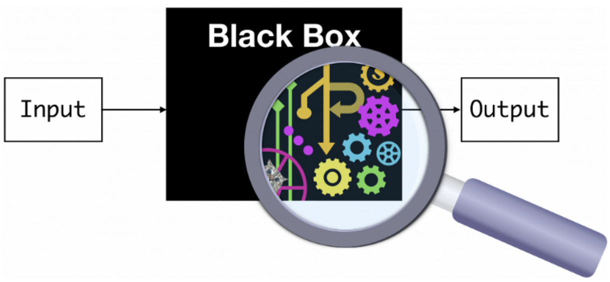
Introduction to System Identification (Year 2 Computing 2020–21)

Dominic Danks, Piper Fowler-Wright, Andreas Freise, Evgueni Goudzovski, Aaron Jones

Updated: October 6, 2020



School of Physics and Astronomy
University of Birmingham
Birmingham, B15 2TT

1 Introduction

In assignment 2 you will be presented with a number of unknown systems or 'black boxes'. The systems are represented by Python code that accepts input data and produces output data. The aim of the assignment is to create suitable input data for the boxes and record the corresponding output data, in order to learn more about what is going on inside each box. This process of determining the transfer characteristics is sometimes referred to as 'system identification'. A common practical example is the need to test subsystems, such as electronic modules, of an experimental setup. Even though the modules come with their data sheets, the actual behaviour might deviate somewhat from the specification. In precision experiments, even very small deviations can cause an experiment to fail in various interesting ways. Measuring the response of each part and creating an accurate model of (parts of) the system is an important aspect of debugging an experiment.

In this assignment we will use electronic circuits as our example systems to test, and each black box emulates the behaviour of one of the circuits described below.

For many systems, comparing input and output signals is much simpler in the frequency domain, i.e. when the input and output signals are described as spectra (signal components as functions of frequency) rather than as a time series (signal over time). The Fourier Transform can be used to transform time domain signal into the frequency domain and vice versa. You will make use of the Fast Fourier Transform module of the `numpy` package to investigate black box response in the frequency domain, which would allow identification of the specific electronic system contained in each box.

2 Discrete Fourier Transform

The theoretical foundations of the Fourier Transform and signal analysis in the frequency domain have been discussed in the Mathematics for Physicists course earlier this term. In the computing laboratory, we will focus on the practical applications of this method. Below we provide a simple recipe to use the discrete Fourier Transform as a tool to solve problems in this context.

Consider an electronic signal $x(t)$ recorded such that we have a finite set of uniformly spaced time-samples

$$x_n = x(t_n) \tag{1}$$

with t_n the equally spaced times ($n = 0, ..., N-1$). The total measurement time is $T = t_{N-1} - t_0$, and the sampling rate or frequency is given as $F_s = (N-1)/T$ samples per second.

Using the Discrete Fourier Transfrom (DFT), we can compute a new array of values that indicate how much of the signal is oscillating at a certain frequency. In the following we assume an even number of samples N.

The results of the DFT are given as

$$X_n = X(\omega_n) = \sum_{k=0}^{N-1} x_k \exp\left(-\frac{2\pi i}{N} kn\right), \tag{2}$$

with ω_n representing a list of uniformly spaced frequencies given in radians 'per cycle' with $\omega_n = 2\pi f_n$. The frequencies up to $n = N/2$ are given as:

$$f_n = nF_s/N. \tag{3}$$

Note that the maximum frequency is given as $F_s/2$. The sampling frequency defines what frequency content can be stored in the discrete time series. Note that the output values X_n are complex numbers. The remaining frequencies (with indices n ranging from $N/2+1$ to $N-1$) correspond to negative frequencies. In physics we often deal with real (measurable) signals, in which case the Fourier components are identical under the transformation $\omega \rightarrow -\omega$, i.e. the positive and negative frequency Fourier components are equal. In other words, all the information of the signal is contained in the first half of the frequency-samples.

We can perform an inverse transform to go back from the frequency domain to the time domain:

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k \exp\left(\frac{2\pi i}{N}kn\right). \tag{4}$$

The `numpy` commands for the DFT and the inverse DFT are:

```
X = numpy.fft.fft(x)
x = numpy.fft.ifft(X)
```

In order to plot the frequency spectrum X, you also need to determine the corresponding sample points in the frequency domain. The `numpy` function `.fftfreq` should be used for this purpose: if the original numpy array passed to `.fft` is of size `N` and has a constant spacing between elements of `timestep`, `.fftfreq(n, d=timestep)` returns the sample frequencies against which the output of `.fft` should be plotted. In general, the values returned by `.fft` will be complex and so it is important to plot the absolute values against these frequencies using `numpy.abs()`.

3 Transfer functions of electric circuits

Simple RLC circuits (made from resistors R, capacitors C and inductors L) are often used to selectively transmit signals according to their frequency and may be categorised as either low pass, band pass or high pass depending on the range of frequencies transmitted. Their behaviour is characterised by their **transfer function** depending on the frequency ω:

$$T(\omega) = \frac{V_{\text{out}}(\omega)}{V_{\text{in}}(\omega)}. \tag{5}$$

Each of the following filter circuits is contained within one of the three black boxes you will to investigate in task 1 of assignment 2:

1. an RL high pass filter (output taken across the inductor);
2. an RC low pass filter (output taken across the capacitor);
3. an RLC band pass filter (output taken across the resistor).

The resistance, capacitance and inductance values for each circuit contained in the black boxes will be unique for your assignment.

Note that an RL circuit can also act as a low pass filter, and an RC circuit can act as a high pass filter, if the output is taken across the resistor.

Recall the relationship between the input and output voltage for a potential divider circuit containing a number of electrical impedances in series:

$$V_{\text{out}} = \frac{Z_{\text{out}}}{Z_{\text{tot}}} V_{\text{in}}. \tag{6}$$

Here Z_{out} is the impedance of the component across which the output is taken and Z_{tot} the total impedance of the circuit; a linear sum for components in series. Given the complex impedance of a resistor (R), an inductor ($i\omega L$) and a capacitor ($1/i\omega C$), expressions for the total complex impedance and hence transfer function of each filter circuit may be determined.

A **Bode magnitude plot** is a graph of the magnitude of the transfer function $|T(\omega)|$ against angular frequency ω. These often use a decibel scale for the vertical axis, and a logarithmic scale for the horizontal axis (though you will not be using decibels in your assignments). The following provides some details on decibels and shows Bode plots of this type. The difference in decibels (dB) between two voltages, V_{out} and V_{in}, is defined by the logarithmic ratio

$$20\log_{10}\left|\frac{V_{\text{out}}(\omega)}{V_{\text{in}}(\omega)}\right| = 20\log_{10}|T(\omega)|\,. \tag{7}$$

The absolute difference is known as a **gain** if $V_{\text{out}} > V_{\text{in}}$, and a loss (or attenuation) if $V_{\text{out}} < V_{\text{in}}$. For example, $|T(\omega)| = \frac{1}{\sqrt{2}}$ corresponds to $20\log_{10}\left|\frac{1}{\sqrt{2}}\right| \approx -3$ dB, i.e. a loss of approximately 3 dB.

A quantity of particular interest is the **cut-off or corner frequency** f_0, and the corresponding angular frequency, $\omega_0 = 2\pi f_0$. This is defined as the frequency for which $|T(\omega)| = \frac{1}{\sqrt{2}}$, corresponding to an attenuation of approximately 3 dB. You will be asked to evaluate the cut-off frequency of an RL high pass circuit in task 2 of assignment 2.

Schematic Bode magnitude plots for each of the three circuits listed above are shown in Figures 1, 2 and 3. The parameter values used are for illustrative purposes only.

4 Interacting with the black box

Throughout assignment 2, you will be required to pass time-domain input voltage signals to the various ‘Box’ objects defined in the first code cell of `assignment2.ipynb`, and analyse the resulting output. This is done by invoking the `.process` method on each box. For example, the call `example_box.process(t, s_in)` passes a signal, whose amplitude at times in `t` is given by `s_in`, to `example_box`, and returns an array containing the amplitude of the resulting output signal. The use of `.process` is demonstrated in `black_box_example.ipynb`. It is highly recommended to run through that example notebook before attempting the assignment.

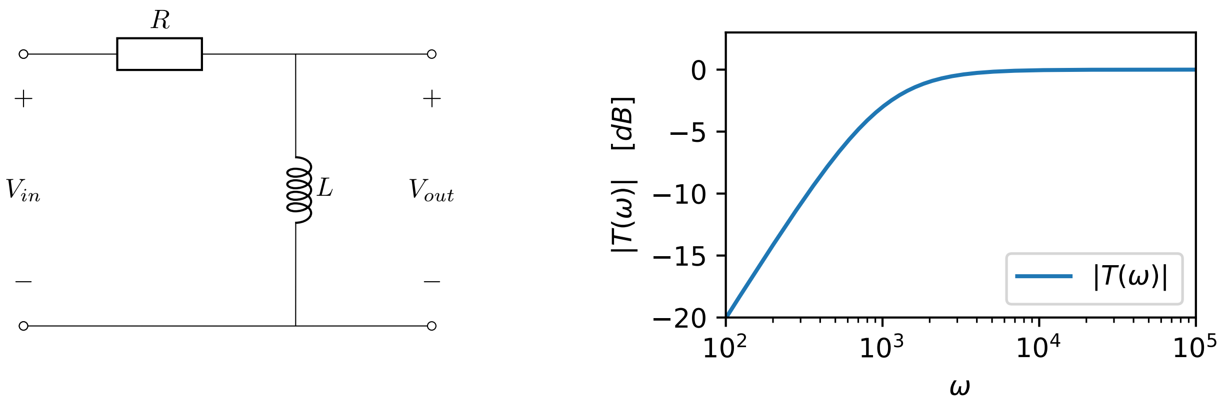
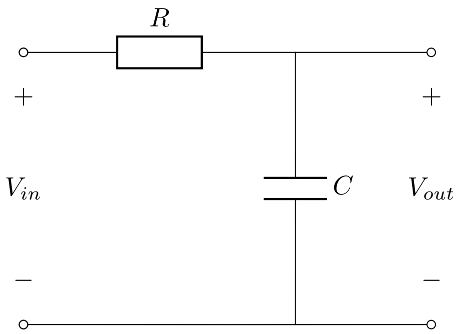


Figure 1: The RL high pass filter circuit (left) and its Bode magnitude plot (right) for $R = 10\ \Omega$ and $L = 10$ mH. The magnitude of the transfer function is given by $|T(\omega)| = \frac{\omega/\omega_0}{\sqrt{1+(\omega/\omega_0)^2}}$, where $\omega_0 = R/L = 10^3$ is the angular cut-off frequency.



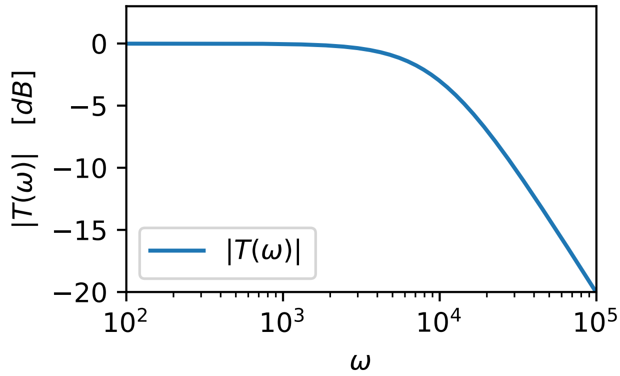


Figure 2: The RC low pass filter circuit (left) and its Bode magnitude plot (right) for $R = 10\ \Omega$ and $C = 10\ \mu$F. The magnitude of the transfer function is $|T(\omega)| = \frac{1}{\sqrt{1+(\omega/\omega_0)^2}}$, where $\omega_0 = 1/RC = 10^4$ is the angular cut-off frequency.

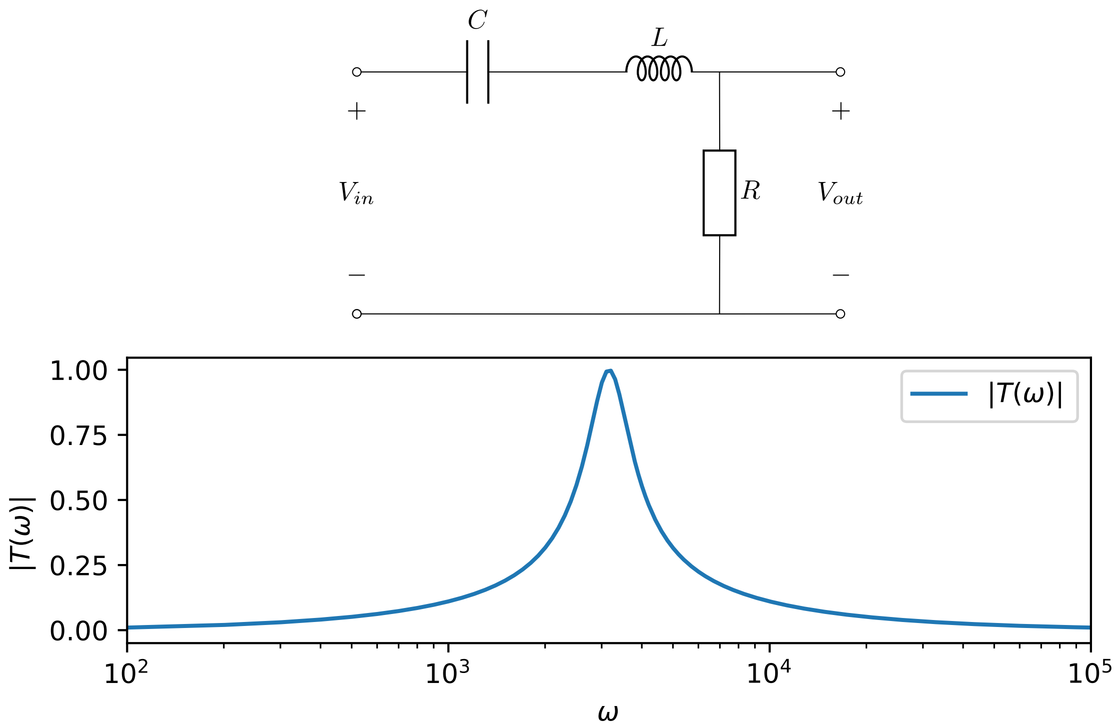


Figure 3: The RLC band pass filter circuit (top) and its Bode magnitude plot (bottom) for $R = 10\ \Omega$, $L = 10$ mH and $C = 10\ \mu$F. The decibel scale is not used for the sake of clarity. The magnitude of the transfer function is $|T(\omega)| = \frac{R}{\sqrt{R^2+(\omega L-1/\omega C)^2}}$.