Lab 0

Generating multisines and random signals in Matlab

Objectives

The goal of this lab is to get acquainted with the use of Matlab to generate multisines and random noise signals, to be used in later labs as excitation signals for dynamic systems. You will learn

- how the DFT is defined (in Matlab in particular) and how it can be used to analyse periodic signals,
- how to generate multisines in the time domain and in the frequency domain,
- how you can construct an excitation signal in a given frequency band, and with a given frequency resolution.

0.1 Discrete Fourier Transform (DFT)

Consider a discrete-time signal $x(n)$, in the time window $n = 0, \ldots, N-1$. Recall the definition of the DFT of $x(n)$, and of the inverse DFT, in Matlab (the functions `fft` and `ifft` respectively):

$$\begin{array}{ll} \text{DFT} & \text{iDFT} \\ X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi kn}{N}}, & x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{\frac{j2\pi kn}{N}} \\ \text{for } k = 0, 1, \ldots, N-1 & \text{for } n = 0, 1, \ldots, N-1 \end{array} \tag{1}$$

An interpretation is that the DFT decomposes the time domain signal $x(n)$ into a linear combination of cosines and sines – or complex exponentials – of which the (complex) amplitudes are given by $X(k)$. The frequency axis k is expressed in *bin*. At bin k, the complex exponential $e^{\frac{j2\pi kn}{N}}$ is periodic in n, and has a period which fits exactly k times in the time interval of N points. This is shown graphically in Figure 0.1, keeping in mind that $e^{j\phi} = \cos(\phi) + j\sin(\phi)$.

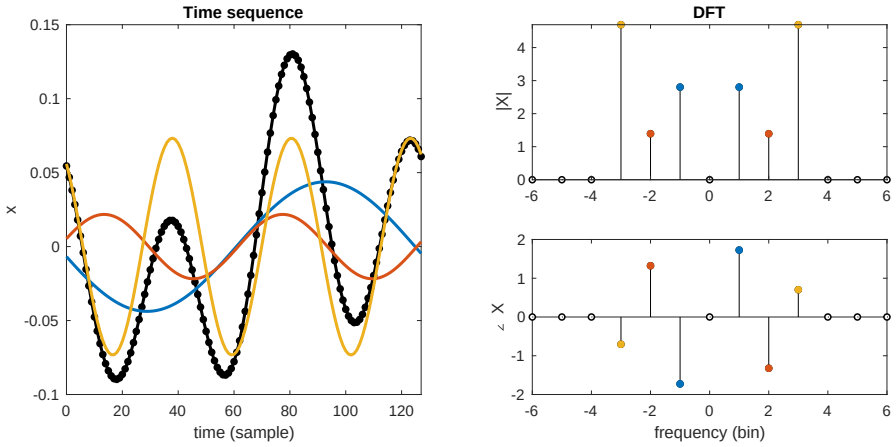

Figure 0.1: Left: time domain signal $x(n)$ (black), and its (co)sine components (coloured). Right: DFT of the signal: amplitude (top) and phase (bottom) of individual components $X(k)$.

If the discrete-time signal $x(n)$ represents a sampled continuous time signal, with a sampling frequency of f_s, then the iDFT can be rewritten as

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k) e^{j\omega_k n T_\mathrm{s}} \tag{2}$$

where $T_\mathrm{s} = \frac{1}{f_\mathrm{s}}$ is the sample time, and ω_k is the angular frequency at bin k.

TASK 0.1.1. Frequency axis in $\mathrm{rad/s}$**.** Prove that

$$\omega_k = \frac{2\pi}{T} k \tag{3}$$

(where $T = NT_\mathrm{s}$ is the window length expressed in seconds). (Hint: compare (1) and (2).) This will allow you to construct the frequency axis of the DFT of a sampled signal: ω_k for $k = 0, 1, \ldots, N/2$.

In fact, the DFT discretises the frequency axis. The resolution of this discretisation increases when the measurement time T increases – i.e. there are more 'bins per Hz' when measuring longer.

TASK 0.1.2. Fundamental frequency (1 bin). Prove that the fundamental frequency ω_1 of the DFT (which corresponds to one bin) is equal to

$$\omega_1 = \frac{2\pi}{T} = 2\pi\frac{f_\mathrm{s}}{N}. \tag{4}$$

TASK 0.1.3. Conjugate symmetric DFT. Prove that

$$X(N-k) = X(-k) = X^*(k). \tag{5}$$

(Hint: use $e^{j2\pi n} = 1$ for $n \in \mathbb{Z}$, and $x(n) \in \mathbb{R}$.)

This demonstrates that the DFT of a real signal is conjugate symmetric around the origin. Thus, the negative frequencies are obtained from the upper half of the DFT: $X(-k) = X(N-k)$.

0.2 DFT of a (co)sine

TASK 0.2.1. DFT of 3 periods of a cosine. Generate a cosine sequence in Matlab with a randomly selected phase, and with a period that fits exactly 3 times in a data sequence of $N = 1000$ samples. Make a plot of the DFT of this sequence (amplitude and phase).

TASK 0.2.2. Perfect reconstruction. From the DFT plot, check that the condition for perfect reconstruction is satisfied. Is there any leakage visible?

TASK 0.2.3. Interpretation of the frequency axis. At which indices of the DFT do you obtain non-zero values? Explain. (Keep in mind that Matlab indices start at 1, not at 0.)

TASK 0.2.4. Frequency axis in bins. Construct the frequency axis for the plots, expressed in bins.

TASK 0.2.5. Frequency axis in Hz. Consider that the sample frequency is $f_\mathrm{s} = 100\,\mathrm{Hz}$. Construct the frequency axis for the plots, expressed in Hz. (Hint: use the results from **Task 0.1.1**.)

0.3 Time domain construction of a multisine

Recall that a multisine is a sum of cosines, with frequencies that satisfy the condition for perfect reconstruction:

$$x(n) = \sum_{m=1}^{K} A_m \cos(\omega_m n T_\mathrm{s} + \varphi_m) = \sum_{m=1}^{K} A_m \cos\left(\frac{2\pi m}{N} n + \varphi_m\right) \tag{6}$$

The frequencies ω_m for which the amplitudes A_m are non-zero are called the **excited frequencies**.

TASK 0.3.1. Time domain random phase multisine. Generate a multisine in the time domain, by implementing (6), with $N = 1000$ samples and $K = 10$ excited frequencies. Set the amplitudes $A_m = 1$, and choose the phases φ_m randomly between 0 and 2π (i.e. a *random phase* multisine). Check that this multisine satisfies the condition for perfect reconstruction by plotting its DFT. Include the frequency axis, expressed in bin.

TASK 0.3.2. Frequency axis in Hz. For the multisine generated in **Task 0.3.1**, consider that the sampling frequency is $f_\mathrm{s} = 100\,\mathrm{Hz}$. Include the frequency axis expressed in Hz in the DFT plot, and the time axis expressed in seconds for the time domain plot.

TASK 0.3.3. Excite specific frequency lines. Generate a random phase multisine with a sampling frequency of 200 Hz, with excited frequencies

$$[4, 8, 12, 16, 20, 24]\,\mathrm{Hz}. \tag{7}$$

Plot the time and frequency domain results, with appropriate axes.

0.4 Frequency domain construction of a multisine

A multisine can easily be generated by immediately specifying the amplitudes and phases of the components. This comes down to constructing $X(k)$ directly (that is, in the frequency domain). One difficulty is that $X(k)$ must be constructed both for the positive and the negative frequencies. However, a trick can be used such that only the positive frequencies must be constructed, as in the following task.

TASK 0.4.1. Trick for frequency domain multisine construction. Consider the vector $\tilde{X}(k)$, such that

$$\tilde{X}(k) = A_k e^{j\varphi_k} \qquad \text{for } 1 \leqslant k \leqslant K \tag{8}$$

$$\tilde{X}(k) = 0 \qquad \text{otherwise} \tag{9}$$

Prove that

$$x(n) = N\Re\left\{\texttt{iDFT}\big(\tilde{X}(k)\big)\right\} = \sum_{k=1}^{K} A_k \cos\left(\frac{2\pi k}{N}n + \varphi_k\right) \tag{10}$$

where $\Re$ denotes the real part. (Hint: use the definition of `iDFT` in (1).)

TASK 0.4.2. Frequency domain multisine. Use the frequency domain approach to construct a random phase multisine, by using the trick from **Task 0.4.1**. Let $N = 1000$, and excite the first 30 bins. Make time and frequency domain plots (frequency axis expressed in bins).

TASK 0.4.3. Specified excited frequency band and frequency resolution. Construct a random phase multisine in the frequency domain, which excites the frequency band $[5, 15]$ Hz at 30 equidistantly spaced frequencies. Choose an appropriate sampling frequency. Make time domain and frequency domain plots (time axis in seconds, frequency axis in Hz). How long is one period of this multisine (expressed in seconds)?

0.5 Influence of the phase of the multisine

Until now, we choose the phases of the components of the multisine to be random. The choice of the phase has an important impact on the time domain properties of the multisine. More specifically, it will influence its crest factor (CF), defined as:

$$\text{Crest Factor} = \frac{\max(|x|)}{\text{RMS}(x)} \qquad \text{with RMS}(x) = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1} x^2(n)}. \tag{11}$$

TASK 0.5.1. Crest Factor. Construct a multisine, with $N = 500$, with the first $K = 60$ bins excited, and with the following phases:

- Random phase: chosen randomly in $[0, 2\pi]$ (uniform distribution),
- Schroeder phase: $\varphi_m = \frac{m(m+1)\pi}{K}$,
- Linear phase: $\varphi_m = m\pi$.

Make time and frequency domain plots (in samples and bins), and compute the Crest Factors. Describe, qualitatively, the relationship between the time domain plot and the crest factor. What is the advantage of a low/high crest factor?

0.6 Random noise signals

A popular excitation signal in the literature is random white noise. This is a signal which excites all frequencies, and which is stochastic, both in the time and in the frequency domain.

TASK 0.6.1. White Gaussian random noise. Generate a normally distributed (Gaussian), random, white noise sequence of $N = 1000$ samples, by using the Matlab function `randn`. Make time and frequency domain plots (axes in samples and bins). Observe that all the bins are excited, with random amplitudes and phases.

Note that this approach generates a signal which excites the full available frequency band. This is often not desired, for multiple reasons. For instance, most practical systems are only active in a limited frequency band. Thus, the input energy outside of that frequency band is typically wasted. Also, exciting the full frequency band is prone to cause alias errors, because harmonics created by the system under test will lie beyond the Nyquist frequency. For these reasons, it is better to limit the excitation frequency band, for instance by filtering.

TASK 0.6.2. Filtered random noise. Generate a filtered random noise sequence with $N = 1000$, sampling frequency 100 Hz, from a Gaussian white noise sequence (use `randn`). Do this by using the function `cheby1` to create a lowpass digital Chebyshev filter of order 5, ripple 2 dB, and such that the passband edge lies at 5 Hz. Filter the sequence by using the function `filter`. Make time and frequency domain plots (axes in seconds and Hz), and check that the excited frequency band is as expected. What do you observe in the stop-band of the filter? Is it equal to 0? Explain.

TASK 0.6.3. Periodic band-limited random noise. Generate a Gaussian random noise sequence (use `randn`), with $N = 1000$ and sampling frequency 100 Hz. Compute the DFT, and set the DFT at all frequencies beyond 5 Hz to zero:

$$\tilde{X}(k) = 0 \qquad \text{for } \omega_k > 2\pi 5\,\text{rad/s}, \tag{12}$$

$$\tilde{X}(k) = X(k) \qquad \text{otherwise}, \tag{13}$$

and use the expression

$$x(n) = 2\Re\left\{\texttt{iDFT}\left(\tilde{X}(k)\right)\right\} \tag{14}$$

to obtain the time sequence. Make time and frequency domain plots (axes in seconds and Hz).

If you repeat this time domain sequence $x(n)$ (by putting multiple copies of the sequence after each other), and computing the DFT of the result, no leakage should occur. Check this, and explain why this is the case.

In fact, the signal generated in **Task 0.6.3** can be interpreted as a multisine, with random phase *and* random amplitude.

0.7 Set the Root-Mean-Square of the signal

For all the signals generated above, the Root-Mean-Square (RMS) of the signal was not specified. The actual RMS value depends on the number of excited frequencies and on the choice of the amplitudes A_k.

TASK 0.7.1. RMS value. Set the RMS value of your favourite signal from the previous tasks to $\mathrm{RMS}_{\mathrm{des}} = 3$:

$$x_{\mathrm{des}}(n) = x(n)\frac{\mathrm{RMS}_{\mathrm{des}}}{\mathrm{RMS}(x)}. \tag{15}$$