RC Circuits and Filtering

Matlab Case Study for Signals and Systems (Draft)

In this course (and possibly others), you have learned that different signals can be expressed as a combination of different component frequencies. Being able to examine which frequencies are present in a signal and at what strengths is essential for working with nearly any system. Often when working with a signal, we will want to control which frequencies are kept and which are discarded.

Fortunately, you already have a tool for examining signals in the frequency domain: your ears! Perhaps at some point you’ve recorded your voice and found that the fan in your room has added some high-frequency noise you’d like to eliminate. Perhaps your downstairs neighbors have asked that you lower the bass of your stereo. Whatever the case may be, examining audio signals is an intuitive way to begin understanding the relationship between signals as a function of time and signals as a function of frequency.

Resistor-Capacitor circuits are common circuits used for analog signal filtering and can be used to construct filters that will reject some frequencies of signals while allowing others through mostly unattenuated. In this lab, we will explore their effects on various frequency inputs, and then apply those observations to some audio recordings.

In this case study, you will:

1. Familiarize yourself with the Simulink software included with MATLAB
2. Simulate the response of three different RC circuits to various sinusoidal inputs
3. Examine magnitude and phase plots of these circuits and relate them to the effect each circuit has on different input frequencies.
4. Examine power spectral density plots of the input and output of these circuits and relate them to the magnitude and phase plots of the circuits.
5. Extend this process to various audio files to connect your conceptual understanding of frequency analysis to your everyday experiences with sound

By the time you’re finished with this case study, you will have a clearer idea of the relationship between the time and frequency domain, the relationship between transfer functions and Bode plots, and the relation between sound and frequency.

# Simulink

Simulink is a piece of software packaged with your MATLAB installation that allows you to create simulated environments out of a wide variety of components, including transfer functions, block diagrams, waveform generators, and more. The folder for this case study includes a pre-made model, *RCcasestudy.slx*, which has some basic instructions for how to navigate the Simulink environment.

Simulink models are composed of blocks, each of which have some combination of input and output ports. By connecting the ports together, we can pass different signals through the model and examine the results.

# Impedance, Phasors, and AC Circuits

You are likely familiar with several methods for analyzing DC circuits in the time domain. Many of these concepts can be extended to analyzing AC circuits using the concepts of phasors and impedance.

A phase vector, or phasor, is a way of representing a periodic signal as a complex number.

# Transfer Functions

A transfer function can be thought of as a way of representing a system or process in terms of the effect it has on different frequencies of input. It is a complex function, meaning it can take in a complex number *s = σ+jω* and output another complex number (for an RC circuit, you can think of *s* as an AC signal. The value *σ* represents the DC power, and the value *ω* represents the AC frequency).

The magnitude of the transfer function output for a particular frequency is called the “gain,” and expresses how the amplitude of that frequency is changed as it passes through the system. Similarly, the phase of the output expresses how the phase of the frequency is changed as it passes through the system.

For instance, imagine we have an RC circuit with the following transfer function:

Say we are interested in knowing what would happen if we applied a 100 rad/s sine wave with an amplitude of 1V to the input of this circuit. This corresponds to an input of *s = σ+jω* = *0 + 100j*. The gain of the transfer function for this input is:

The phase of the transfer function for this input is:

This means that if we put a 100 rad/s sine wave through the RC circuit, the output will be a 100 rad/s wave with an amplitude that has been reduced by a factor of . In addition, the output will lag behind the input by , or an eighth of a period.

A close up of a mans face

Description automatically generated

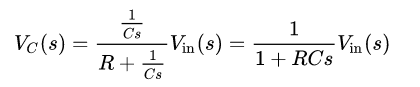
# Three Passive RC Filters

## Circuit 1:

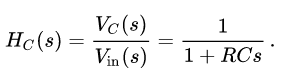
A picture containing clock

Description automatically generated

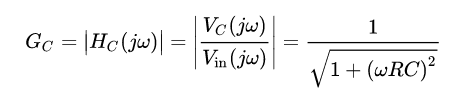
This circuit measures the voltage across the capacitor as a function of the voltage across both elements. You may notice that it resembles a voltage divider, with one resistor swapped for a capacitor. Using the concept of complex impedance, we can represent the resistance of the capacitor as and write the following expression:



Giving us the transfer function:



Remember that *s* is a complex number  *s = σ+jω,* where *ω* is frequency. Taking the magnitude of this transfer function:



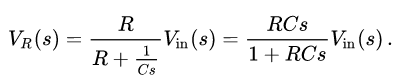
This equation models the gain of the circuit – the ratio between the amplitudes of the output and input. What happens to the gain as frequency increases? What happens when it is very large?

## Circuit 2

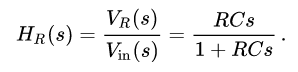
A picture containing object, clock

Description automatically generated

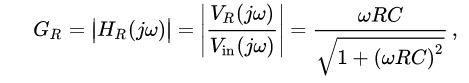
This circuit is similar to the previous one, but the voltage divider measures the voltage across the resistor instead.



This gives us the transfer function:



Which has magnitude:



What happens to the gain as frequency decreases? What happens when it is very small?

## Circuit 3

A close up of a antenna

Description automatically generated

This third circuit is effectively a cascaded system: the output of circuit 1 has been attached to the input of circuit 2. The transfer function for this circuit is not given here, as you will explore its properties during the case study.

# Case Study

Navigate to the Circuits Case Study folder in MATLAB and open the RCcasestudy.slx Simulink model. (If you are not in the correct folder, some aspects of the model may not load. Try running init.m in the Circuits Case Study folder and then running the simulation to fix this.). Complete the following tasks:

* Examine the Simulink model and read the annotations for each part of the model. Take some time to experiment with the model by connecting different input signals to the three RC circuits and observing the output. Each time you change the input, you will need to run the simulation again to update the output.
* What happens to high frequency signals as they pass through each circuit? What happens to low frequency signals? What happens to signals that are superpositions of both high and low frequency signals? Record your observations in your writeup.
  + Be sure to note changes in both the magnitude and the phase of these signals
* Circuit 1 and Circuit 2 both have a “cutoff frequency” of 1/(2πRC) Hz. Compare the output of both circuits using this frequency as an input. Record your observations in your writeup.
* Examine the Bode plots of each of the three circuits. Explain how the plots are consistent with your observations about which frequencies are attenuated by each circuit and which are not. Record your observations in your writeup.
* The chirp\_timeseries block contains a sound clip that starts at 1 Hz and increases logarithmically to 10 kHz over the course of three seconds. Put it through each of the RC filters and comment on the results.
* Make a brief (1-5 seconds) recording of your own and load it into the simulation using the sound2ts() function in the init.m script. Use the sound() function in matlab to play the sound back before and after putting it through circuit 3. If you don’t hear a difference, try adding some low and high frequency noise to the sample. Plot the power spectral density of your sample using fft(). Record your observations in your writeup.
  + For best results, record your sample using a mono track sampled at 44100 Hz.
  + The output of circuit 3 is saved to the workspace as the timeseries “simdata.output.”
  + The input of circuit 3 is saved to the workspace as the timeseries “simdata.input.”