(\frac{P(0)}{P(z)}\right) = a_{\mathrm{dB}}(z), \quad \text{at } z = 1\mathrm{km},$$

or $P(0)/P(z) = 10^{a_{\mathrm{dB}}(z)/10}$ at $z = 1\mathrm{km}$. Since $P(0)/P(z) = e^{\alpha z}$, we have

$$\begin{aligned}
\alpha &= \frac{\log(10^{\frac{a_{\mathrm{dB}}(z)}{10}})}{z} \\
&= \frac{\log(10)}{10z} a_{\mathrm{dB}}(z) \\
&= \frac{\log(10)}{10} a_{\mathrm{dB}} \quad \mathrm{km}^{-1} \\
&= 10^{-4}\log(10) a_{\mathrm{dB}} \quad \mathrm{m}^{-1}.
\end{aligned}$$

We write $a_{\mathrm{dB}} = 0.2$ dB/km to mean $a_{\mathrm{dB}}(z = 1\mathrm{km}) = 0.2$ dB. The notation dB/km should not suggest that a_{dB} has unit m^{-1}.

References

[1] R. G. Gallager, *Principles of Digital Communication.* Cambridge, UK: Cambridge University Press, 2008.

[2] A. Lapidoth, *A foundation in digital communication.* Cambridge University Press, 2017.

[3] G. D. Forney, Jr., "Principles of digital communication II." Massachusetts Institute of Technology, Mar. 2005, MIT OpenCourseWare, Course no. 6.451, Lecture Notes.

[4] G. Kramer, M. I. Yousefi, and F. Kschischang, "Upper bound on the capacity of a cascade of nonlinear and noisy channels," in *IEEE Info. Theory Workshop*, Jerusalem, Israel, Apr. 2015, pp. 1–4.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press Cambridge, 2016.