Communication Theory
LAB 3: Analog Modulations - PM, FM

Objectives

In this activity the student will learn to:

- Study analog signals in MATLAB.
- Analyze the effect of non-coherent receivers on amplitude modulations.
- Draw conclusions about the properties of different modulations.
- Analyze the effect of noise in some analog modulations.

Phase modulation (PM)

Download the MATLAB files `P2_Demo_PM.m` and `etiquetas.m` and copy them to the working directory. The file `P2_Demo_PM.m` demonstrates the basic process of modulation and demodulation of a phase modulation for a very simple modulating signal, a sinusoidal signal with a frequency of 5 kHz, using the MATLAB communications library, in particular functions:

- `pmmod`: PM signal modulation
- `pmdemod`: PM signal demodulation

It is recommended to start the practice by running this demo file to understand how to simulate the modulation and demodulation of signals, and how to visualize these signals in the temporal and frequency domains.

Exercise 1

Looking at the figures, and the demo code to see how the different figures are obtained, carry out the following sections:

a) Represent the modulating and modulated signals in the interval $2 \leq t \leq 2.2$ seconds and the estimates of their power spectral densities.

   i) Explain in detail whether or not the results obtained coincide with those expected, and if not, what is the difference that can be seen with respect to the theoretical results.

b) Explain the difference between the results of estimating the signal-to-noise ratio without filtering and with filtering prior to demodulation.

Exercise 2

Modulating the same sinusoid as in the demo file, and with the same carrier frequency:

a) With a standard deviation for the discrete noise $\sigma = 0.2$, and using a phase deviation

$$k_p = \frac{\pi}{2} \times C$$

calculate the signal-to-noise ratio by demodulating without filtering (equivalent to low-pass filtering with a bandwidth of 1000 Hz) and filtering between 80 and 120 Hz for values of C between 0.1 and 1.0 in steps of 0.1[1].

   i) Represent the signal-to-noise ratios as a function of C, both linear and dB.

[1] Since the noise realizations are random, to have statistically reliable results, the result obtained with 100 different realizations should be averaged

II) Discuss in detail whether the results obtained are as expected, or if not, how they differ from the theoretical results.

b) With a phase deviation

$$k_p = \frac{\pi}{2}$$

calculate the signal-to-noise ratio by demodulating without filtering (equivalent to low-pass filtering with a bandwidth of 1000 Hz) and filtering between 80 and 120 Hz for noise standard deviation values between 0.1 and 1.0 in steps of 0.1.

I) Represent the signal-to-noise ratios as a function of C, both linear and dB.

II) Discuss in detail whether the results obtained are as expected, or if not, how they differ from the theoretical results.

Frecuency modulation (FM)

Download the MATLAB files `P2_Demo_FM.m` and `etiquetas.m` and copy them to the working directory. The file `P2_Demo_FM.m` demonstrates the basic modulation process for frequency modulation with a very simple modulating signal, a sinusoidal signal with a frequency of 5 kHz (in this case a cosine), using the library MATLAB communications, in particular the function:

- `fmmod`: PM signal modulation

It is recommended to start by running this demo file to understand how to simulate signal modulation, and how to visualize these signals in the temporal and frequency domains.

Exercise 3

Looking at the figures, and the demo code to see how the different figures are obtained, carry out the following sections:

a) Represent the modulating and modulated signals in the interval $2 \leq t \leq 2.2$ seconds and the estimates of their power spectral densities.

I) Explain in detail whether or not the results obtained coincide with those expected, and if not, what is the difference that can be seen with respect to the theoretical results.

II) Compare the modulated signal in this interval with the modulated signal previously obtained for a PM modulation, where the modulating signal was a sine instead of a cosine, and explain the similarities and/or differences between the two.

b) Varying the frequency deviation from 2 to 10 Hz in steps of 2, compare the FM modulated signal in the interval $2 \leq t \leq 2.2$ seconds with the PM modulated signal from Exercise 1:

I) Draw both modulated signals in each case.

II) Calculate the covariance of the difference signal between both (which measures the energy of said difference).

III) In view of the previous results:

1) For what phase deviation are the two modulated signals most similar?

2) Can there be any value for which the modulated signals are completely equal? If so, calculate this value and check whether the signals are equal or not.

Annex A : Some MATLAB functions

Name	Short description
clear	Clears variables and functions from memory (clear removes them all)
close	Closes all rendering windows (close all closes them all)
randn	Generate random numbers with Gaussian distribution
length	Gets the length of a vector
size	Gets the size of a vector, matrix or cell array
mean	Calculates the mean value of a vector or the rows/columns of a matrix
max	Calculates the maximum value of a vector or the rows/columns of a matrix
min	Calculates the minimum value of a vector or the rows/columns of a matrix
var	Gets the variance of a vector (or the columns of a matrix)
cov	Gets the covariance of a vector (or the covariance matrix of a matrix)
std	Gets the standard deviation of a vector (or the columns of a matrix)
abs	Gets the module of the elements of a vector or matrix
angle	Gets the phase of the elements of a vector or matrix
fft	Calculates the discrete Fourier transform (DFT) using the fast algorithm *Fast Fourier Transform* (FFT)
periodogram	Estimation of power spectral density using periodograms.
figure	Generates a new representation window
plot	Represents a continuous function in time
subplot	Divides a representation window into several rows and columns to put several figures in the same window
stem	Representation for a discrete time function
semilogy	Represents a function with a logarithmic scale on the ordinate axis
axis	Controls the scale and appearance of a figure
xlim	Specifies the range of the ordinate axis (x-axis) of a displayed figure
ylim	Specifies the range of the abscissa axis (y axis) of a displayed figure
xlabel	Allows you to put a label on the abscissa axis of a figure
ylabel	Allows you to put a label on the ordinate axis of a figure
title	Allows you to give a title to a figure
legend	Allows you to label the different curves of a figure
hold	Allows you to draw a function in a window maintaining the previous representation
grid	Draws a grid over a representation, making it easier to identify values
find	Find the indices of the elements of a vector that satisfy a condition
print	Allows you to save a figure in different graphic formats

Tabla 1: Some general MATLAB commands that may be useful.

To find more information about these functions or any other functions, you can use the Matlab help command.

Annex B: Representation of continuous signals in MATLAB

MATLAB is a software that works with vectors and matrices, which does not directly handle continuous-time signals. However, sometimes, as in this practice, MATLAB is used to represent continuous time signals. In this case the usual methodology is to obtain samples of the continuous time signal at regular intervals (every T_s seconds), which produces a discrete sequence of values

$$x[m] = x(t)\big|_{t=mT_s} = x(mT_s)$$

which can be handled in MATLAB.

If it is desired to produce a representation of these signals, the sampling rate $f_s = \frac{1}{T_s}$ must be high enough so that visually the union of the signal samples with lines is an approximation reasonable signal in continuous time. In that case, these samples can be represented with the function `plot`[2]. Although this strategy gives useful results, it must be kept in mind that in MATLAB we do not have the entire signal but only samples of it in a limited set of time instants, which has some consequences in signal processing. Some of these consequences are discussed below.

The frequency representation of a signal, such as the Fourier transform (FT) or the power spectral density (PSD), are based on the discrete Fourier transform (DFT) of a discrete sequence, which has several implications when used to estimate samples from a Fourier transform (see Annex C for a detailed explanation). The MATLAB function that calculates the DFT of a discrete sequence is `fft`, but there are also functions that allow estimating power spectral densities, such as the `periodogram` function (implements a periodogram, which averages modules squared of DFT).

For the calculation of the frequency representation, the sampling rate is relevant, since the bandwidth (in Hz) that can be represented is up to half of that sampling rate.

To calculate the energy/power of a signal, the time axis associated with the samples must be taken into account, since without this axis the values obtained will be scaled.

Another important limitation appears in the representation of white processes. In MATLAB only finite energy sequences can be generated, while by definition a white random process, in continuous time, is a process with infinite power. What actually allows you to simulate a discrete-time sequence in MATLAB is a white process filtered with an ideal low-pass filter whose bandwidth is half the sampling frequency, whose power is already finite.

Annex C : Relationship among samples of the DFT and the FT of a continuous signal

This appendix discusses some aspects related to the relationship between samples of the discrete Fourier transform of a discrete sequence of N samples and the Fourier transform of a continuous-time signal. These aspects are important for handling continuous-time signals with MATLAB.

Relationships among samples

The N-sample discrete Fourier transform (DFT) of a N-sample discrete signal is defined as

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

The Fourier transform (FT) of a continuous-time signal $x(t)$ is

$$X(j\omega) = \int_{-\infty}^{\infty} x(t)\ e^{-j\omega t}\ dt,$$

while the Fourier transform of a discrete-time signal is

$$X\left(e^{j\omega}\right) = \sum_{n=-\infty}^{\infty} x[n]\ e^{-j\omega n}.$$

[2]This function, by default, represents the union of the values of a discrete sequence with straight lines

If the discrete-time signal $x[n]$ is obtained by sampling a continuous-time signal $x(t)$ at a sampling rate $f_s = \frac{1}{T_s}$ samples/s, so that

$$x[n] = x(nT_s),$$

The Fourier transform of this discrete signal $x[n]$ is related to the Fourier transform of the continuous-time signal $x(t)$ in the following way (result of the Sampling Theorem)

$$X\left(e^{j\omega}\right) = \frac{1}{T_s}\sum_k X\left(j\frac{\omega}{T_s} - j\frac{2\pi}{T}k\right).$$

Taking into account the previous expressions, it is evident that if we have a discrete-time signal $x[n]$ of duration N samples, the DFT of N samples of it provides samples (values at specific frequencies) of the FT of the discrete sequence, which is implicitly equivalent to scaled samples of the FT of the continuous-time signal. Specifically, it can be seen that

$$X[k] = X\left(e^{j\omega}\right)\Big|_{\omega=\frac{2\pi}{N}k} = \begin{cases} \frac{1}{T_s}X\left(j\omega\right)\Big|_{\omega=\frac{2\pi}{NT_s}k} & \text{if } k \leq \frac{N}{2} \\ \frac{1}{T_s}X\left(j\omega\right)\Big|_{\omega=\frac{2\pi}{NT_s}(k-N-1)} & \text{if } k > \frac{N}{2} \end{cases}$$

Which means that when using the DFT to try to represent the FT of a signal in continuous time, in addition to rearranging the DFT samples (the `fftshift` function allows you to rearrange the DFT samples so that the null frequency is in the central position), a scale factor T_s must be introduced.

Windowing effect

When the duration of the signal $x(t)$ is greater than $N \times T_s$, the N samples from which the DFT is obtained do not represent $x(t)$, but rather a truncated or windowed version from it

$$x_w(t) = x(t)\, w_{NT_s}(t), \quad \text{with } w_T(t) = \begin{cases} 1, & \text{if } t \leq T \\ 0 & \text{in other case} \end{cases}$$

since the N discrete-time samples only represent that part of the signal $x(t)$. The fact of multiplying in the time domain by a window of duration NT_s seconds (windowed), assumes that the frequency response is affected by the convolution with a *sinc* function, which is the Fourier transform of a rectangular time window (see Figure **??**), which can distort the estimate of the FT from the DFT.

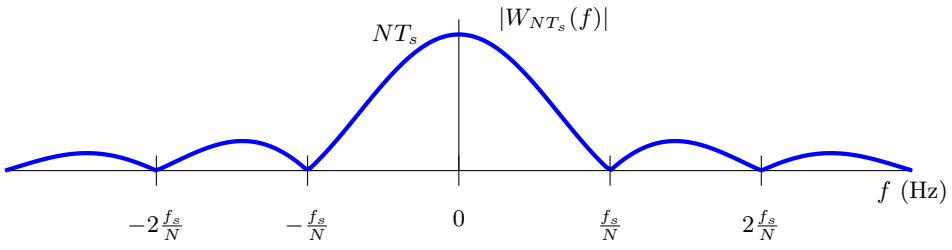

Figura 1: Frequency response of a window of NT_s seconds.

Spectrum of sinusoidal signals

When the frequency response of sinusoidal signals is to be represented, some peculiarities of this type of signals must be taken into account. The frequency response of a sinusoid of infinite length is Dirac deltas located at the frequency of the sinusoid. The particular properties of delta functions, and in particular their infinitesimal support, have some implications. On the one hand, normally the amplitude of the FT is the same whether the frequency response is being calculated in f (Hz) or $j\omega$ (rad/s), but for sinusoids this is not the case since the amplitude is $\frac{1}{2}$, or this quantity multiplied by 2π when working in $j\omega$ (a factor necessary to compensate for the fact that the delta support does not scale by that factor when going from f to $j\omega$). Furthermore, these functions do not comply with the scaling property of the Fourier transform, since a scaling of the time axis (change in frequency) does not affect the scaling in the amplitude of the transform, as happens with the rest of the functions (again this is due to the infinitesimal support of the delta in frequency; it can also be related to the fact that the power of the sinusoid is not modified when changing the frequency).

The fact that a sinusoid has infinite duration implies that the representation of its spectrum using DFT is affected by the windowing effect. The Fourier transform of a windowed sinusoid using a DFT of N samples and a sampling rate of f_s samples/s is plotted in Figure **??**.

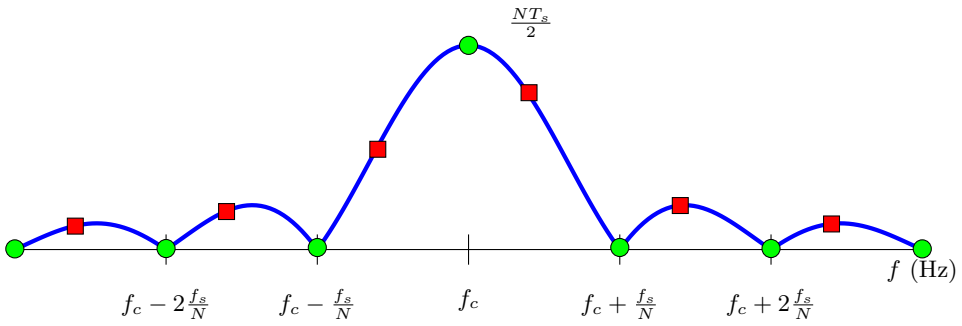

Figura 2: Frequency response of a windowed sinusoid in NT_s seconds.

It is important to note that the zero crossings of the `sinc` are spaced exactly the same amount as the frequencies at which the DFT samples of the FF are separated (remember that these frequencies are $\omega = \frac{f_s}{N}k$). This means that if the frequency of the sinusoid is a multiple of $\frac{f_s}{N}$, the frequencies where the FT is sampled will coincide with the zeros of the `sinc` and with the center of the main lobe (green circles in the figure), so the DFT result returns a delta. However, if this is not the case, it will be sampling at other points in the different lobes (red squares in the figure), which can distort the representation of the FT of the continuous time signal.