Adaptive Filtering with the Least Mean Squares Algorithm

ECE4703

12/15/2011

Christopher Kevorkian

Kurt Snieckus

Introduction

In this lab, we develop an adaptive FIR filter and showcase its utility in system identification and noise cancelling. Using the FIR filter developed in Lab 2, we implement the Least Mean Squares algorithm to turn the original FIR filter into an adaptive filter. Using our newly developed adaptive filter, we identify the functionality of several black box systems. Then, as a demonstration of a real-world use for adaptive filters, we use our adaptive filter to perform real-time noise cancelling.

Background

Least Mean Squares

Our adaptive filter is implemented using the Least Mean Squares (LMS) algorithm. The goal of the LMS algorithm is to minimize the error signal being fed back into the adaptive filter. This is accomplished by updating the filter coefficients according to the current error signal. We use a scalar μ to specify the scaling factor that controls how big of a change is made in the filter coefficient. The bigger the step, the less time it takes for the adaptive filter to converge such that the error signal is minimized. However, if the step size is too large, we run the risk of causing our system to become unstable. Equation 1 defines the LMS filter algorithm where b[k,n] is the k^{th} filter coefficient used when sample x[n] has just been received. μ is the scaling factor, and e[n] is the difference between d[n] and y[n] where d[n] is the output of the unknown signal and y[n] is the output of our adaptive filter.

$$b[k, n+1] = b[k, n] - \mu e[n] x[n - k] \qquad k = 0, 1, \ldots, N-1$$

Equation 1: Least Mean Squares Filter Algorithm

Methods

The FIR filter we had implemented in Lab 2 was a 73^{rd} order bandpass filter. To convert it to an adaptive filter, we implemented the LMS algorithm to update the filter coefficients according to the algorithm outlined above. After adding the 4-5 lines of code necessary to implement the algorithm, we sought to do a very basic verification that our additional code didn't break our FIR filter. To accomplish this, we set μ to 0 and left the filter coefficients as their previous values used to implement a bandpass filter. When attempting to run this new code, we discovered that our additional computations resulted in our ISR exceeding the real-time requirements. To fix this issue, we enabled compiler optimization, which brought our execution cycles significantly under the ~5100 cycle limit.

Next, in order to verify that our adaptive filter functioned properly, we fabricated inputs simulating the system identification scenario for a known system. For our known system, we used the bandpass filter from labs 2 through 4. To simulate this, we used MATLAB and generated white noise as our x[n] signal. We then generated the appropriate FIR filter coefficients and used MATLAB's filter function to filter x[n]. The output of our filter became d[n]. We set x[n] and d[n] as the left and right channels of a WAV file so we could output the signals to the DSP board.

To perform the test, we connected the audio out from the PC to the line in on the DSP board and the line out on the DSP board to the microphone input of the PC. This setup allowed us to send our custom signals to the DSP and record the DSP output to verify proper functionality of our adaptive filter.

When we first started testing our adaptive filter, we chose a very small value of μ to ensure that our adaptive filter would converge. We decided that since there is no time limit on performing However, even with this design decision, we observed in the watch window that our filter coefficients were exploding after running for a few seconds. We determined that this was due to a sign reversal in our implemented algorithm. In the example provided in class, the scaled error is subtracted from the current filter coefficients. However, this requires that μ be negative. We had made our μ positive and therefore decided to alter our code to add the scaled error to the filter coefficients instead of subtracting. This change in the algorithm can be seen in Equation 2.

$$b[k, n+1] = b[k, n] + \mu e[n] x[n - k] \qquad k = 0, 1, \ldots, N-1$$

Equation 2: Modified LMS Algorithm

Problem Solution

System Identification

Our final implementation of the LMS algorithm followed the process sequence outlined in Figure 1. The inputs to our system identification module are the primary input signal x[n] (left channel) and the output of the unknown system d[n] (right channel). Using our FIR filter and the LMS algorithm, we generate our outputs e[n] and y[n]. When our adaptive filter has converged, e[n] converges to 0 and y[n] matches d[n].

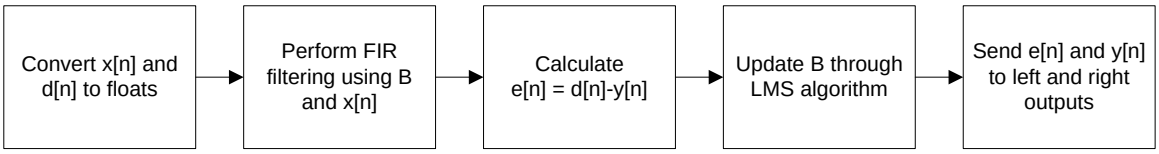

Figure 1: LMS Algorithm Implementation Flowchart

Noise Cancellation

The implementation to perform noise cancellation is almost identical to the configuration used for system identification. The one difference is that the input signals for noise cancellation are switched. Now, the noise is coming in on the right channel and the noisy music is coming in on the left channel. The sequence of operations performed on the inputs is unchanged from our system identification implementation.

Results

Here we present the results of our efforts to implement the Least Mean Squares algorithm on the C6713 DSK.

Part 1 – System Identification

In this part, we discuss the results of the tests performed on the LMS system identification algorithm which was implemented as described above.

Maximum Value of μ

In order to get a working algorithm an appropriate value of the filter update step-size, μ, must first be chosen. Iterative testing of values for μ resulted in picking a value of 1e-10 for accurate convergence of the algorithm. The value of μ can be increased to about 1e-7 at which point any larger values saturate the size of the floating point container however even at 1e-7 the coefficients are wildly unstable resulting in a poor if completely inaccurate filter impulse response. The value of 1e-10 was chosen for these tests because it resulted in a very stable convergence that did not take unreasonably long to converge.

Functionality Testing With a Known System

We first tested the system identification algorithm with a known system we designed. A 72nd order elliptic FIR band-pass filter was designed and applied to a white noise signal. The unfiltered and filtered white noise was then applied to the left and right input to the DSK running the system identification algorithm. After giving the algorithm approximately 10 seconds to converge, the output was recorded

with MATLAB and the adapted filter coefficients saved from the C6713 memory. Figure 2 shows the frequency response of the filter designed in MATLAB and used as the system to be identified along with the response of the filter resulting from the LMS algorithm. These two responses match up near perfectly (except for a gain offset necessary because of scaling in the computer and DSK codecs) indicating successful implementation of an LMS algorithm that converges properly. Figure 3 shows the error values computed for each sample by the LMS algorithm as output from the DSK. These values are very small compared to the filter signal output further indicating correct convergence of the filter.

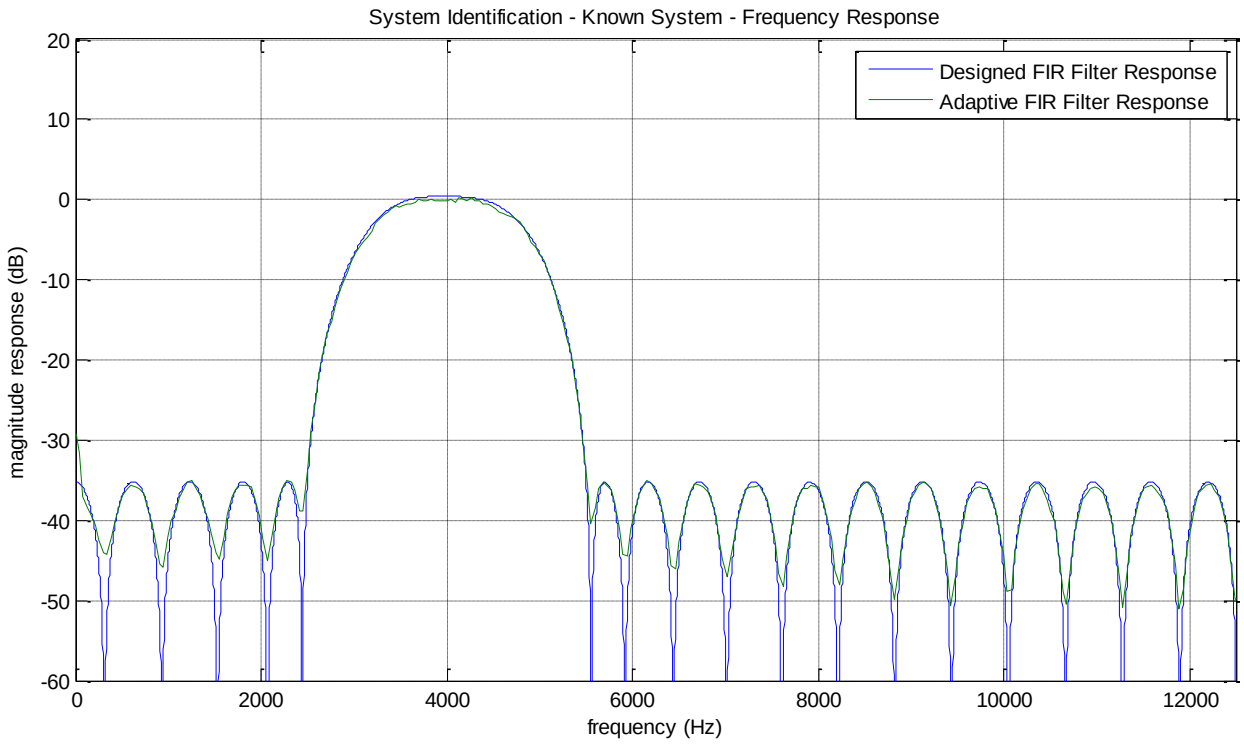

Figure 2: Designed and Adapted Frequency Responses (top) and Impulse Responses (bottom) for a known system applied to the LMS system identification algorithm.

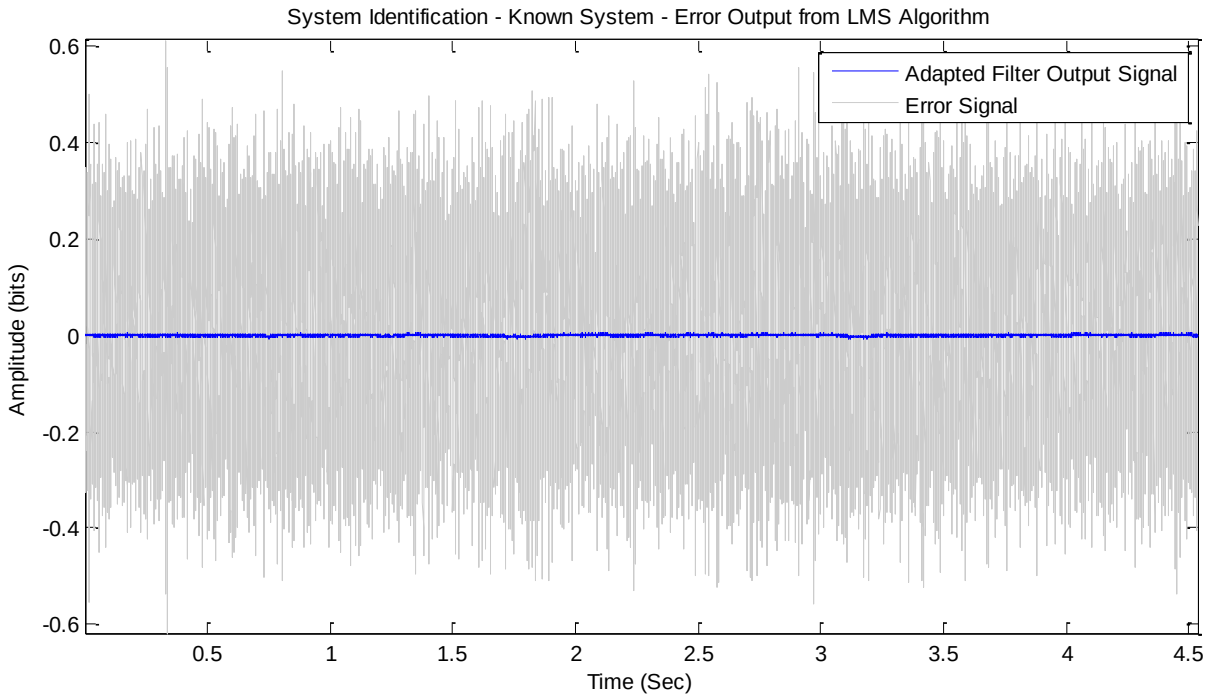

Figure 3: Error value calculated for each sample as used in the LMS algorithm. This is shown against the adapted filter output for size comparison.

Since the system being identified was known to be of 72nd order, an adaptive filter of the same order was used for this test and the coefficients produced by MATLAB can be seen in plotted in Figure 4. After the LMS algorithm was allowed to converge with this system as the input, the coefficients produced by the LMS algorithm were saved from the C6713 memory and are shown alongside the coefficients produced by MATLAB in Figure 4. This further confirms the correct convergence of the LMS algorithm on the impulse response of the input system as all the coefficients match up very closely, if not perfectly throughout the 72nd order filter. These slight discrepancies can be attributed to quantization noise and other analog noise and imperfections in the two digital to analog converters and two analog to digital converters used in the signal chain of this test as well as the slight imperfections of the audio player used when the input signals were repeated.

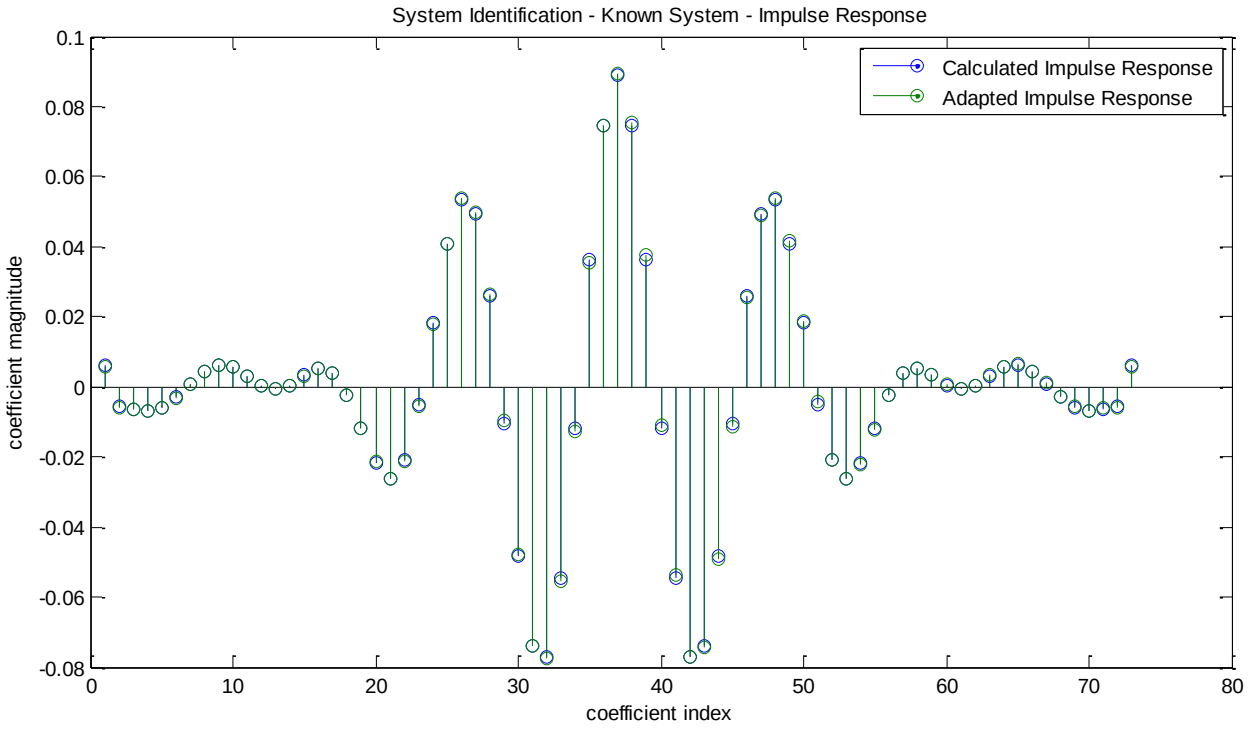

Figure 4: A stem plot of the impulse responses of the known system for both the MATLAB generated filter coefficients and the adapted filter coefficients.

Identification of Unknown System 1

Figure 5 shows the frequency response of the filter adapted by the LMS algorithm to the input of unknown system 1 provided with the lab instructions. The similarity between filter lengths 10 and 50 compared to the filter length 9 indicates that the unknown system is of order 10. From the order 10 response, we can conclude that this system likely consists of a 3.5kHz low-pass filter with a particularly low rejection around 10kHz. Figure 6 further shows that the unknown system is likely of order 10 because all coefficients above 10 are essentially zero.

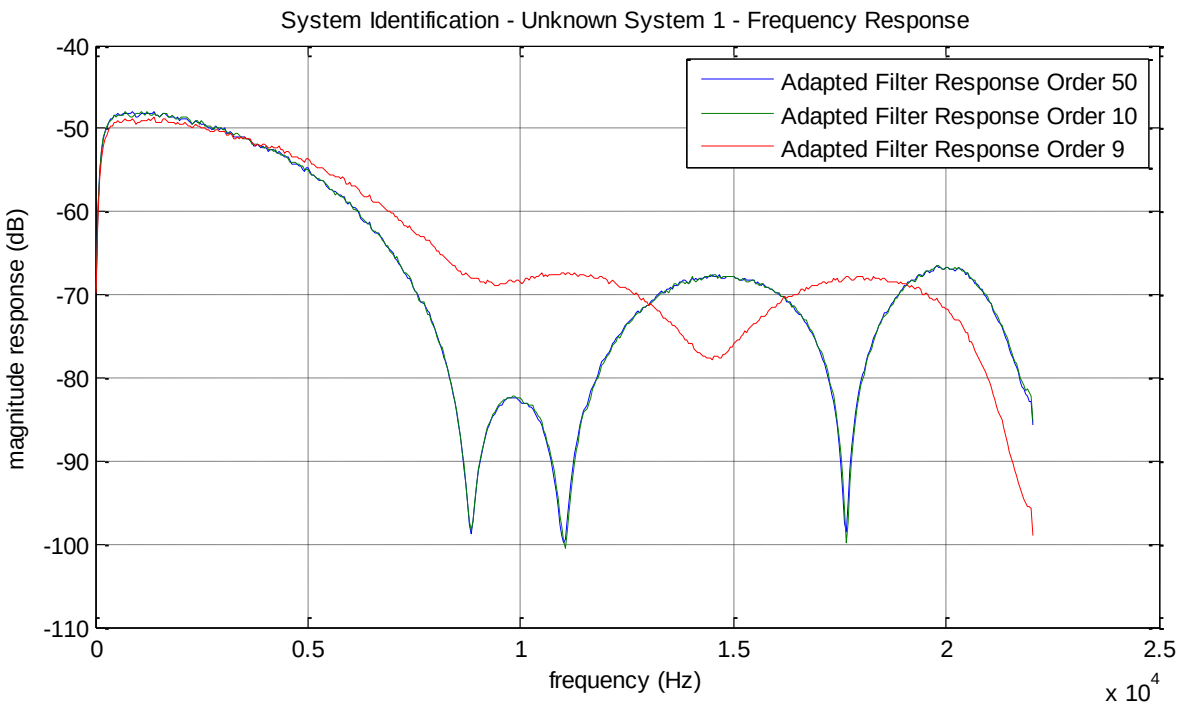

Figure 5: Frequency response of the filter adapted to unknown system 1 for several different filter lengths.

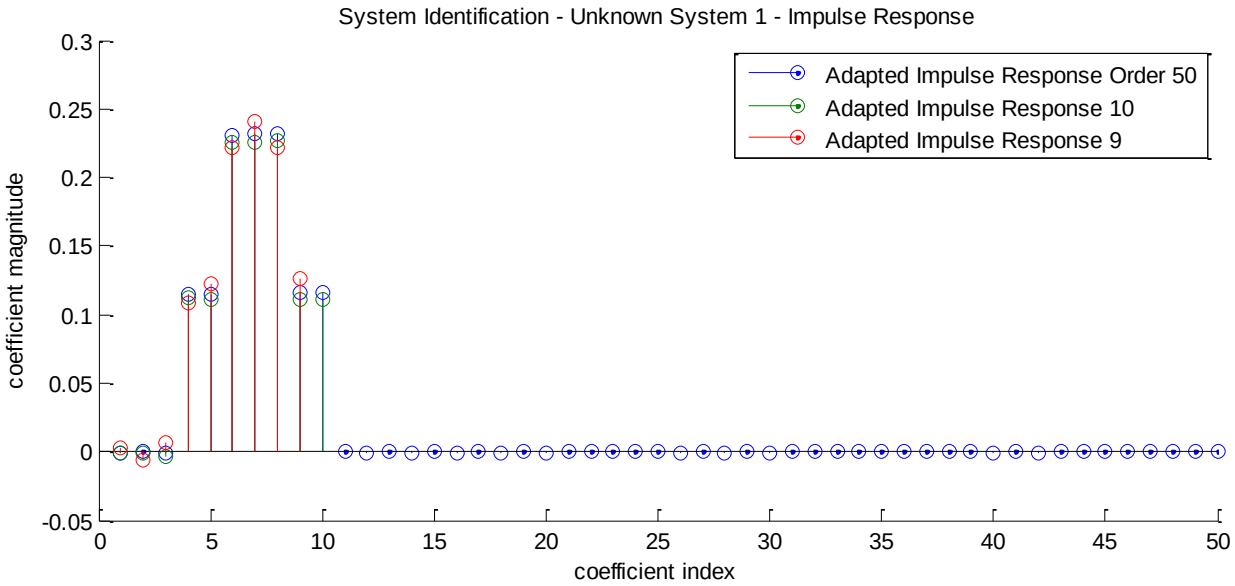

Figure 6: Impulse response of adapted filter for unknown system 1 shown for several different filter orders.

Identification of Unknown System 2

The second unknown system provided with the laboratory assignment was tested with the LMS algorithm and the adapted frequency response is shown in Figure 7. From this, we can see that the unknown system's order is likely near 64 and the unknown system consists of notch filters at about

4.5kHz and 9.5kHz. Figure 8 further confirms the filter order of about 64 by showing the impulse response for the three adaptive filter orders that were tested.

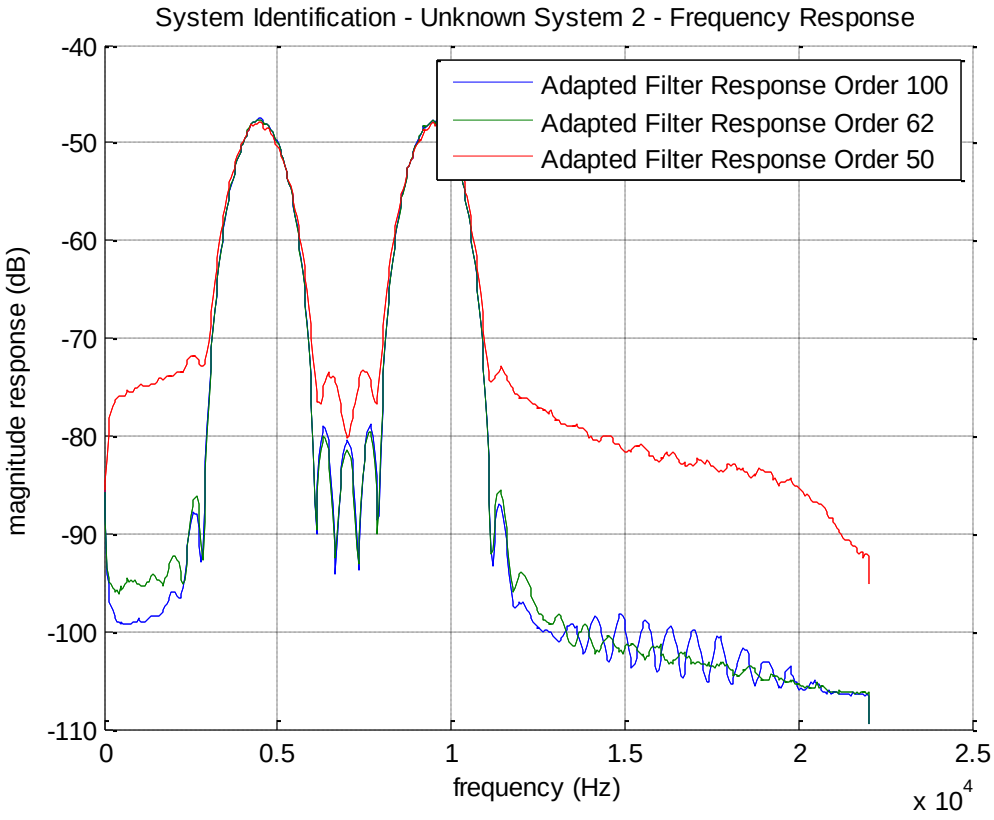

Figure 7: Frequency Response of Unknown System 2 for adaptive filter orders 50, 62, and 100.

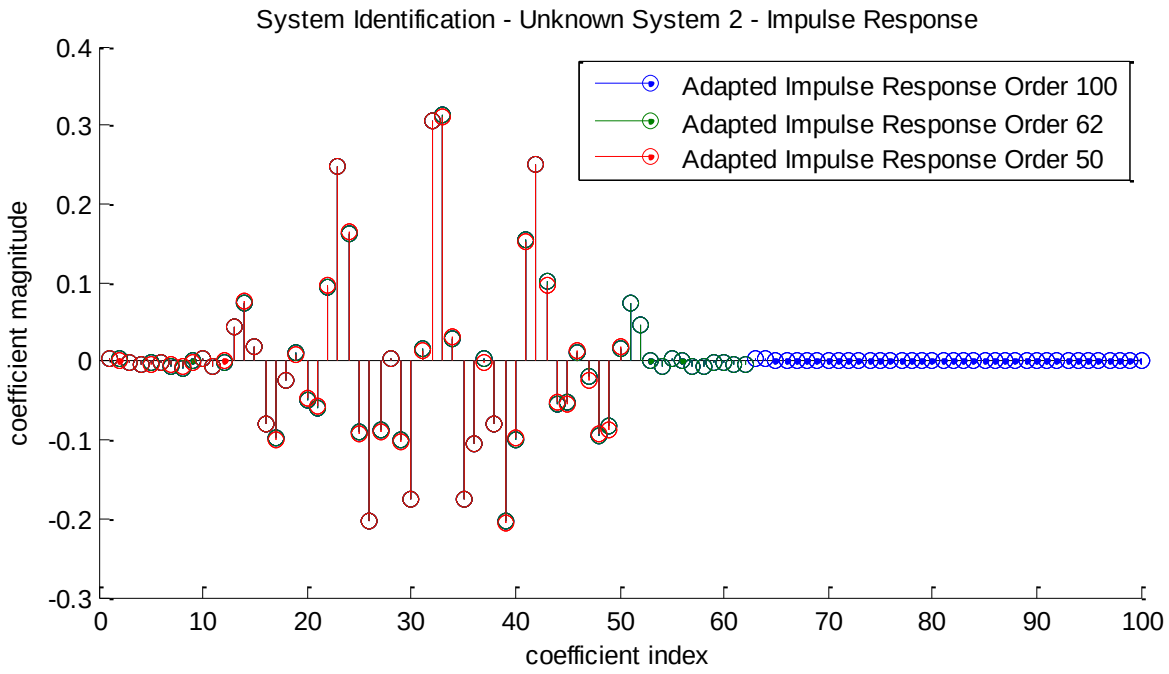

Figure 8: Impulse responses for several adaptations to unknown system 2.

Identification of Unknown System 3

The last unknown system provided for identification produced the frequency response shown in Figure 9. As is indicated by the large error signal output by LMS algorithm for unknown system 3, as can be seen in Figure 10, the LMS algorithm did not completely converge system did not seem to completely converge for this unknown system. The largest adaptive FIR filter we could implement on the DSK is of order 150, but this was not enough to cause the filter to converge. This may be either because the filter in the unknown system is of order larger than 150, or possibly that the filter is an Infinite Impulse Response filter, which would require an infinite number of FIR filter coefficients to replicate.

Figure 9: Frequency response for several orders of filter adapted to unknown system 3.

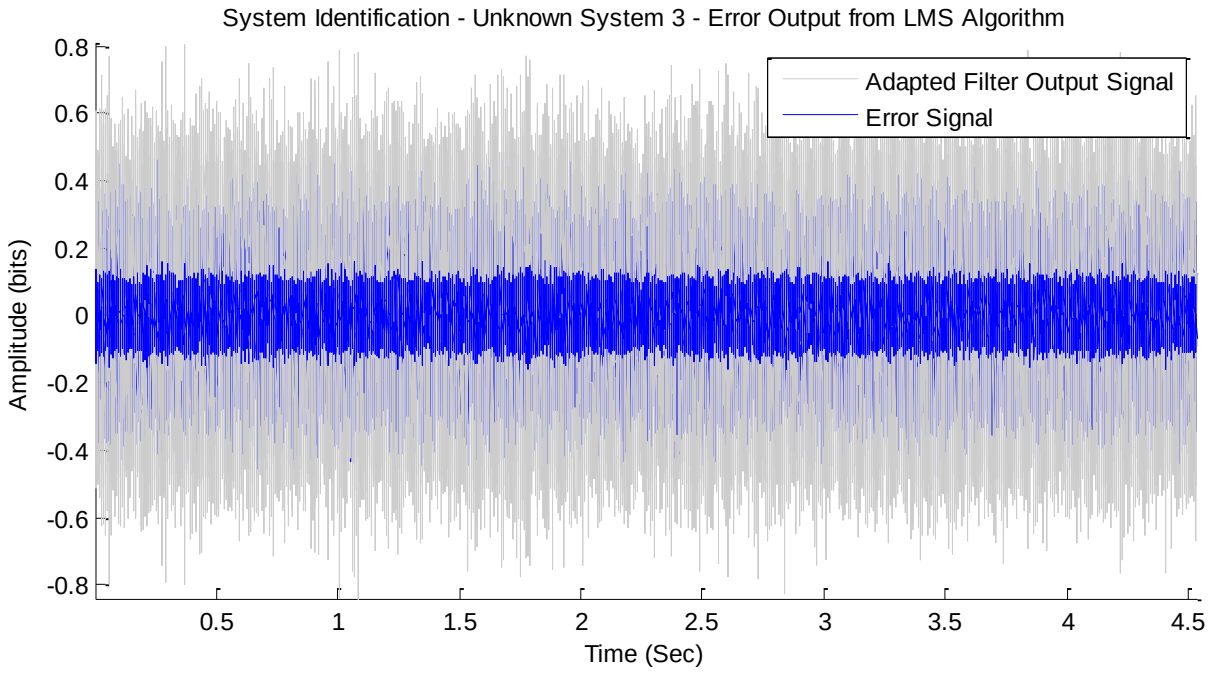

Figure 10: Error between adapted filter output and unknown system input for each samples compared to the Adapted Filter Output.

Part 2 – Noise Cancellation

The second section of this laboratory assignment is concerned with implementing noise cancelling algorithms with the same Least Mean Squares algorithm used for the system identification purpose. An example signal and overlaid noise were provided with this laboratory assignment that we played through the noise cancelling algorithm running on the DSK to confirm its proper functioning. The example signal with overlaid noise consists of a Pink Floyd song (the signal) overlaid with band-passed white noise with the pass band jumping upwards every few seconds. The noise cancellation algorithm had to re-adjust for each sudden change of the noise system. Figure 11 shows a spectrogram of the input signal with overlaid noise as it was input to the left channel of the DSK running the noise cancellation algorithm. The noise is the dominant sound in this signal, and almost completely obscures the music when heard.

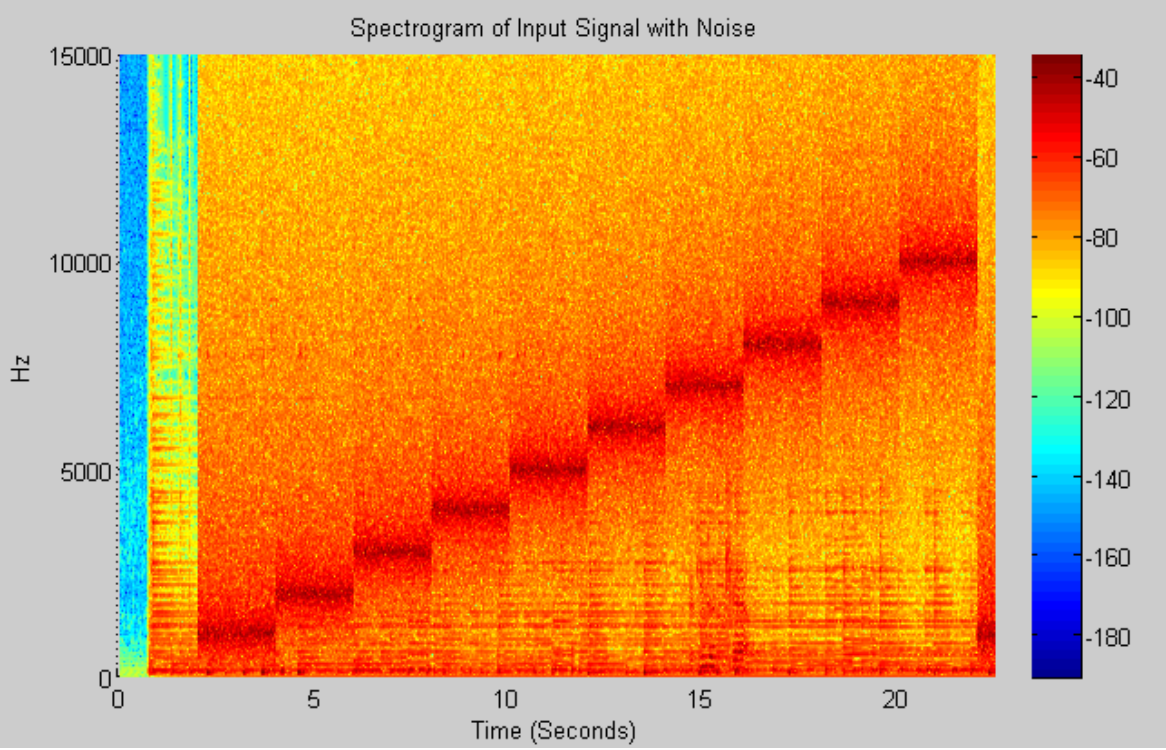

Figure 11: Signal and noise input to the noise cancellation algorithm. The noise can be seen as the staircase of intensity, while the signal is mostly obscured by the noise, except for in the lower right corner.

Figure 12 shows the right input to the noise cancellation algorithm, the pure noise that was filtered before overlaying on the desired signal. The algorithm uses this signal as the correlated input to be filtered before subtraction from the input signal with noise.

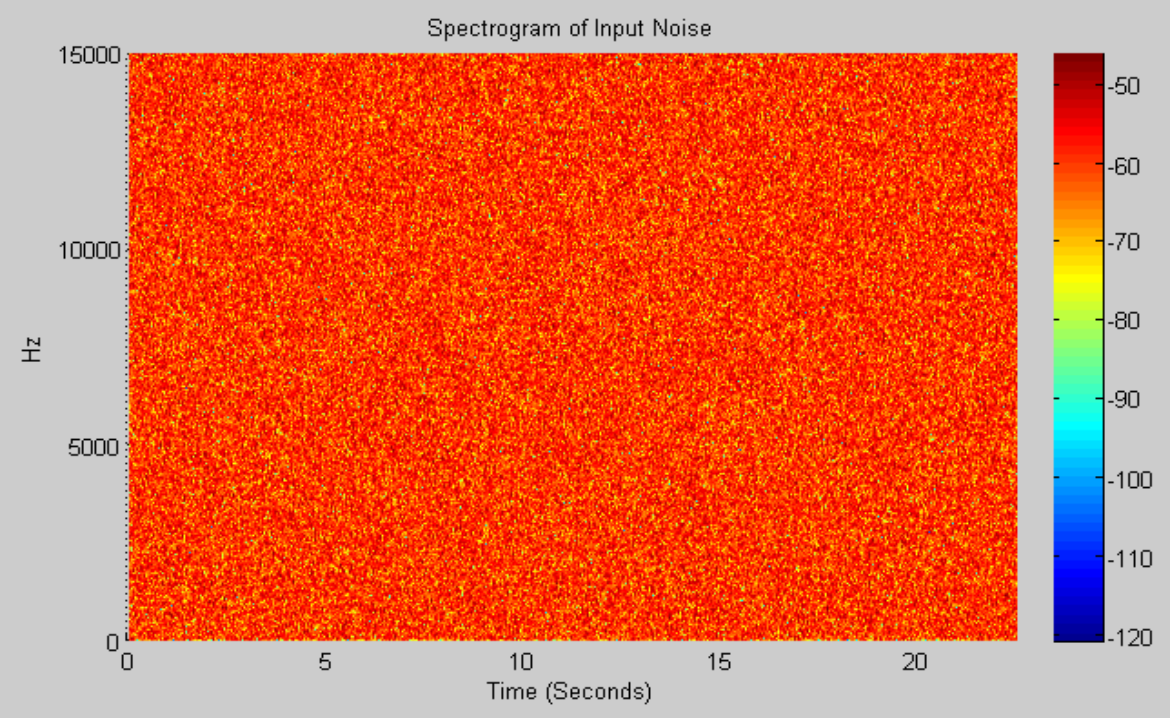

Figure 12: Spectrogram of pure noise input to noise cancellation algorithm.

Figure 13 shows the noise cancelled output of the working noise cancellation algorithm using an LMS update step size, μ, of 0.01[1] and an adapted filter order of 150. While the noise has not been completely cancelled, even after the algorithm has converged, the signal amplitude over the noise amplitude is significantly better – the music signal is now visible in the top left corner, unlike in the spectrogram of Figure 11 and the noise cancelled output sounds significantly better to the ear as well.

[1] This value of μ is significantly larger than the one used in Part 1 because a different scaling was applied to the input and output signals for part 2 because the value of μ was running up against the limits of the Eclipse watch window.

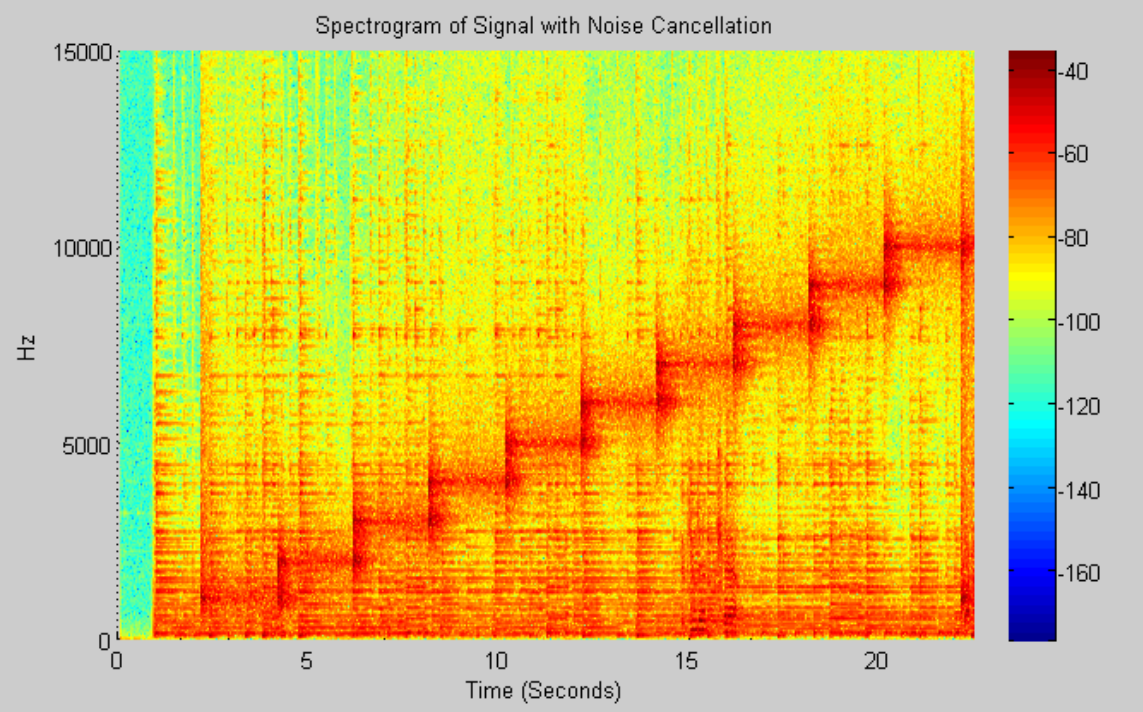

Figure 13: Noise cancelled signal with an LMS update step size, μ, of 0.01 and an adapted filter order 150.

Both the LMS update step size and the adapted filter order was adjusted to observe the effect they had on the noise cancellation. Figure 14 shows noise canceled signal with an adapted filter order of 50 instead of the order 150 used in Figure 13. This decreases the performance of the noise cancellation both visibly and audibly.

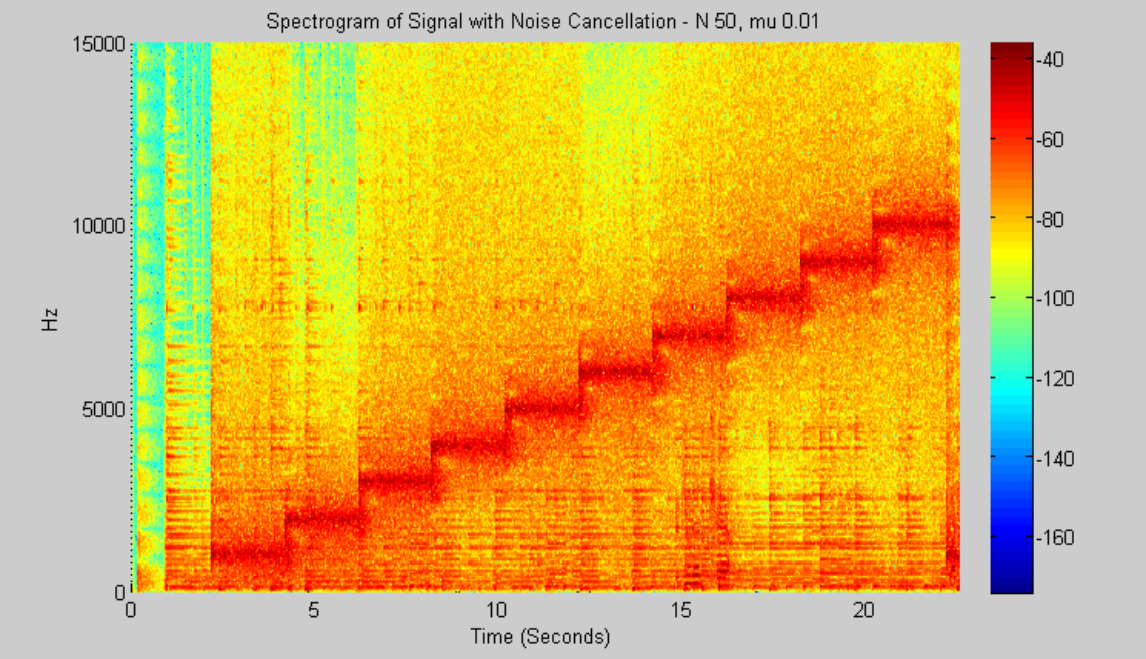

Figure 14: Noise cancelled signal with an LMS update step size of 0.01 and an adapted filter order 50

The best value of step size was determined experimentally to be $\mu = 0.01$. Larger values produced too much distortion of the desired signal, and smaller values did not finish converging before the noise changed. Figure 15 shows an example where the update size is too small, resulting in residual filtering of parts of the signal that no longer have noise applied. Figure 16 shows the opposite end of the spectrum, with a step size that is too large. The adaptive filter is still relatively stable; quickly removing any noise, however when listening to the output of the algorithm with these parameters it can be seen that the algorithm is starting to distort the desired music signal.

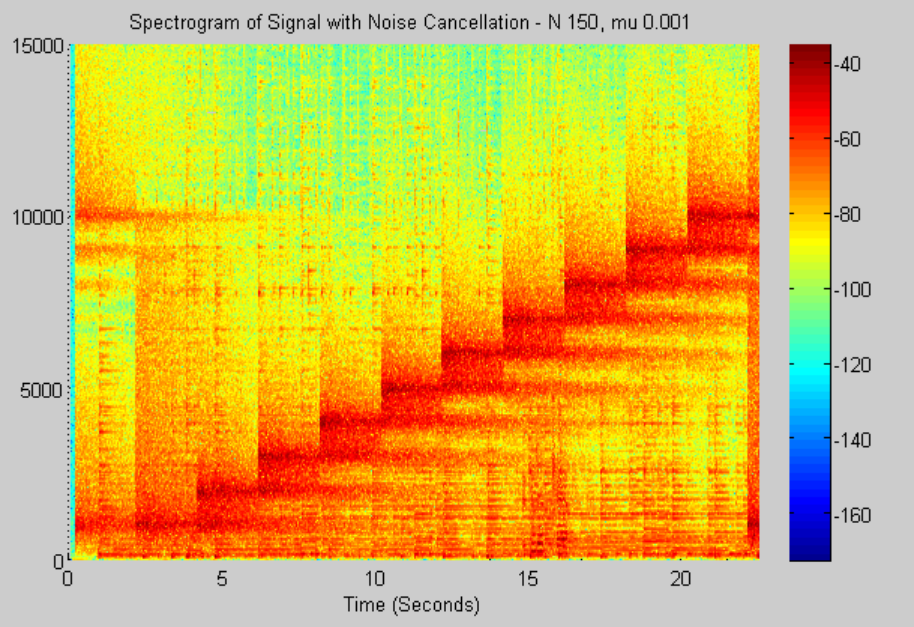

Figure 15: Spectrum of noise cancelled signal for an update step size of 0.001. Here the filter does not adapt fast enough to keep up with the changing noise.

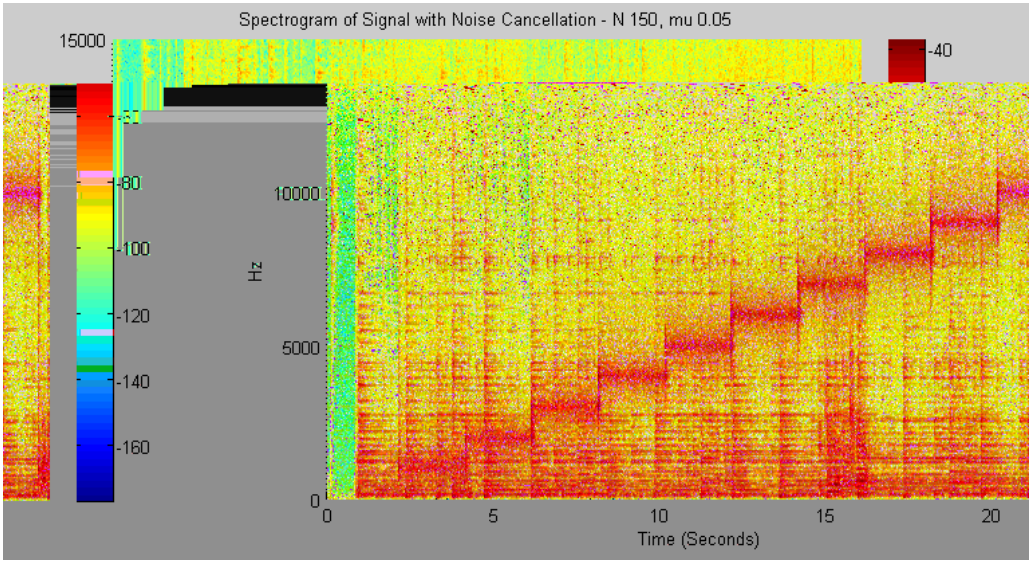

Figure 16: Spectrum of noise cancelled signal for an update step size of 0.05. It is not immediately obvious from this spectrogram, but the adaptive filter with this step size was significantly distorting the desired signal.

Conclusion

In this laboratory assignment, we successfully implemented the Least Mean Squares algorithm for both the purpose of system identification and noise cancellation. These two related applications of signal processing are used widely in the communications systems we use on a daily basis to improve communication quality while allowing for lower power and faster transmissions. While we were unable to go into detail about the proof and derivation of the LMS algorithm, we successfully used it to accomplish the required tasks, indicating its effectiveness for a wide range of applications. Overall this lab, and those that came before have laid a foundation for the development of complex signal processing systems that can provide a wide range of functions in the development of signal based systems.