System Identification and Digital Signal Processing

Yousef Shamasneh 1190300, Yazan AbdAlmutee 1190145 and Saif Battah 1170986, dept. Electrical and Computer Engineering

# Intrtoduction

System identification is a fundamental step in digital signal processing (DSP) for modeling and controlling dynamic systems. It involves using input-output measurements of a system to estimate its parameters and create a mathematical model of the system. This model can be used to predict the behavior of the system under different input signals, analyze the system's stability and performance, and design control algorithms to achieve desired system properties. The process of system identification typically consists of several stages:

The choice of methods and algorithms used in each stage of system identification can have a significant impact on the accuracy and reliability of the estimated model, so it is important to choose appropriate techniques based on the properties of the system and the available data.

# Problem Specification

System identification in Digital Signal Processing (DSP) is the process of estimating the parameters of a dynamic system from its inputs and outputs. The goal of system identification is to determine the mathematical model that best represents the relationship between the inputs and outputs of the system. The model can be used to predict the response of the system to new inputs, control the system, or analyze its behavior. In DSP, system identification can be performed using various techniques, such as least-squares, Kalman filtering, and subspace methods. Two popular algorithms for system identification in DSP are the Least Mean Squares (LMS) algorithm and the Normalized Least Mean Squares (NLMS) algorithm.

The LMS algorithm is an adaptive filtering technique that adjusts the parameters of the model to minimize the mean squared error between the estimated outputs and the actual outputs of the system. It uses a gradient descent approach to update the parameters, where the step size is proportional to the error. The NLMS algorithm is similar to the LMS algorithm, but it normalizes the step size to mitigate the effects of large input signals. This allows the algorithm to converge more quickly and accurately, especially when the input signals are highly correlated or have large dynamic ranges.

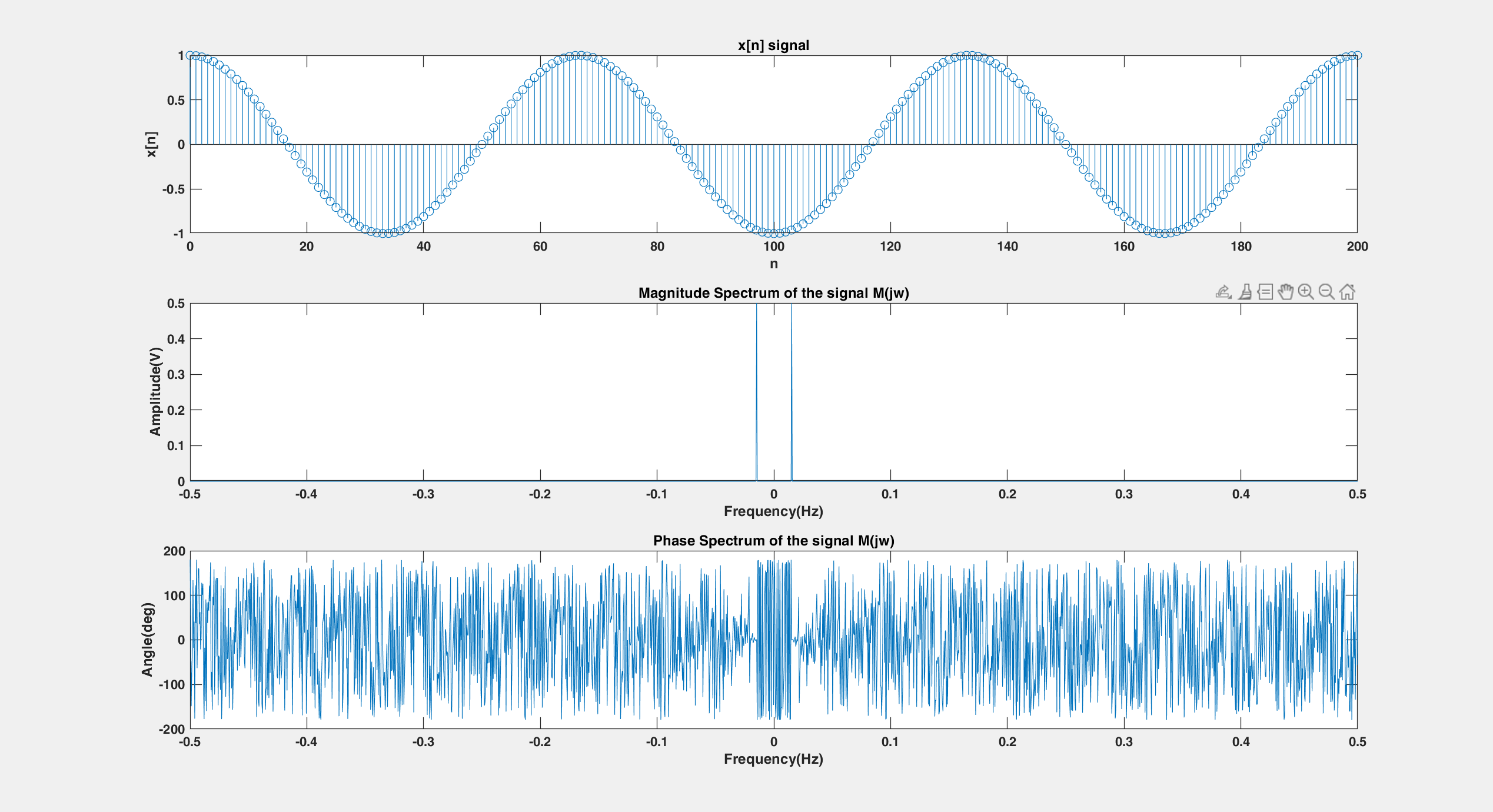
Both the LMS and NLMS algorithms can be used for system identification in various applications, including noise reduction, channel equalization, and system modeling. The choice between the LMS and NLMS algorithm depends on the specific requirements of the application and the trade-off between computational complexity and convergence performance.

In this project, both LMS and NLMS are implemented using MATLAB in order to analyze the performance of each algorithm in the process of system identification and exploring the differences between both of them. Error curves are used to compare between the two different algorithms. Moreover, Additive White Gaussian Noise (AWGN) are added to the input signals with different SNR Values to determine the effect of adding noises on the behavior of the algorithms while predicting the input-output relationship of the system.

# Data

The input signal that is used to feed the model is a sinusoidal signal with N = 2000 sample points.

The signal is generated and plotted based on the provided equation and its spectrum is plotted as well.

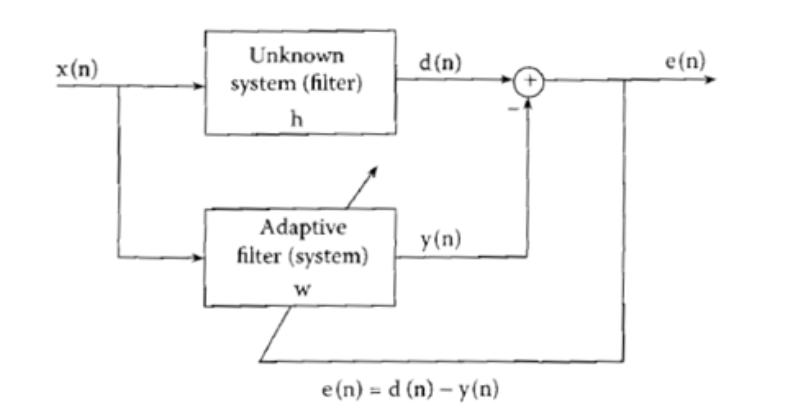


1. Input Signal and its Magnitude and Phase Spectra.

Since the input signal x[n] is a sinusoidal signal, its magnitude spectrum is 2 delta functions at and . There are 2000 samples, then the frequency is

# Evaluation Creiteria

Error signals e[n] will be used to assess the performance of both LMS and NLMS algorithms. e[n] is defined as the difference between the output of the unknown system d[n] and the output of the adaptive filter y[n].



1. System Block Diagram.

Moreover, another metric is used to assess the performance of the implemented algorithms. J vs iteration step is plotted, where J is defined as the squared value of the error.

In addition, another curve that is based on the J[n] curve is plotted. The other curve is defined as:

# Approach

There are several steps to follow in this project in order to achieve the objectives.

### The input signal x[n] is generated.

### The input signal and its spectrum are plotted.

### The respone of the FIR Filter is obtained.

#### LMS algorithm is implmented as follows:

#### Initialize the filter coefficients: The filtercoefficients

#### are initialized with small random values.

#### Provide input signal: Provide the input signal to the

#### system, and measure the corresponding output signal.

#### Calculate error signal: Calculate the error signal by

#### subtracting the estimated output signal from the actual output signal.

#### Update filter coefficients: The filter coefficients are

#### updated based on the error signal and the input signal. The update rule for the LMS algorithm is given by:

#### w(k+1) = w(k) + μ \* e(k) \* x(k)

#### where w(k) is the filter coefficients at time k, μ is the step size, e(k) is the error signal at time k, and x(k) is the input signal at time k.

#### Repeat steps 2 to 4: Repeat steps 2 to 4 for N

#### iterations, where N is the number of iterations required to achieve convergence.

#### Estimation of System Transfer Function: The final

#### filter coefficients obtained after convergence can be used to estimate the transfer function of the system.

#### The step size (μ) is an important parameter that

#### needs to be carefully chosen to avoid overshooting or convergence to a local minimum.

### Error curves and Frequency respone of the obtained

### filter are plotted.

### MATLAB Built-in function is used to make sure that

### the obtained results from the implmented function are correct.

### Different μ values are tried and AWGN is added to

### input signal.

### Ensemble averaging over different iterations is tried

### in order to analyze its effect on the error curve.

### NLMS algorithm is implmented as follows:

#### Initialize the filter coefficients: The NLMS

#### algorithm requires initial values for the filter coefficients. The coefficients can be initialized to zero or any other suitable value.

#### Input data: The input signal is the input data to the

#### system that needs to be identified. The input signal is passed through the filter to produce the filtered output.

#### Filtered output: The filtered output is obtained by

#### multiplying the input signal with the filter coefficients and summing the products.

#### Error calculation: The error between the desired

#### output and the filtered output is calculated. The desired output is usually a known or desired signal, or the desired response of the system.

#### Update coefficients: The filter coefficients are

#### updated based on the error calculated in the previous step. The NLMS algorithm uses the input signal, filtered output, and error to calculate the new coefficients.

#### The update rule for the coefficients h[n] is given by:

*h[n] = h[n] +* μ *\* e[n] \* x[n] / ||x[n]||^2*

#### where:

#### e[n] is the error between the actual output y[n] and the estimated output y\_est[n], defined as

#### e[n] = y[n] - y\_est[n].

#### μ is the step size, which determines the amount of adjustment made to the coefficients h[n] in each iteration. It is a positive scalar value that should be chosen carefully to ensure that the algorithm converges to the optimal solution.

#### ||x[n]||^2 is the norm of the input vector x[n].

#### Repeat steps 3 to 5: The steps 3 to 5 are repeated for

#### every sample in the input data. The updated coefficients are used to obtain the filtered output for the next sample.

#### Final filter coefficients: After repeating the steps for

#### all samples, the final filter coefficients are obtained. These coefficients can be used to identify the system, or to obtain the desired response of the system.

### Error curves and Frequency respone of the obtained

### Filter using NLMS are plotted.

### MATLAB Built-in function is used to make sure that

### the obtained results from the implmented function are correct.

### Different μ values are tried and AWGN is added to

### input signal.

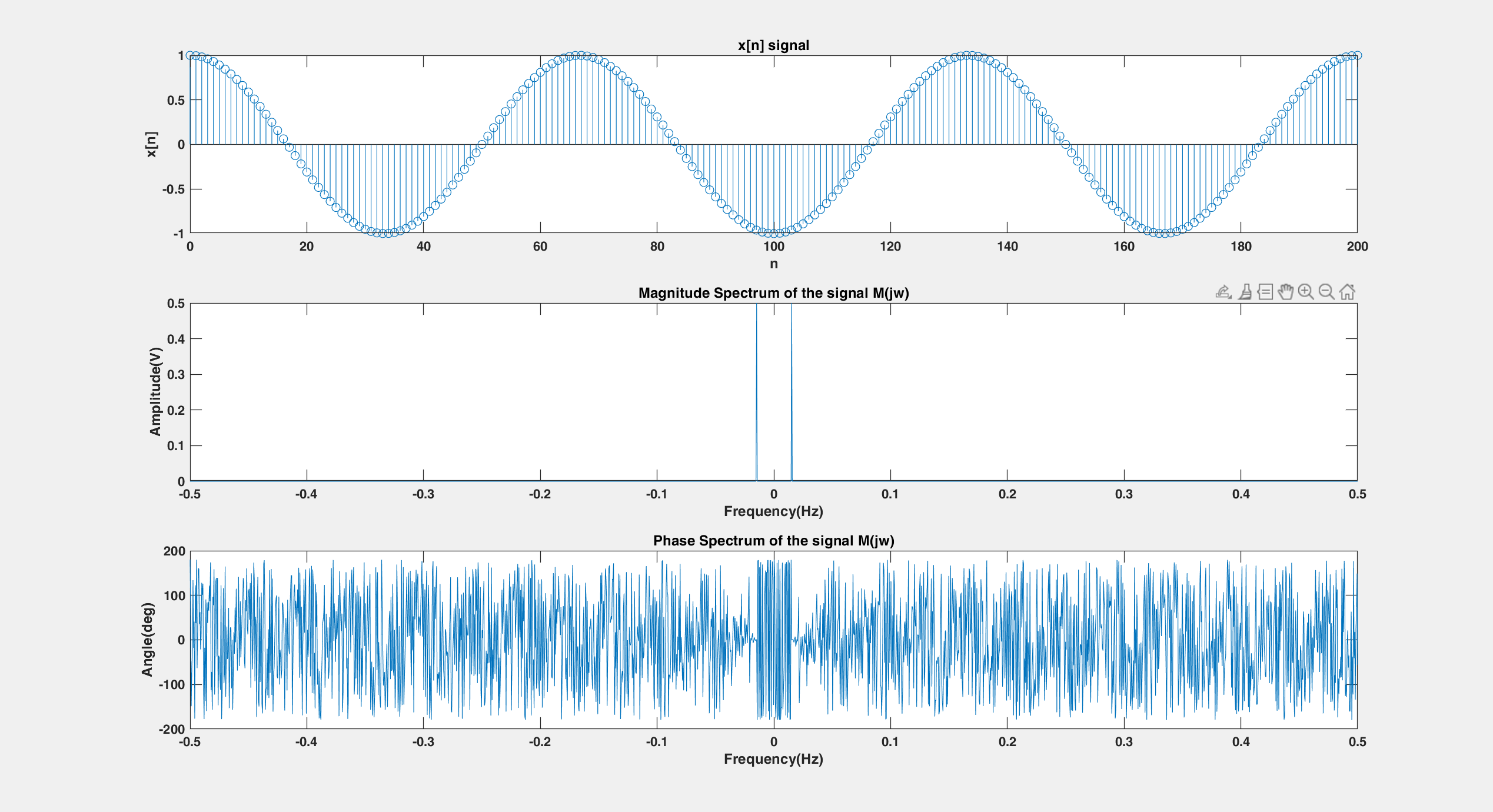
### Ensemble averaging over different iterations is tried

### in order to analyze its effect on the error curve.

#### The key difference between the LMS and NLMS algorithms is that the LMS algorithm updates the coefficients based on the mean squared error between the desired output and the filtered output, while the NLMS algorithm updates the coefficients based on the normalized mean squared error.

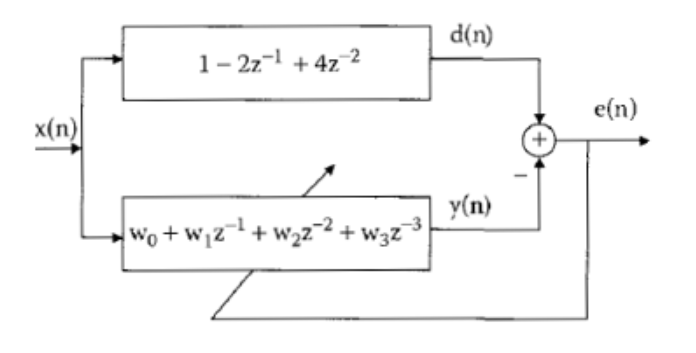
# Results and Analysis

The input signal x[n] is generated and plotted. Then, its magnitude and phase spectrum are plotted as shown in Fig.3.

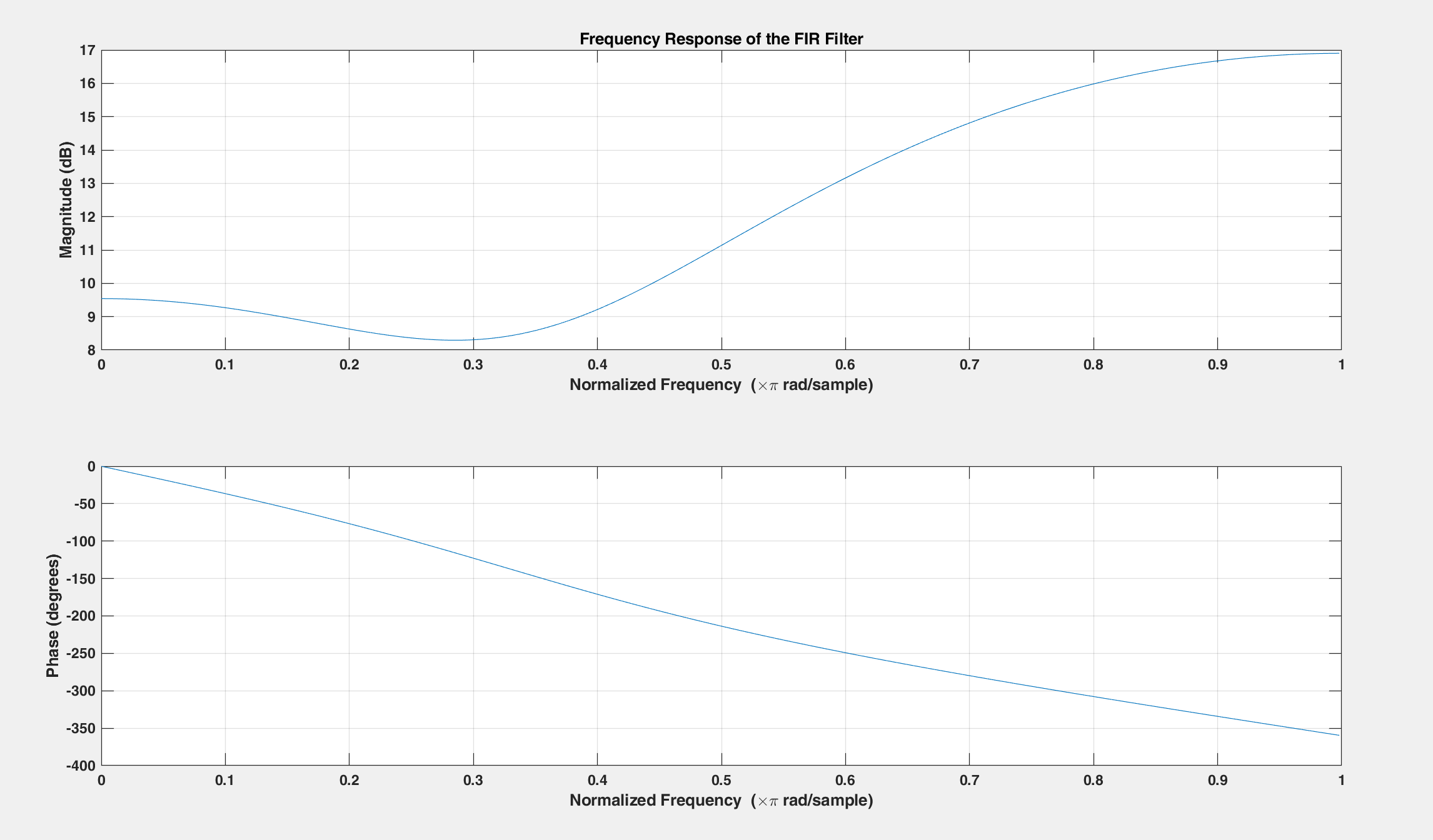


1. Input Signal and its Magnitude and Phase Spectra.

Based on the provided Block Diagram as shown in Fig.4, the response of the FIR Filter is obtained.



1. System Block Diagram.

The coefficients of the FIR filter are [1, -2, 4]. The response of the FIR filter is shown in Fig. 5. It can be noticed that the FIR Filter if a High-Pass-Filter (HPF).

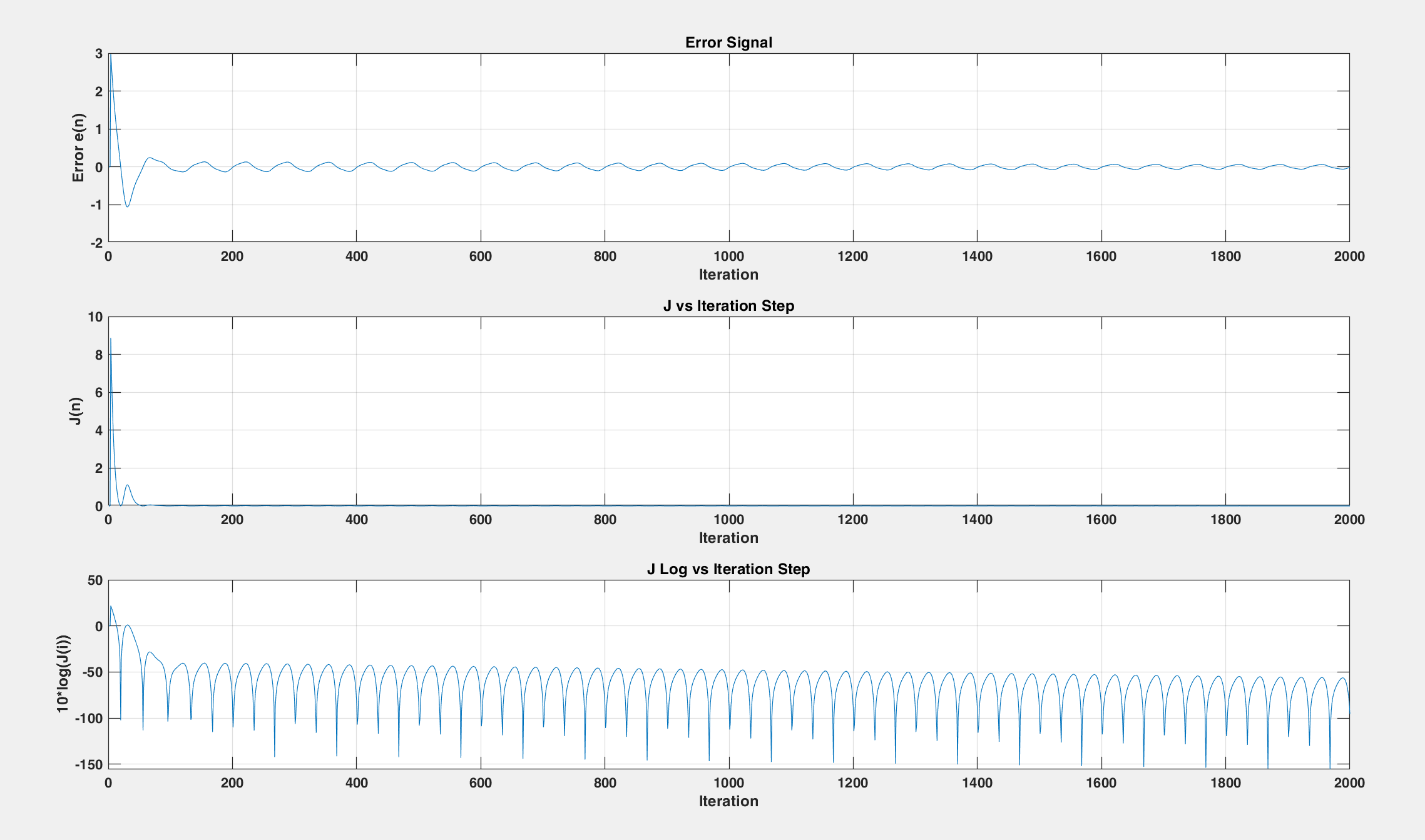
1. FIR Filter response (magnitude and phase)

After implementing LMS algorithm, the coefficients of the adaptive filter are obtained while setting the parameters to be:

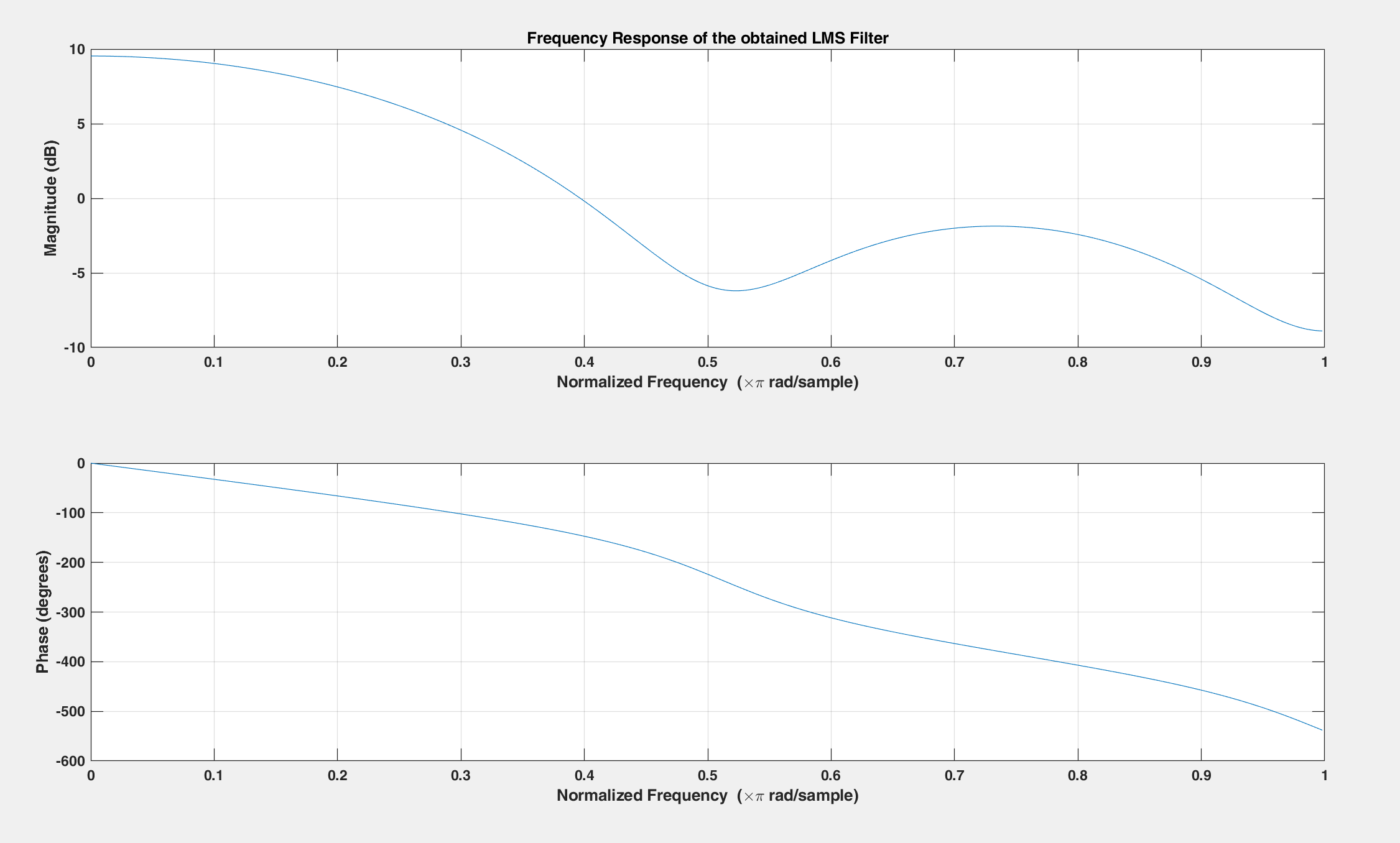
μ = 0.01

M = 4 (Order of the adaptive filter)

Then, the error curves are plotted and J vs iteration is curve is plotted as well.

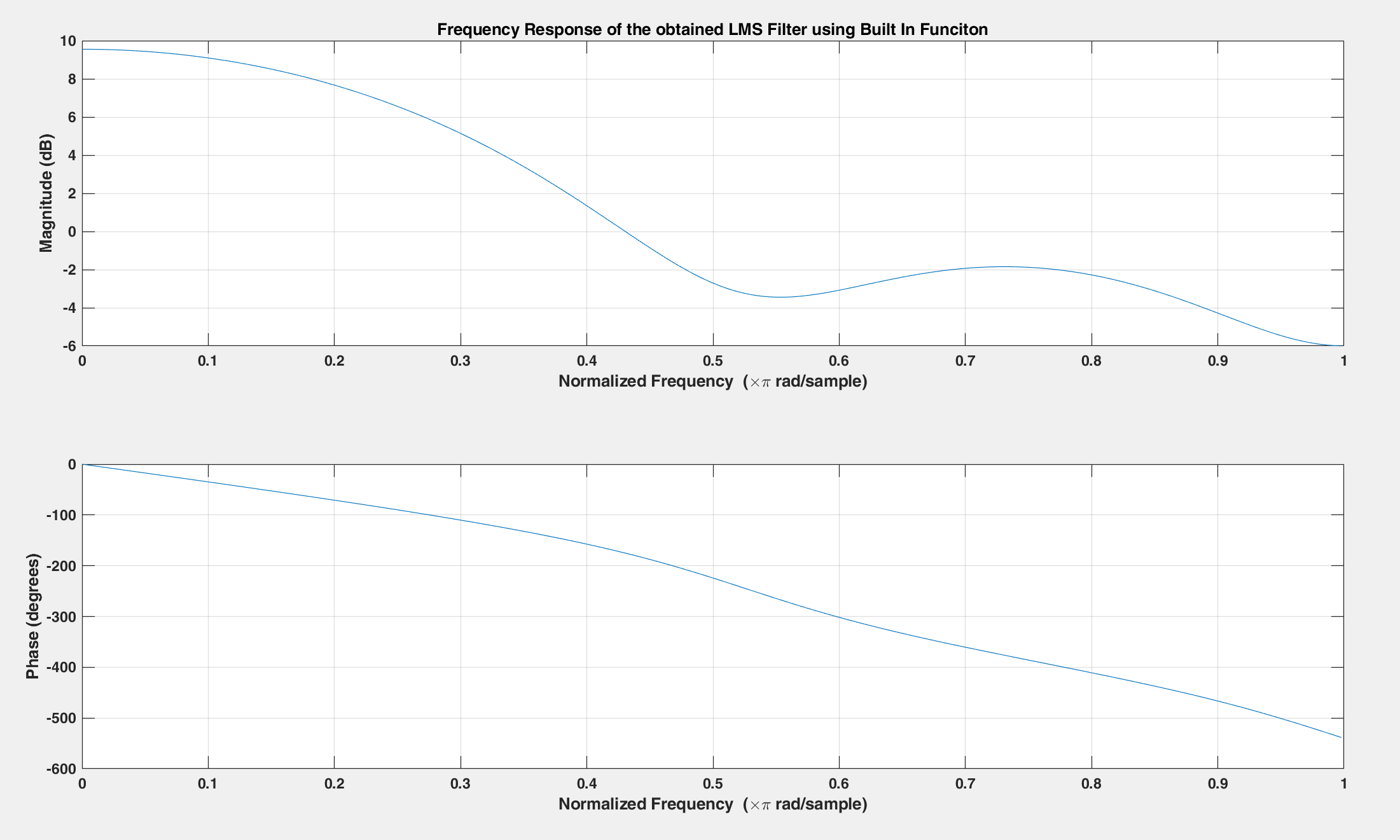


1. Error curve and J vs Iteration curve.

After that, the response of the obtained LMS filter is obtained as shown in Fig. 7. The coefficients of the LMS Filter are [0.476, 0.662, 0.843, 1.016].

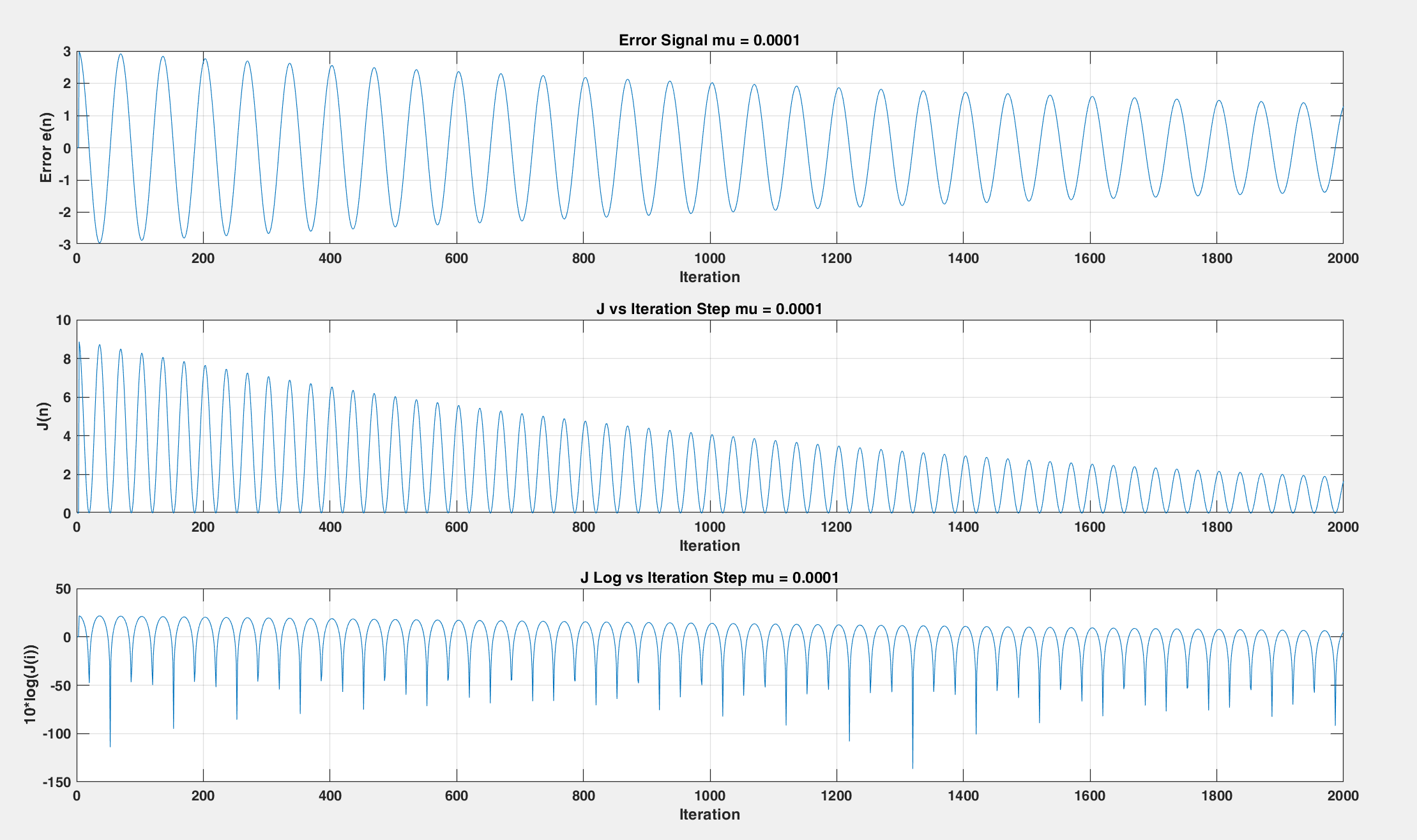
1. Response of LMS obtained filter.

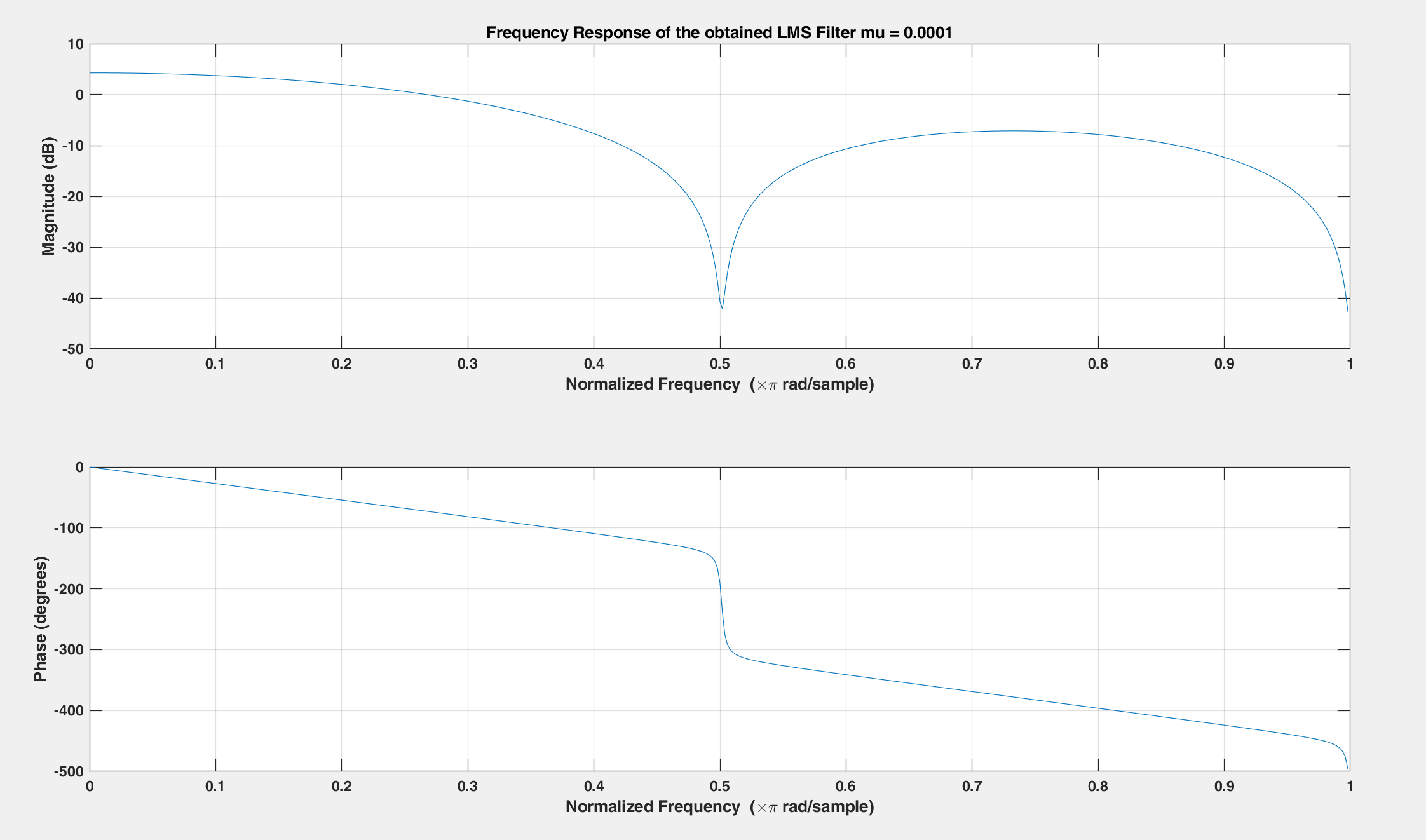
MATLAB Built-in function is used to obtain the coefficients of the LMS filter to confirm that the implemented algorithm is correct. The response of the filter obtained from dsp.LMSFilter Function is shown in Fig.8. The coefficients of the LMS Filter from MATLAB Function are [0.363, 0.621, 0.888, 1.131].

1.  Response of LMS filter from MATLAB Built-in Function.

Accordingly, the obtained results from the implemented algorithm and MATLAB function are the same, then the algorithm is implemented correctly.

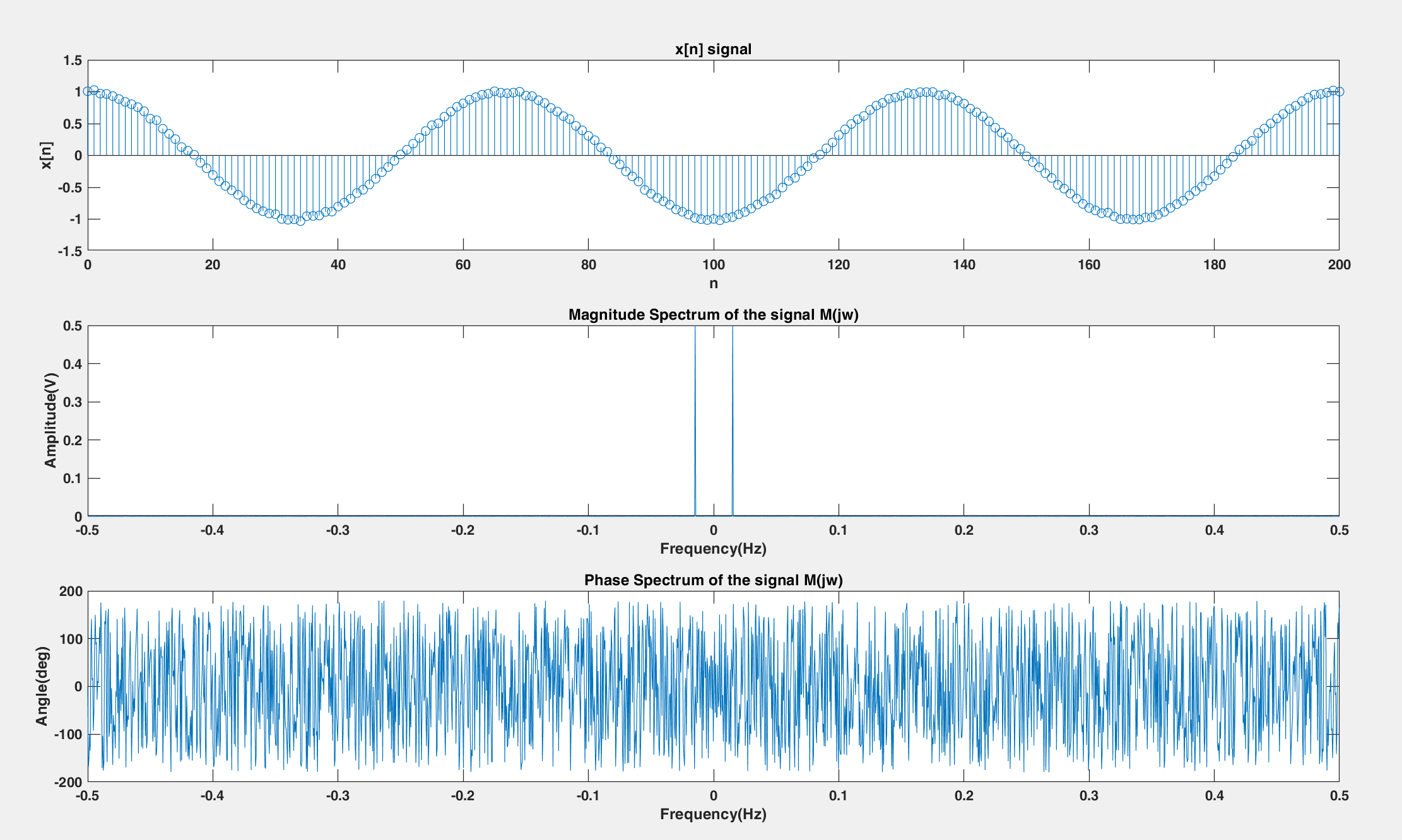
After using the LMS algorithm, it is expected that the adaptive filter coefficients will converge to values that result in the adaptive filter's output being a close approximation of the desired response. As the algorithm continues to update the coefficients, the mean squared error between the desired response and the output of the adaptive filter should decrease. When the mean squared error reaches a minimum, the coefficients have converged to the optimal values for the problem. Then, the error curves are plotted and J vs iteration is curve is plotted as well.

In the next step, different μ value is tried to analyze the effect of decreasing the step size on the learning process to obtain the filter coefficients.

1. Error curve and J vs Iteration curve, where μ = 0.0001.
2. Response of LMS obtained filter , where μ = 0.0001.

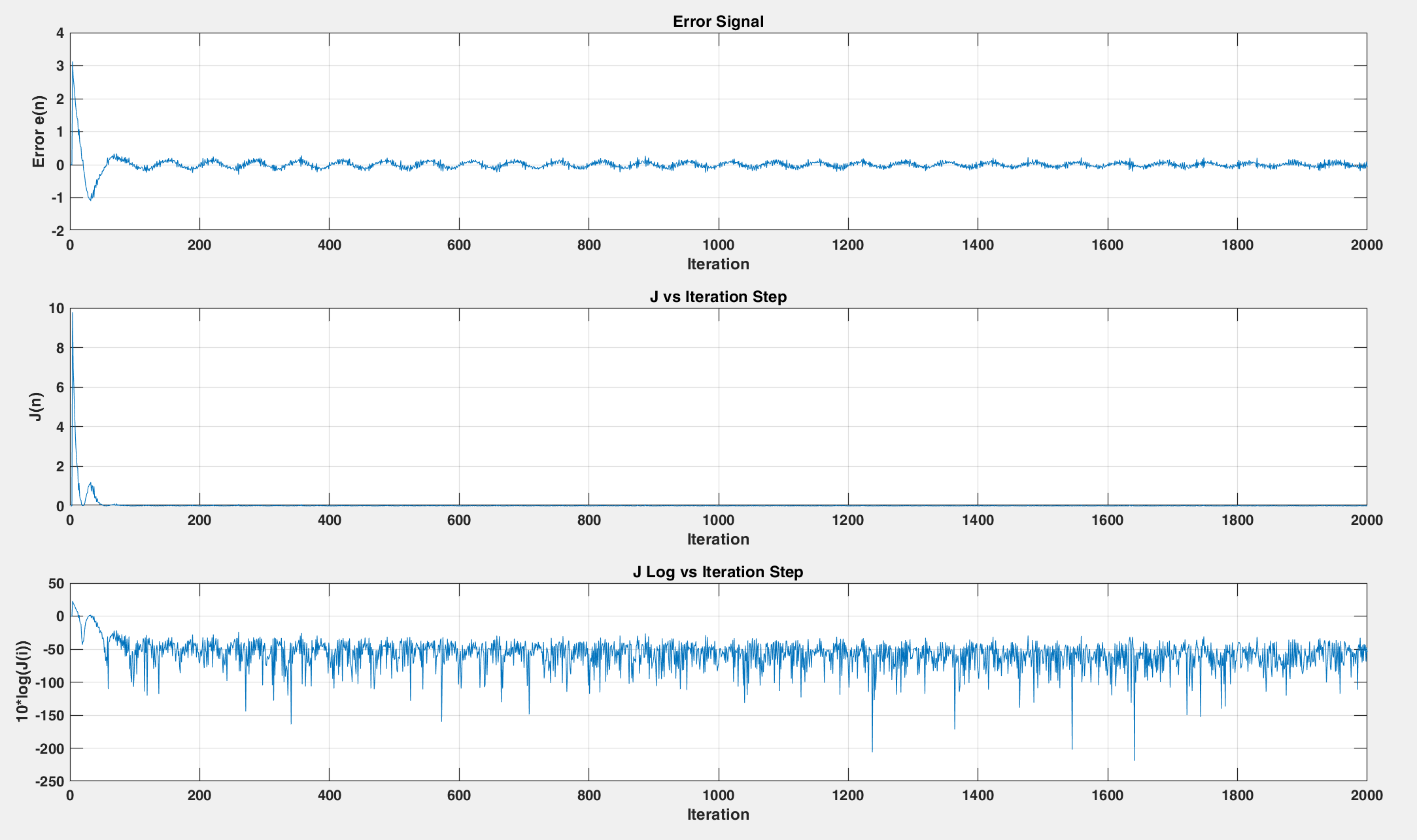
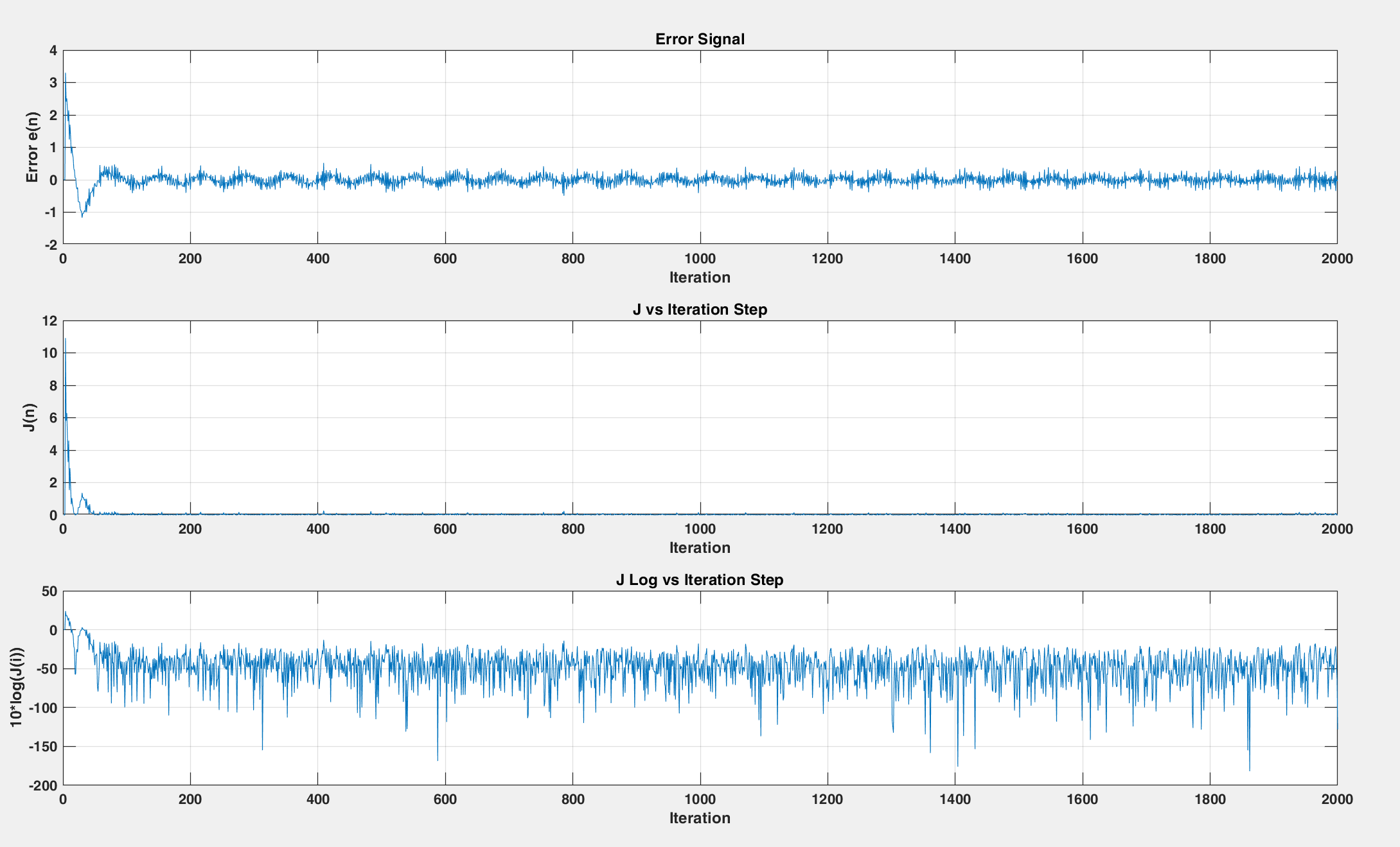
The step size (mu) in the LMS algorithm determines the convergence speed of the algorithm, as well as the steady-state error. Decreasing mu generally results in a slower learning process, but also leads to a smaller steady-state error. The step size controls the rate at which the filter coefficients are updated in response to the error between the desired signal and the output of the adaptive filter. A smaller step size means that the filter coefficients are updated more gradually over time. This leads to a slower convergence rate, but also ensures that the filter coefficients do not overshoot or oscillate, which can lead to instability in the filter. A smaller step size can also help to reduce the steady-state error by allowing the filter to adjust more precisely to the desired response. On the other hand, increasing the step size leads to a faster convergence rate, but also increases the risk of overshooting and instability in the filter. A larger step size can also lead to a larger steady-state error, since the filter may not have enough time to fully converge to the desired response.

Then, AWGN of 40 dB is added to the input signal x[n] using the awgn built-in MATLAB function.



1. Noisy Input Signal and its Magnitude and Phase Spectra.

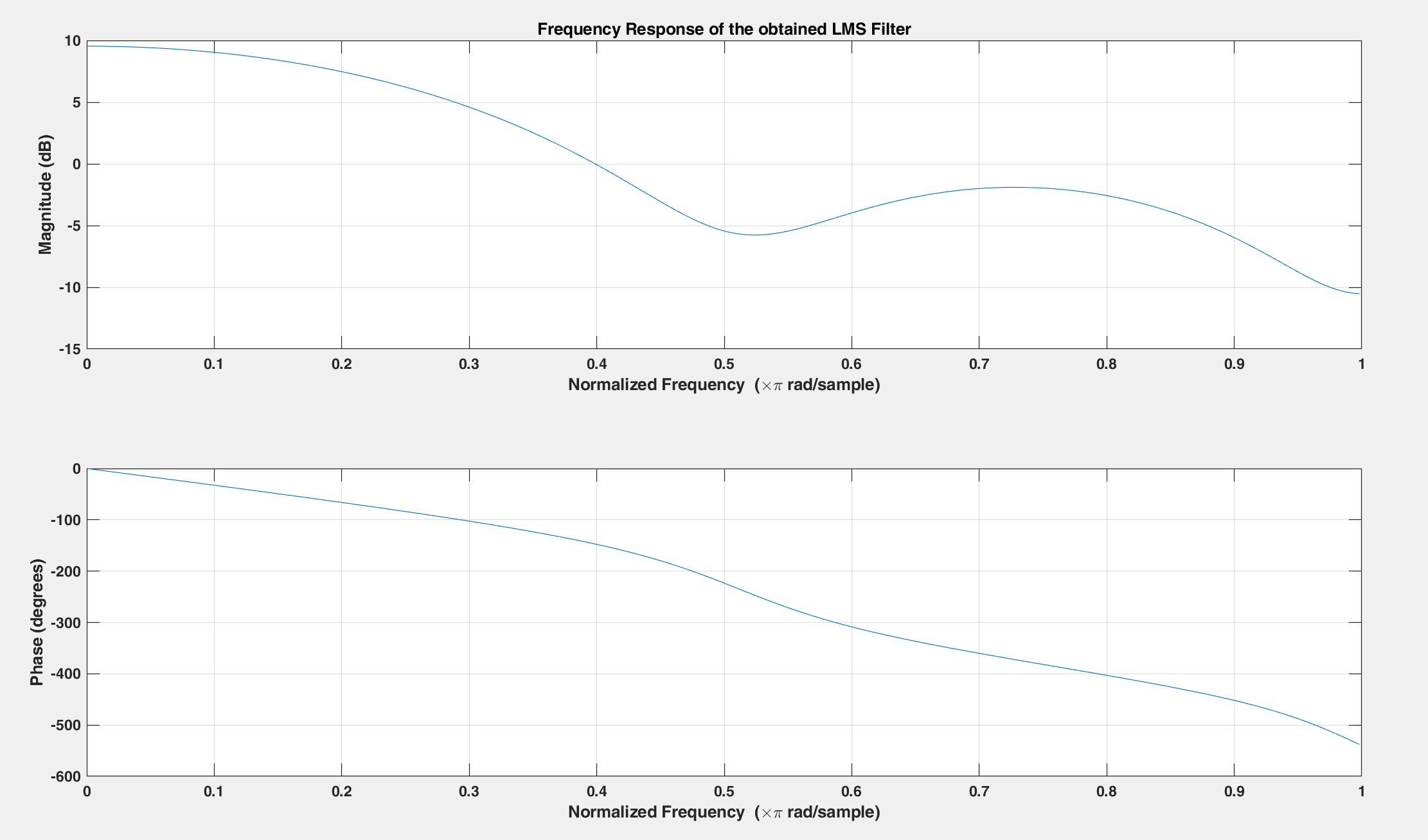
LMS Algorithm is applied on the noisy input signal x[n] and the error curves are shown in Fig. 12.



1. Error curve and J vs Iteration curve of the Noisy Signal.

The LMS filter response is shown in Fig. 13.

1. Response of LMS obtained filter.



The added noise can cause the LMS algorithm to converge more slowly and lead to larger steady-state errors. This is because the added noise makes the input signal more difficult to model, leading to a higher level of estimation error.

Then, AWGN of 30 dB is added to the input signal x[n] using the awgn built-in MATLAB function.

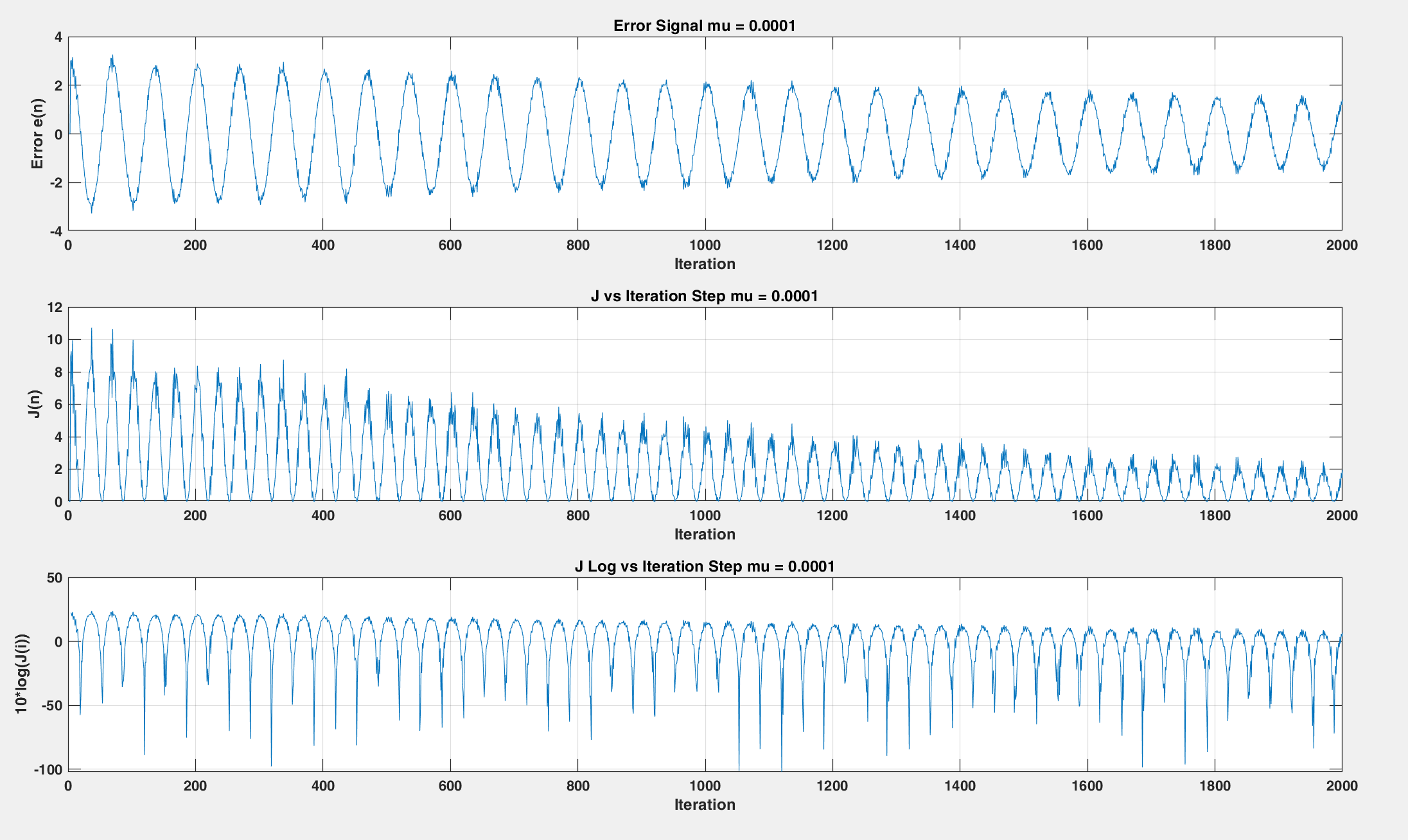


1. Noisy Input Signal and its Magnitude and Phase Spectra [30 dB].

LMS Algorithm is applied on the noisy input signal x[n] and the error curves are shown in Fig. 15.

1. Error curve and J vs Iteration curve of the Noisy Signal [30 dB].

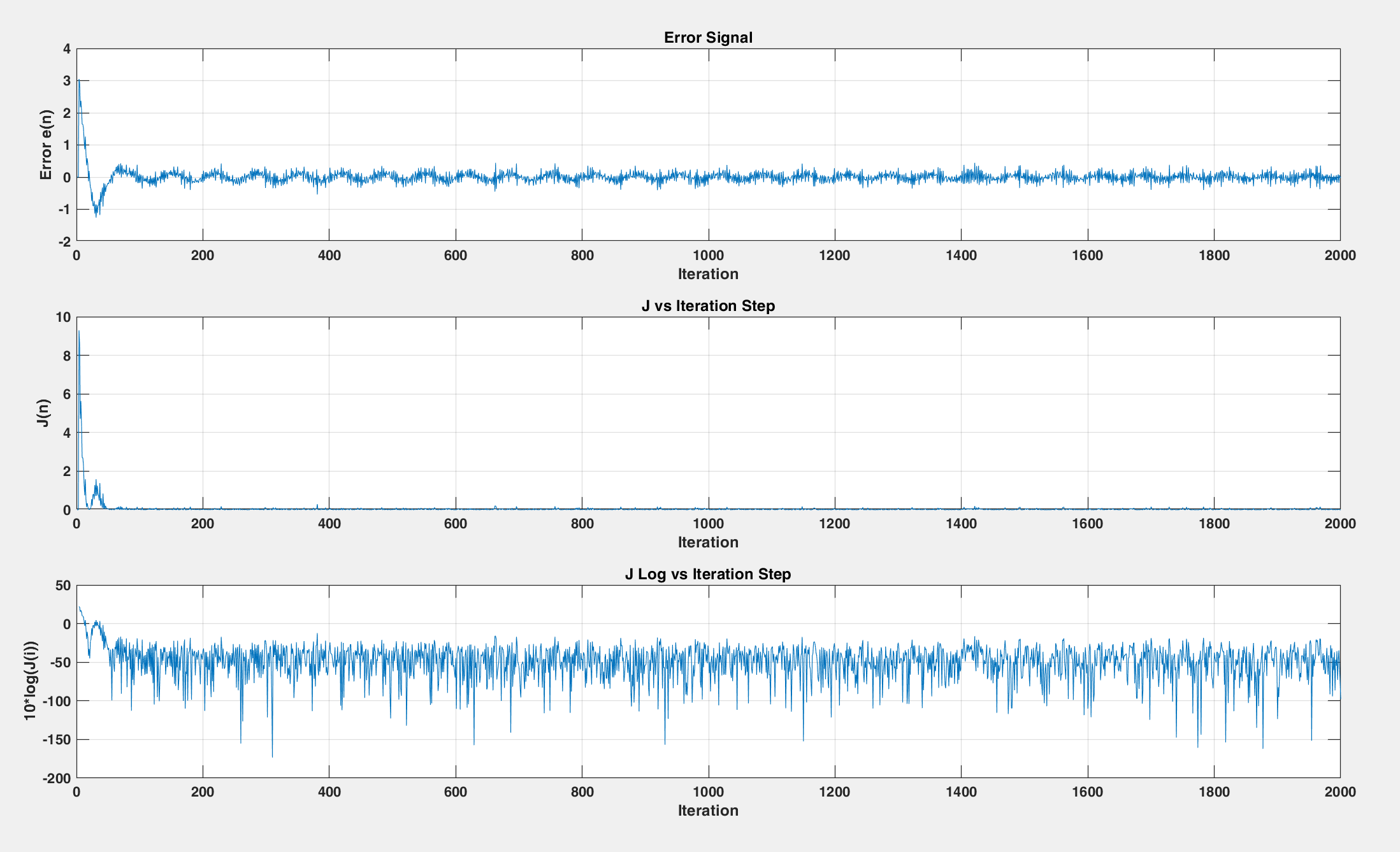
The LMS filter response is shown in Fig. 16.

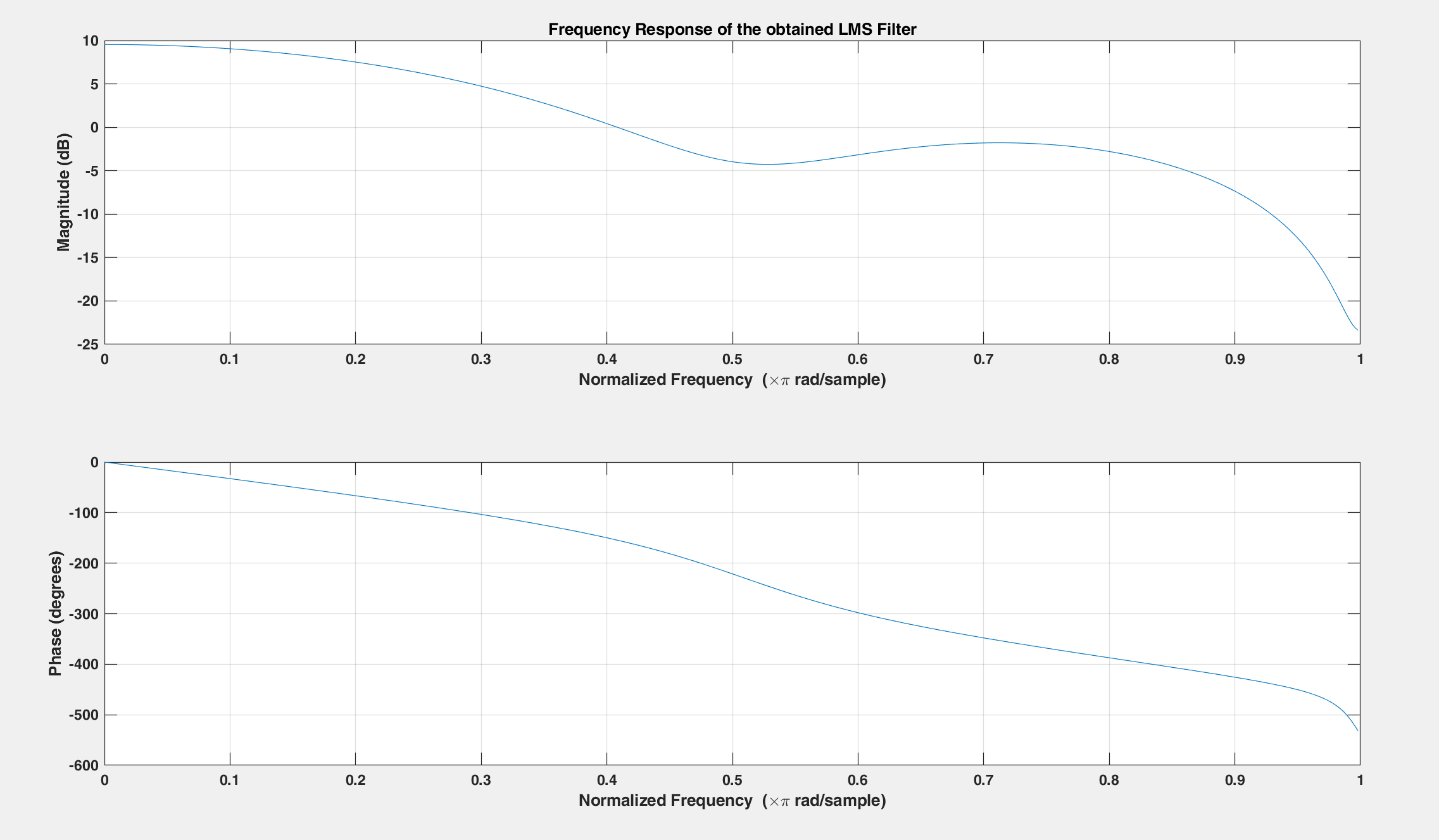
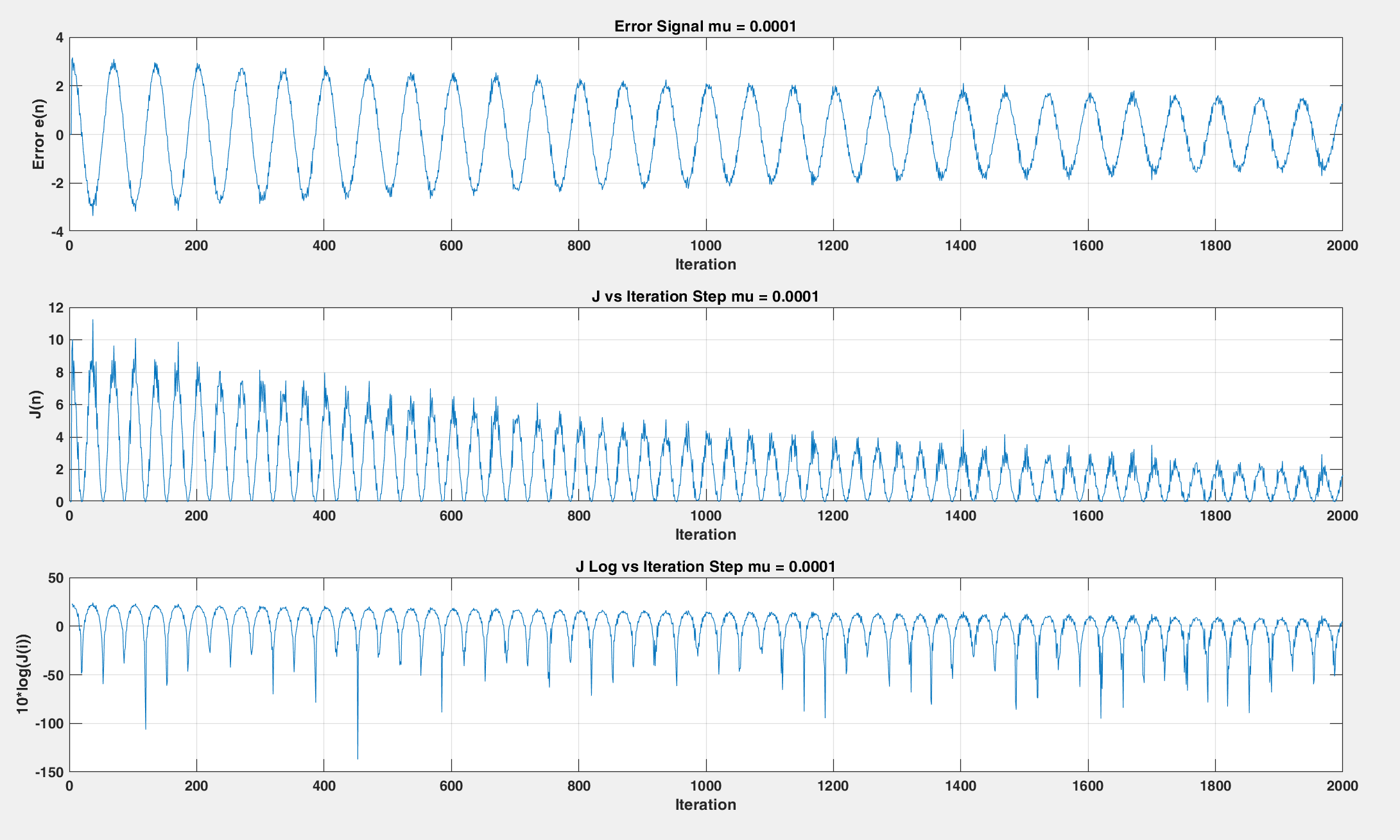


1. Error curve and J vs Iteration curve of the Noisy Signal [30 dB, mu = 0.0001].

The added noise can cause the LMS algorithm to converge more slowly and lead to larger steady-state errors. Therefore, if the SNR is decreased, the steady state error will increase as shown in Fig. 16.

Ensemble averaging over 1000 trails is performed to obtain J over the number of trials and its average is plotted.



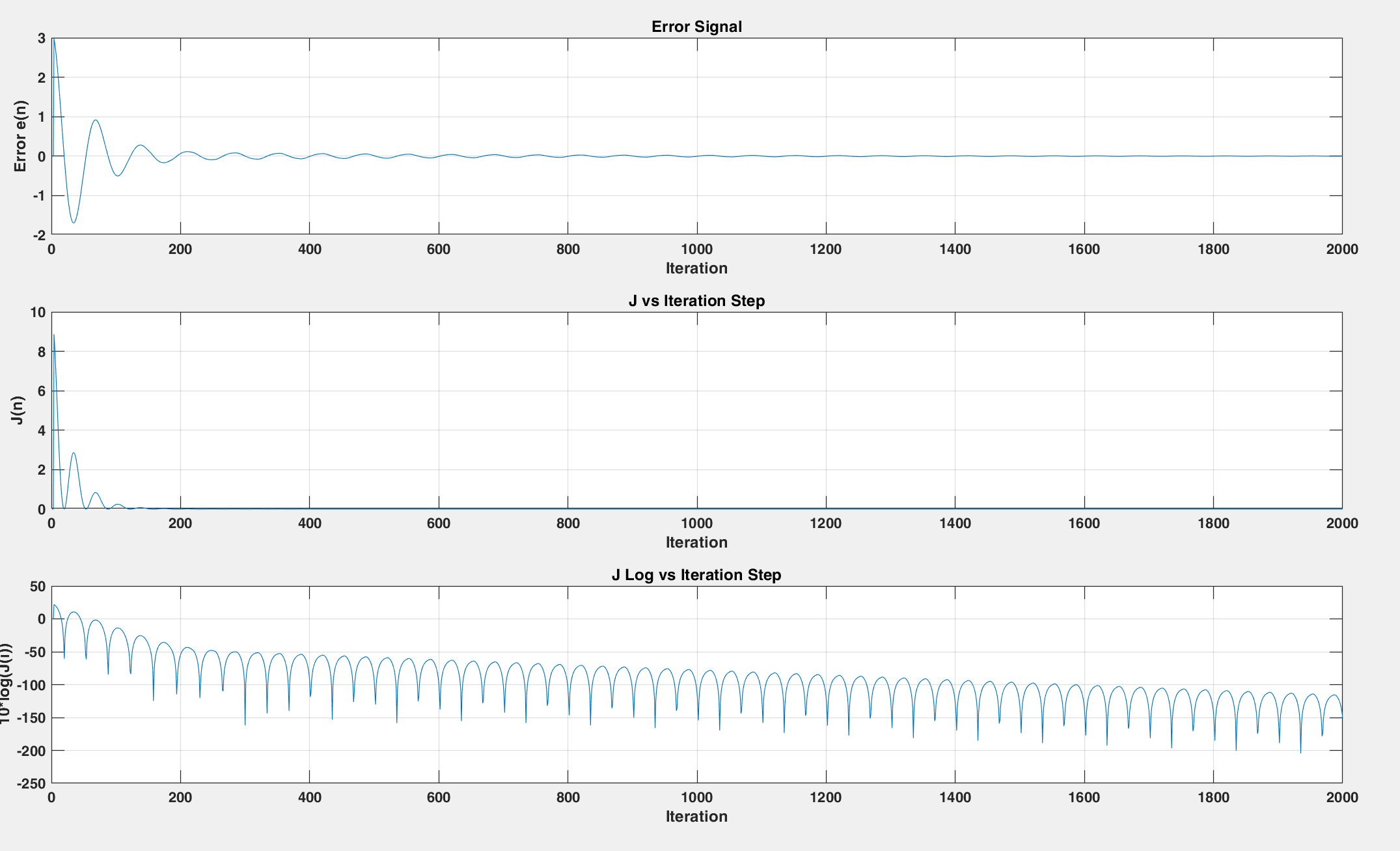
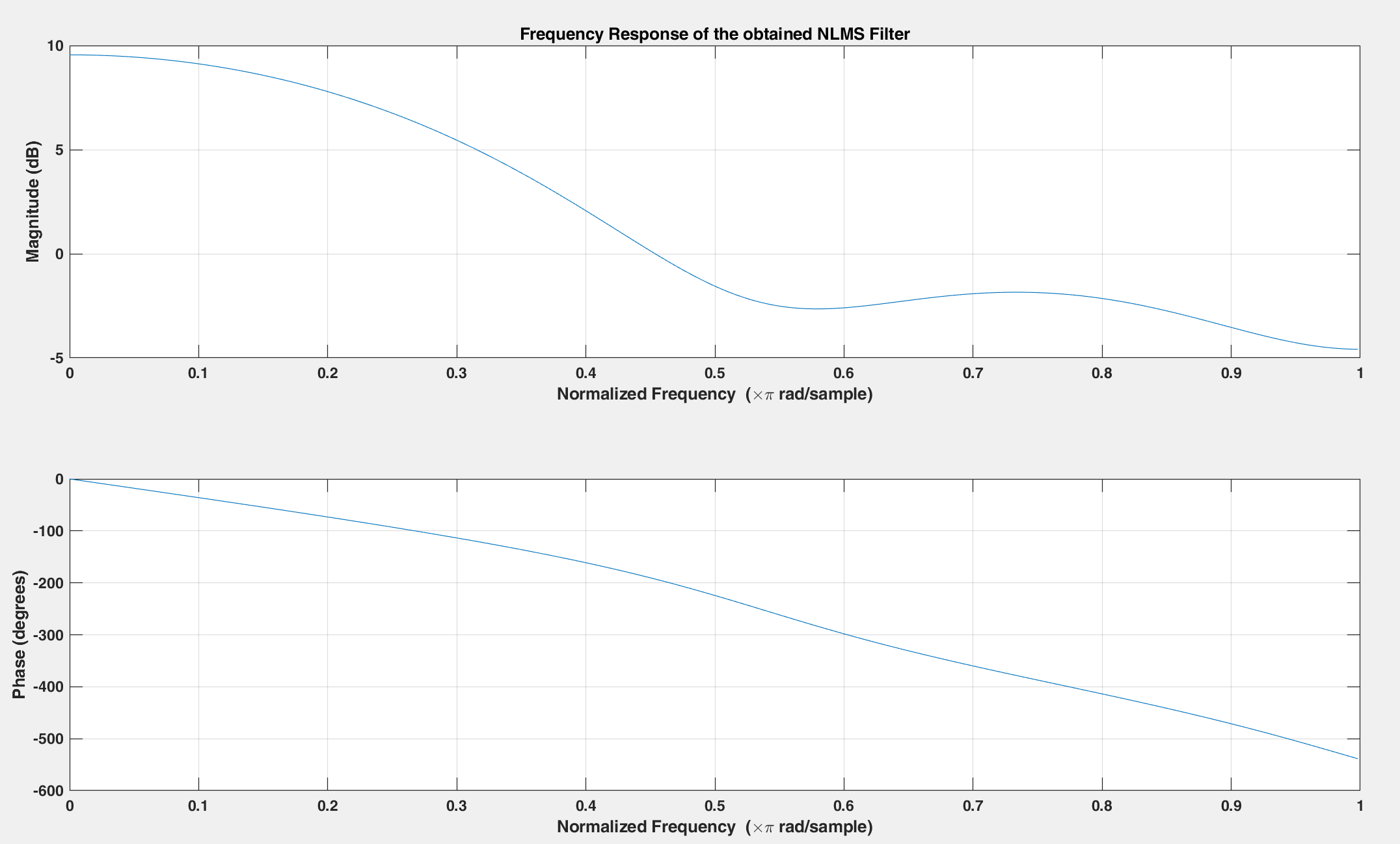
1.  Ensembled Error curve and J vs Iteration curve of the Noisy Signal.
2.  Ensembled Response of LMS obtained filter.
3.  Ensembled Error curve and J vs Iteration curve of the Noisy Signal [mu = 0.0001].
4. Ensembled Response of LMS obtained filter [mu = 0.0001].

After implementing LMS algorithm, the coefficients of the adaptive filter are obtained while setting the parameters to be:

μ = 0.01

M = 4 (Order of the adaptive filter

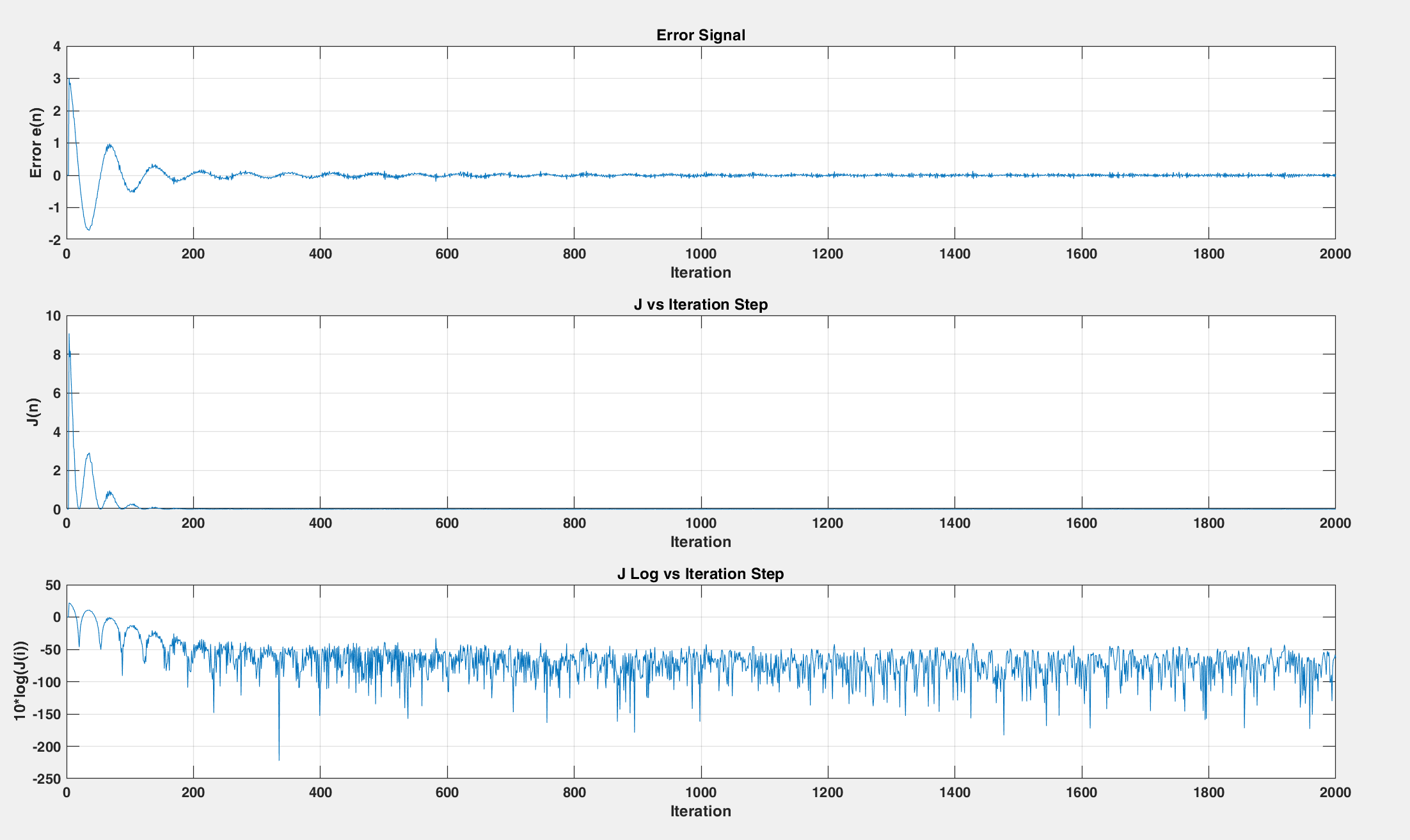
1. Error curve and J vs Iteration curve (NLMS).

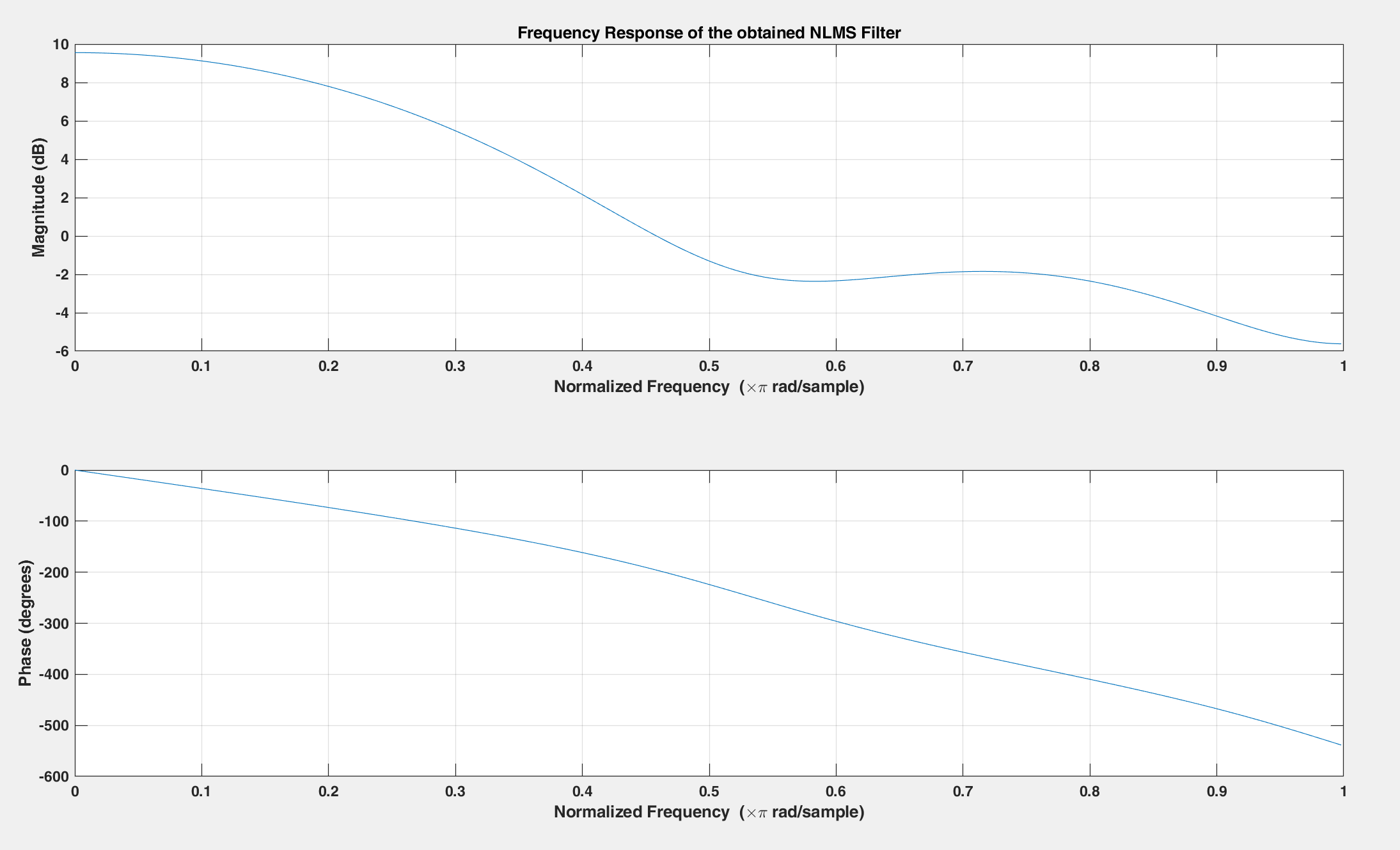


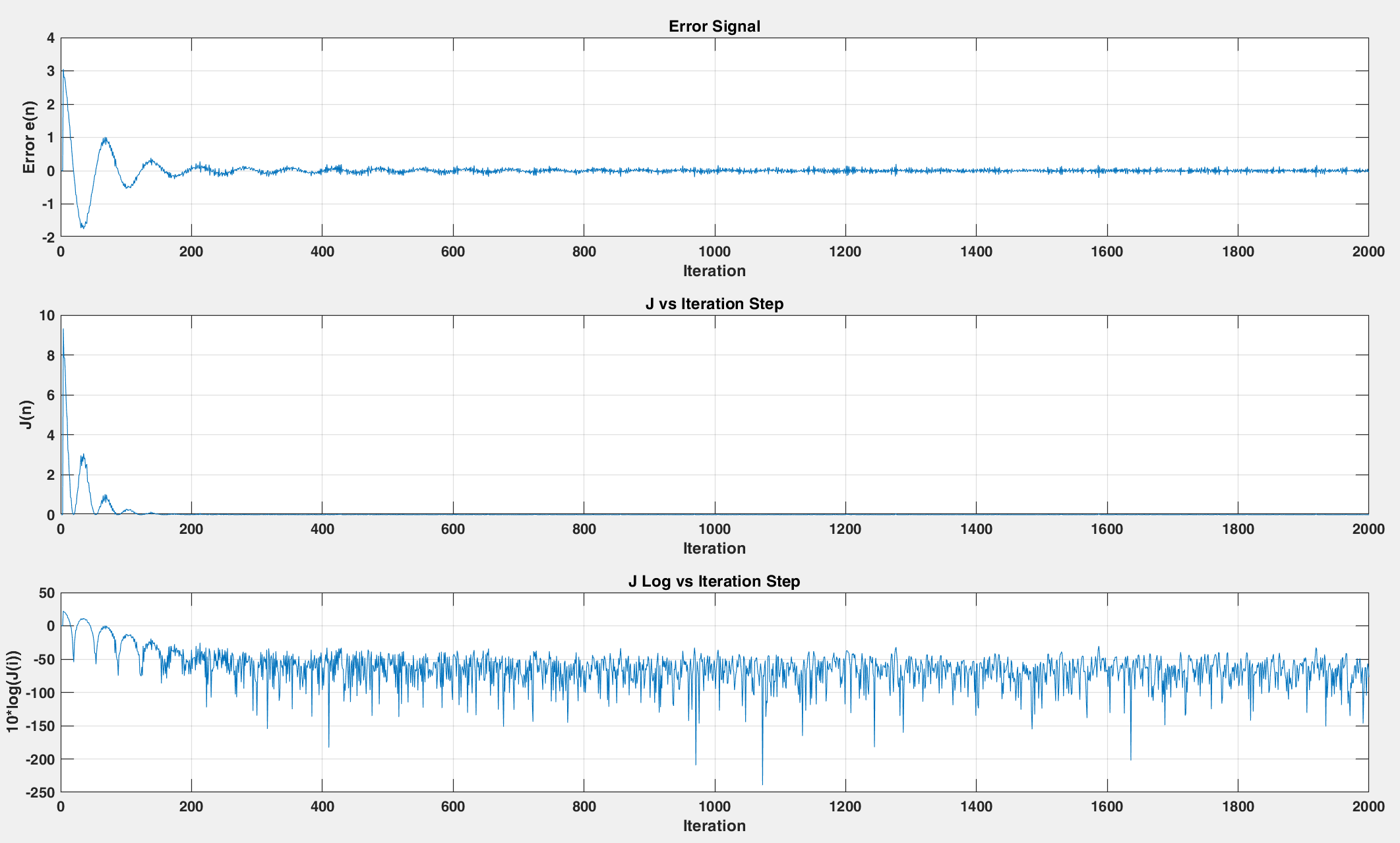
1. NLMS Filter Response.

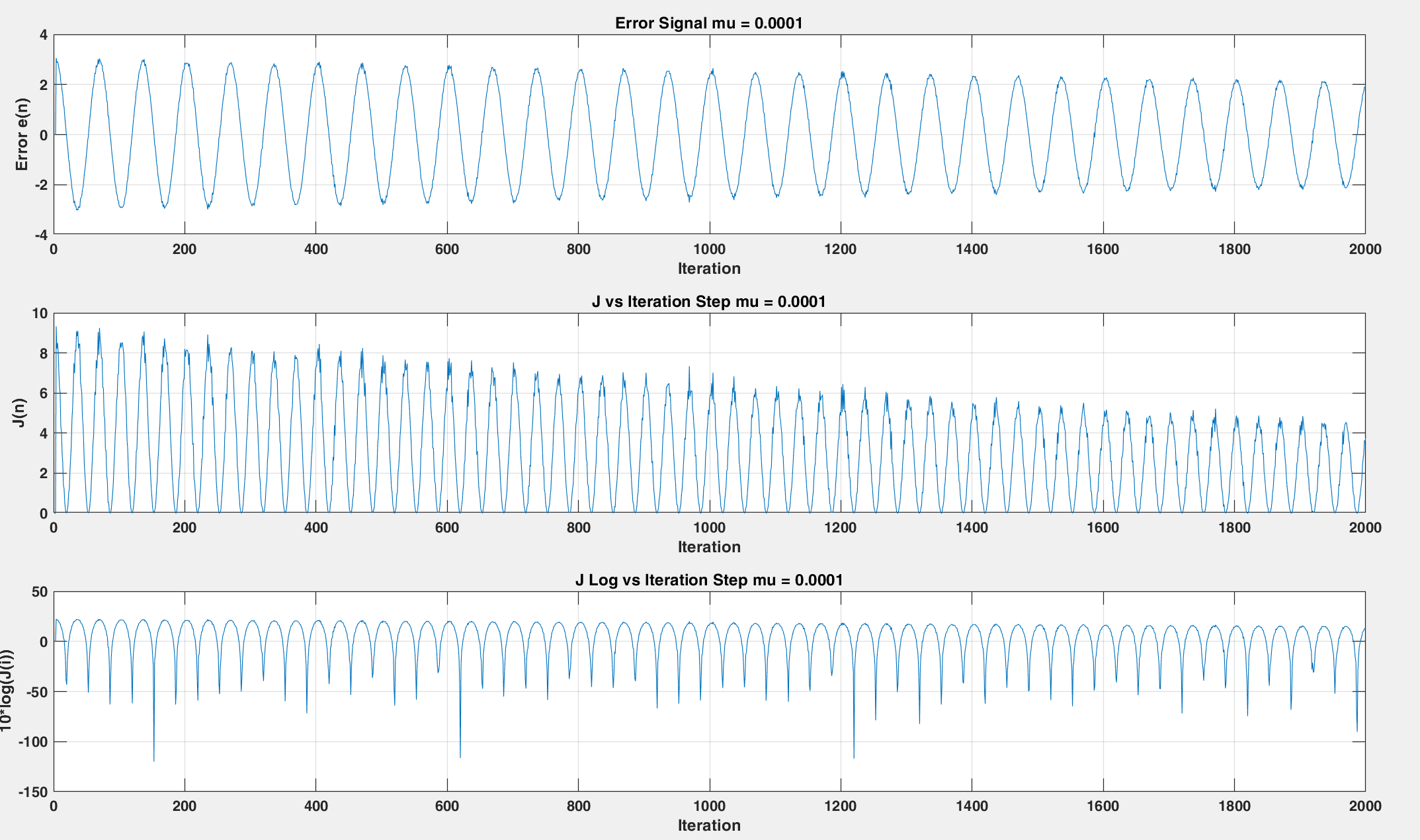
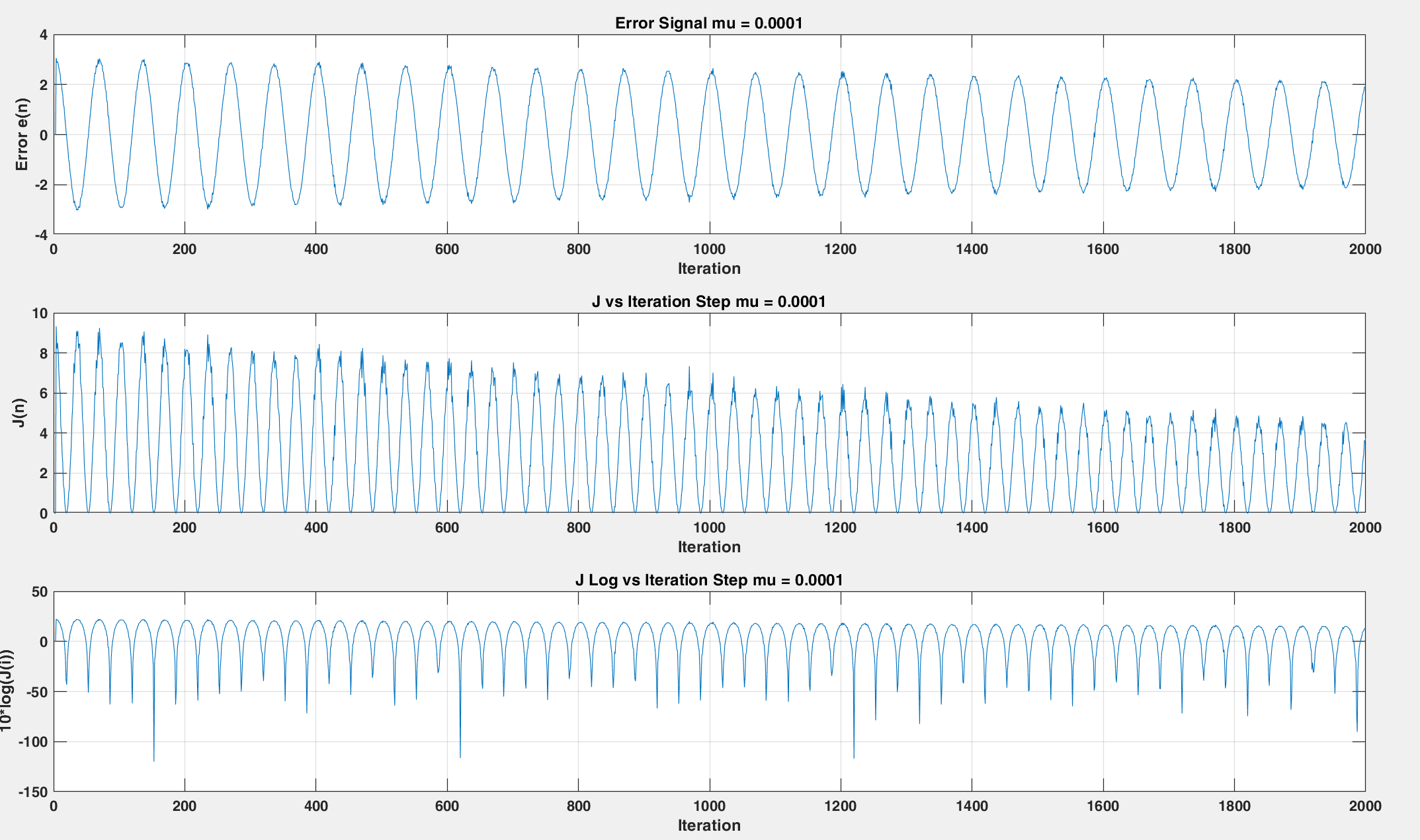
Then, mu value is decreased to 0.0001 and the error curve and NLMS filter response are obtained.

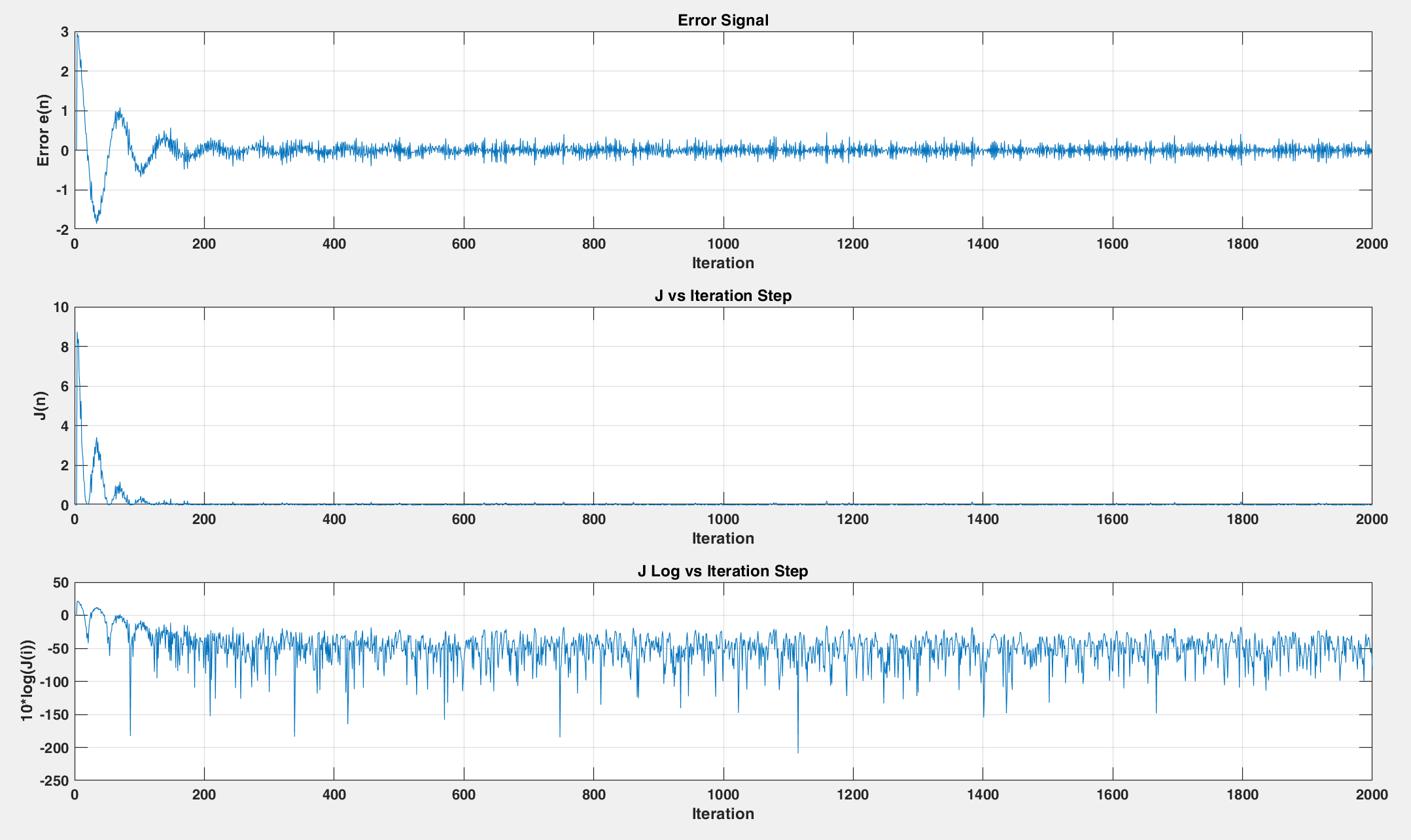
1. Error curve and J vs Iteration curve (NLMS) [mu = 0.0001].



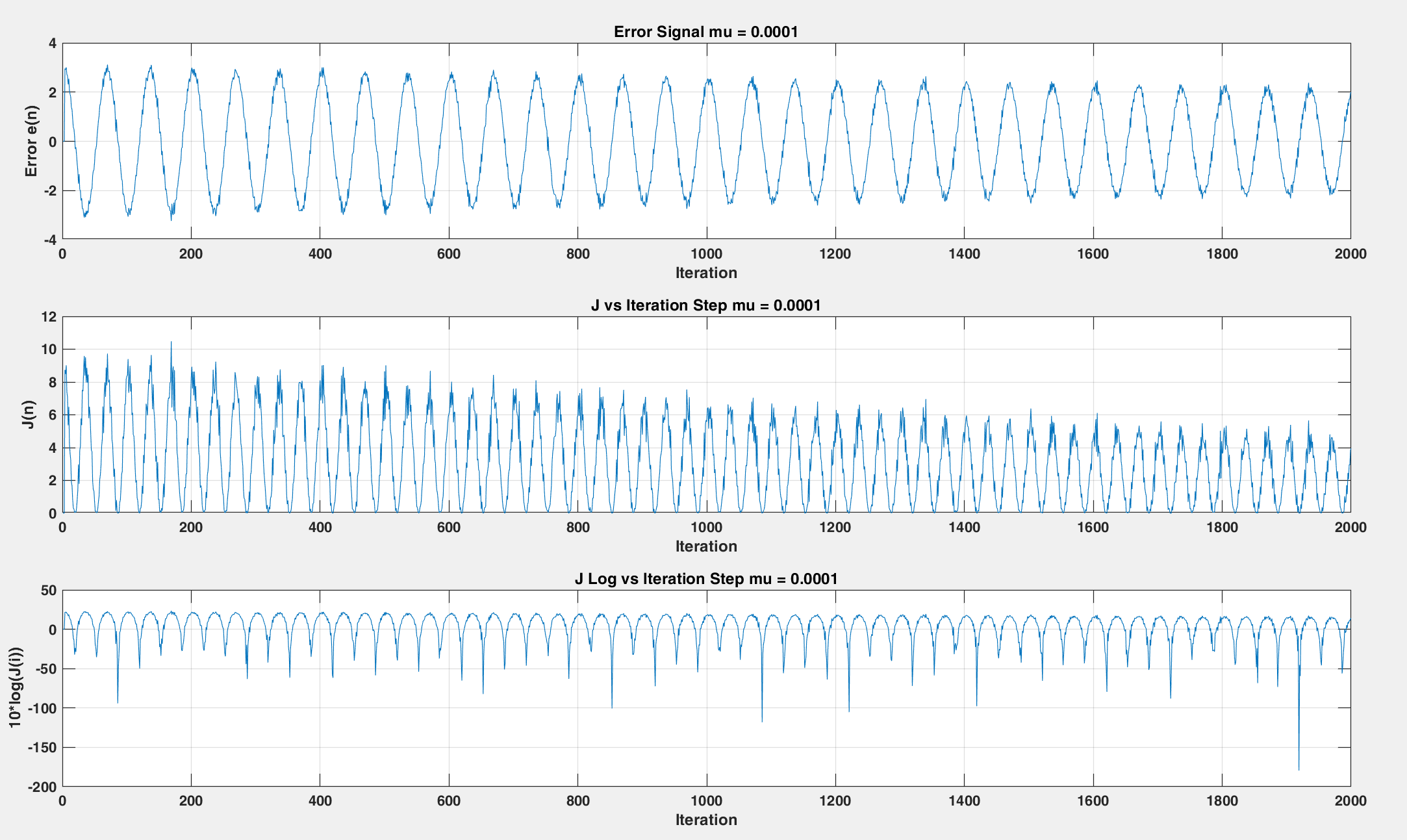
1. NLMS Filter Response [mu = 0.0001].

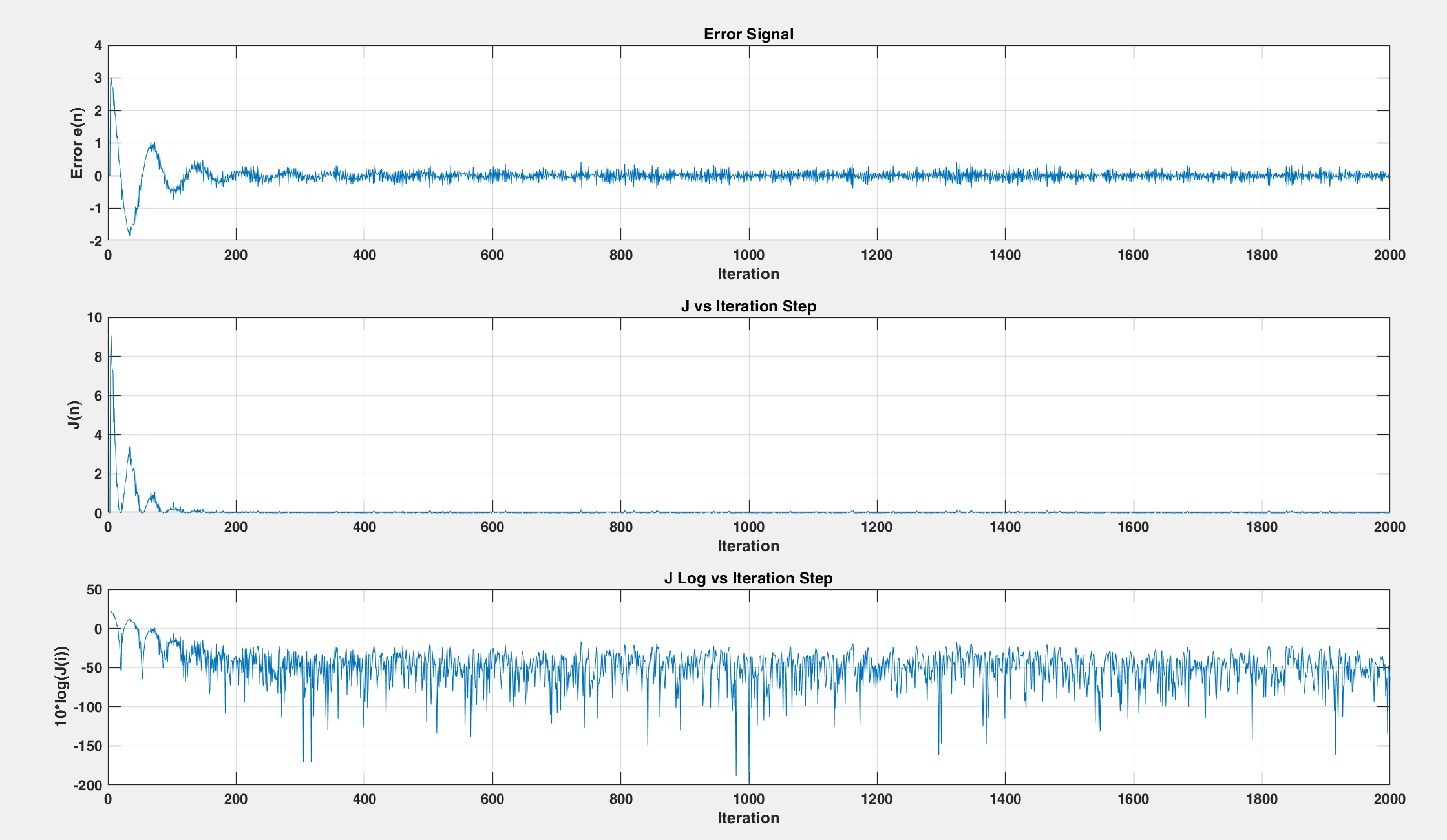
Then, 40 dB AWGN is added to the input signal.

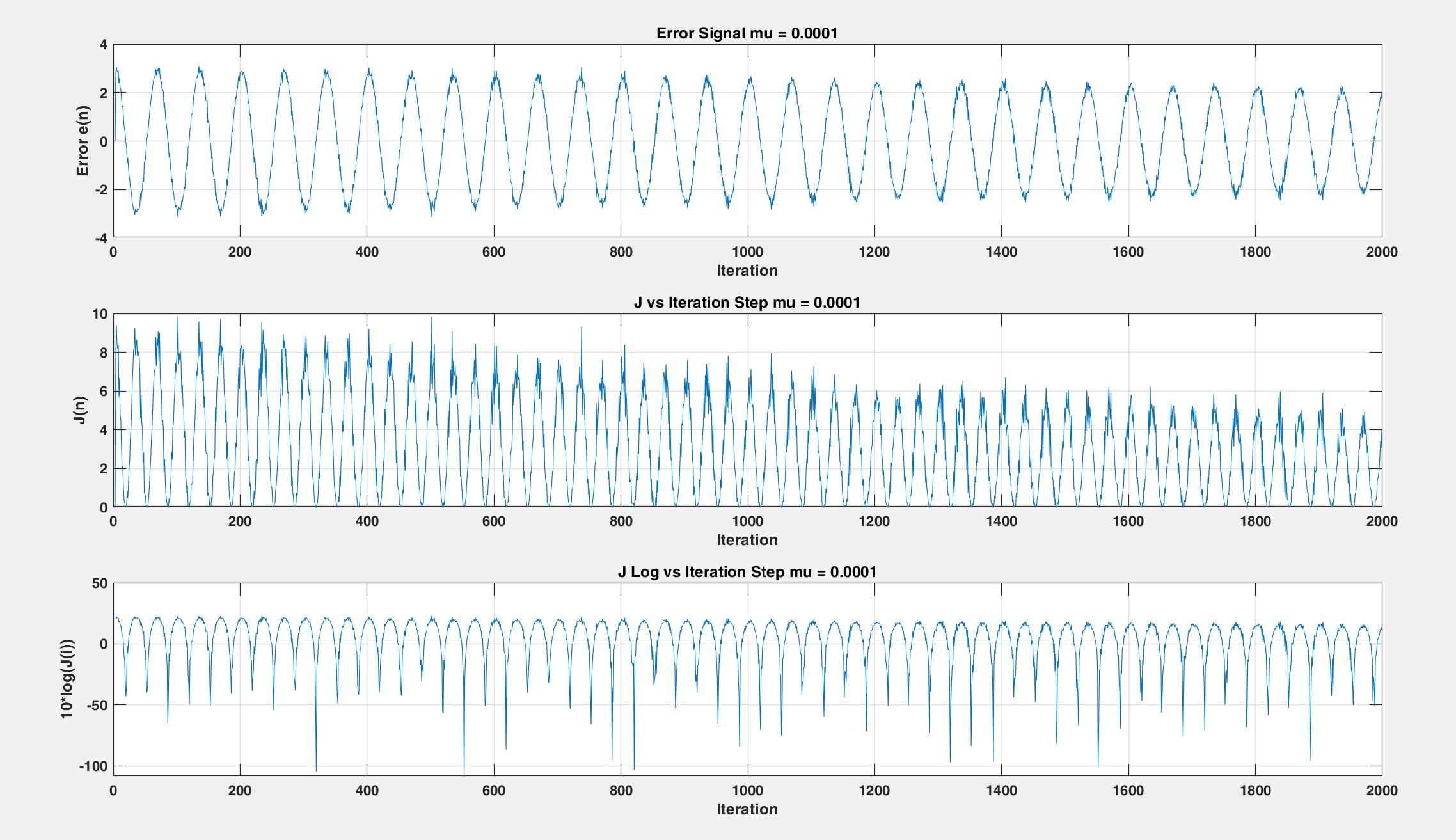
1. Error curve and J vs Iteration curve (NLMS) [SNR = 40 dB].
2. Error curve and J vs Iteration curve (NLMS) (mu = 0.0001) and [SNR = 40 dB].

After that, AWGN of 30 dB is added to the input signal.

1. Error curve and J vs Iteration curve (NLMS) [SNR = 30 dB].
2. Error curve and J vs Iteration curve (NLMS) (mu = 0.0001) and [SNR = 30 dB].



Ensemble averaging over 1000 trails is performed to obtain J over the number of trials and its average is plotted.

1. Ensembled Error curve and J vs Iteration curve of the Noisy Signal.
2. Ensembled Error curve and J vs Iteration curve of the Noisy Signal [mu = 0.0001].

# Development

The algorithms can be used with different input signals and different parameters (mu, SNR) in order to make sure that it can lead to correct results given different conditions. Moreover, the algorithm can be optimized to decrease the time and memory complexity.

# Conclusions

Least Mean Squares (LMS) and Normalized Least Mean Squares (NLMS) are two popular algorithms used in System Identification in Digital Signal Processing (DSP). Both algorithms are used to estimate the unknown system parameters in a signal processing system by minimizing the mean square error between the actual output and the estimated output.

Advantages of LMS algorithm:

* LMS algorithm is easy to implement and has a simple structure.
* It requires only a small amount of memory and computational resources.
* The convergence rate of LMS algorithm is very fast compared to other algorithms.

Disadvantages of LMS algorithm:

* LMS algorithm is sensitive to the choice of the step size (mu) value. If the value of mu is too large, the algorithm may not converge, and if it is too small, the convergence rate will be slow.
* LMS algorithm is sensitive to the presence of noise in the input signal, which can cause the algorithm to diverge.
* The LMS algorithm may be affected by the initialization of the filter coefficients, which may result in poor performance.

Advantages of NLMS algorithm:

* The NLMS algorithm is not sensitive to the choice of the step size value.
* The NLMS algorithm is less sensitive to the presence of noise in the input signal compared to LMS algorithm.
* The NLMS algorithm can be used for both online and batch processing.

Disadvantages of NLMS algorithm:

* The NLMS algorithm has a higher computational complexity than the LMS algorithm.
* The convergence rate of the NLMS algorithm is slower than that of the LMS algorithm.
* The NLMS algorithm requires a higher amount of memory compared to the LMS algorithm.

In conclusion, both LMS and NLMS algorithms have their own advantages and disadvantages. The choice between the two algorithms depends on the specific requirements of the application, such as computational complexity, speed of convergence, noise immunity, and memory requirements.in digital signal.

##### References

1. B. Widrow, S. Stearns, Adaptive signal processing. *Prentice Hall*, 1985
2. S. Haykin, Adaptive filter theory (3rd ed.), *Prentice-Hall*, 1996.
3. B. Farhang-Boroujeny, "Adaptive Filters Theory and Applications", *John Wiley & Sons*, 2010.