EXPERIMENT No. 3,4,&5

AIM: Implementation of the steepest-descent method, least-mean square (LMS) algorithm, and normalized least-mean square(NLMS) algorithm.

SOFTWARE TO BE USED: PYTHON

THEORY:

Steepest Descent method:

Steepest descent is a classical, deterministic method, which is the basis for stochastic gradient-based methods. The steepest-descent algorithm provides a local search method for seeking the minimum point of the error-performance surface, starting from an arbitrary initial point.
It is an iterative approach to find the optimal filter weight, $\mathbf{w_o = R^{-1}P}$ without inverting $\mathbf{R}$ matrix, where, $\mathbf{R}$ represents the autocorrelation matrix of the input signal and $\mathbf{P}$ represents the cross-correlation vector of the input signal and the desired signal.

Hence, in order to iteratively find $\mathbf{w_o}$, we use the method of steepest descent. To illustrate this concept, let $M = 2$, in the 2-D space $\mathbf{w(n)}$, the mean square error (MSE) $J_{\mathbf{w}}$ between the desired signal and its estimation forms bowl-shaped curve as shown in figure1.

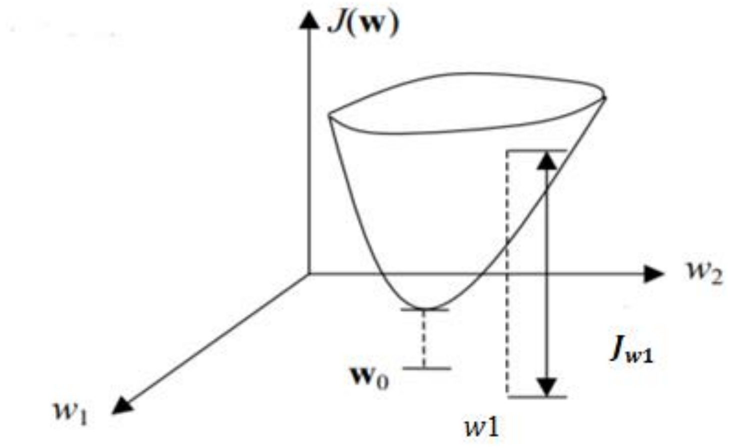

Figure 1: MSE forming Bowl-shaped function in 2D space.

From figure 1, if we are at a specific point in the Bowl, we can imagine dropping a marble. It would reach to the minimum (the optimal point w_o) point passing through many intermediate points. This minimum point can be obtained through an iterative approach known as the Steepest descent method.

The weight updation in the Steepest descent method is described as

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \frac{1}{2}\mu[-\Delta J_{\mathbf{w(n)}}],$$

where $\triangle J_{\mathbf{w(n)}}$ denotes the derivative of error curve at $\mathbf{w(n)}$.

Upon solving the derivatives $\triangle J_{\mathbf{w(n)}}$, we get

$$\Delta J_{\mathbf{w(n)}} = -2\mathbf{P} + 2\mathbf{Rw(n)}\,.$$

Hence,
$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu[\mathbf{P} - \mathbf{Rw(n)}] \text{ for } n = 0,1,2,\cdots$$

where μ is called the step size.

The output of filter is defined as

$$\mathbf{y(n)} = \mathbf{w}^{\mathbf{T}}\mathbf{(n)u(n)}$$

Stability: Steepest descent method employs feedback, there are limits on the step size that ensures the stability that can be established with respect to the eigenvalues of R.

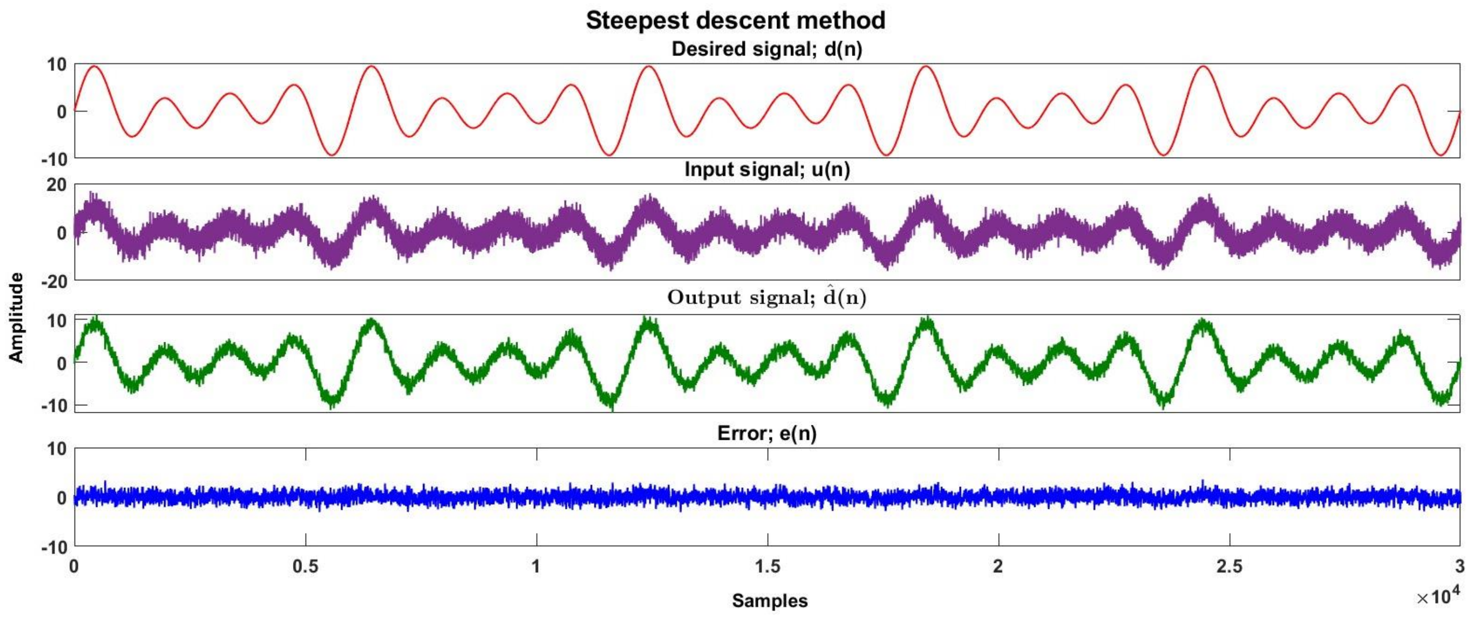

Figure 2: The figure shows input signal, input signal with added noise, result from steepest decent method, the error signal and the weight matrix used in the steepest decent method for **M=8** & **μ=0.0001** .

Least Mean Square (LMS):

The LMS algorithm is one of the simplest and most widely used algorithms for adaptive filtering. The LMS algorithm is based on the stochastic gradient descent method to find a coefficient vector which minimizes a cost function. In contrast to the Wiener filter, the parameters of the LMS algorithm change for each new sample received.

In the steepest descent method, the weight-vector update equation is given by

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu[\mathbf{P} - \mathbf{Rw(n)}].$$

The autocorrelation matrix is defined by $\mathbf{R} = E[\mathbf{u(n)u^H(n)}]$, however the same can be approximated for real time processing as

$$\mathbf{R} \approx \mathbf{u(n)u^H(n)}.$$

Similarly, the cross-correlation matrix $\mathbf{P} = E[\mathbf{u(n)}d^*(n)]$ can be approximated as

$$\mathbf{P} \approx \mathbf{u(n)}d^*(n).$$

Incorporating this estimate into the steepest descent method, the weight updation equation becomes

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu[\mathbf{u(n)}d^*(n) - \mathbf{u(n)u^H(n)^*w(n)}]$$

In the ensemble average is estimated using a one-point sample mean. Finally, combining the equations leads to the update equation in that is known as LMS algorithm.

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu\mathbf{u(n)}e^*(n),$$

where $\mathbf{w(n)}$ is the estimate of the weight value, vector at time n, $\mathbf{u(n)}$ is the input signal vector, $e(n)$ is the filter error vector and μ is the step-size which determines the filter convergence rate and overall behavior.

One of the difficulties in the design and implementation of the LMS adaptive filter is the selection of the step-size μ. This parameter must lie in a specific range, so that the LMS algorithm converges:

$$0 < \mu < \frac{2}{\lambda_{\max}},$$

where $\lambda_{\max}$ is the largest eigenvalue of the autocorrelation matrix $\mathbf{R}$.

The error between desired signal and the filter output is defined as

$$e(n) = d(n) - y(n) = d(n) - \mathbf{w}^H\mathbf{(n)u(n)}$$

The output of filter is defined as

$$\mathbf{y(n)} = \mathbf{w^T(n)u(n)}$$

Different way of updating weight matrix,

1. **Sign-Sign LMS**

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu sign(\mathbf{u(n)}) sign(e^*(n))$$

2. **Sign-Regressor LMS**

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu e^*(n) sign(\mathbf{u(n)})$$

3. **Sign-Error LMS**

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu \mathbf{u(n)} sign(e^*(n))$$

4. **Normalized LMS**

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu \frac{\mathbf{u(n)} e^*(n)}{\|\mathbf{u(n)}\|^2}$$

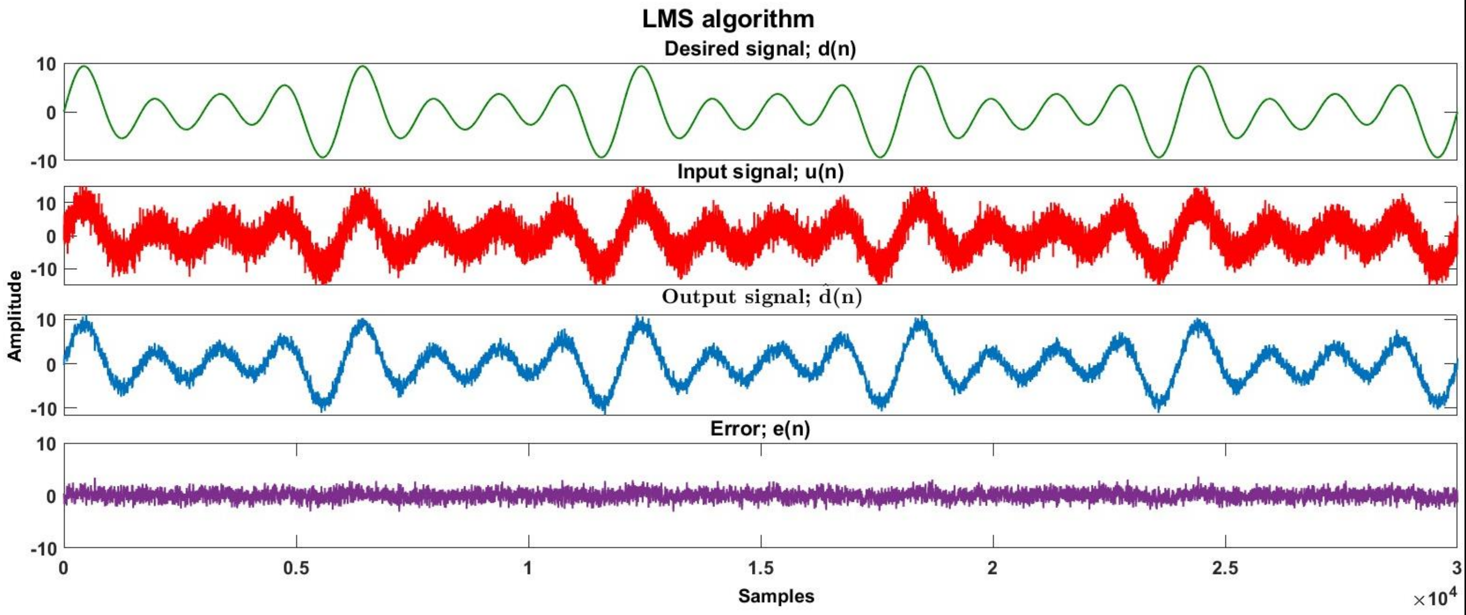

Figure 3: The figure shows input signal, input signal with added noise, result from Least mean square algorithm and the error signal for **M=8** & **μ=0.0001**.

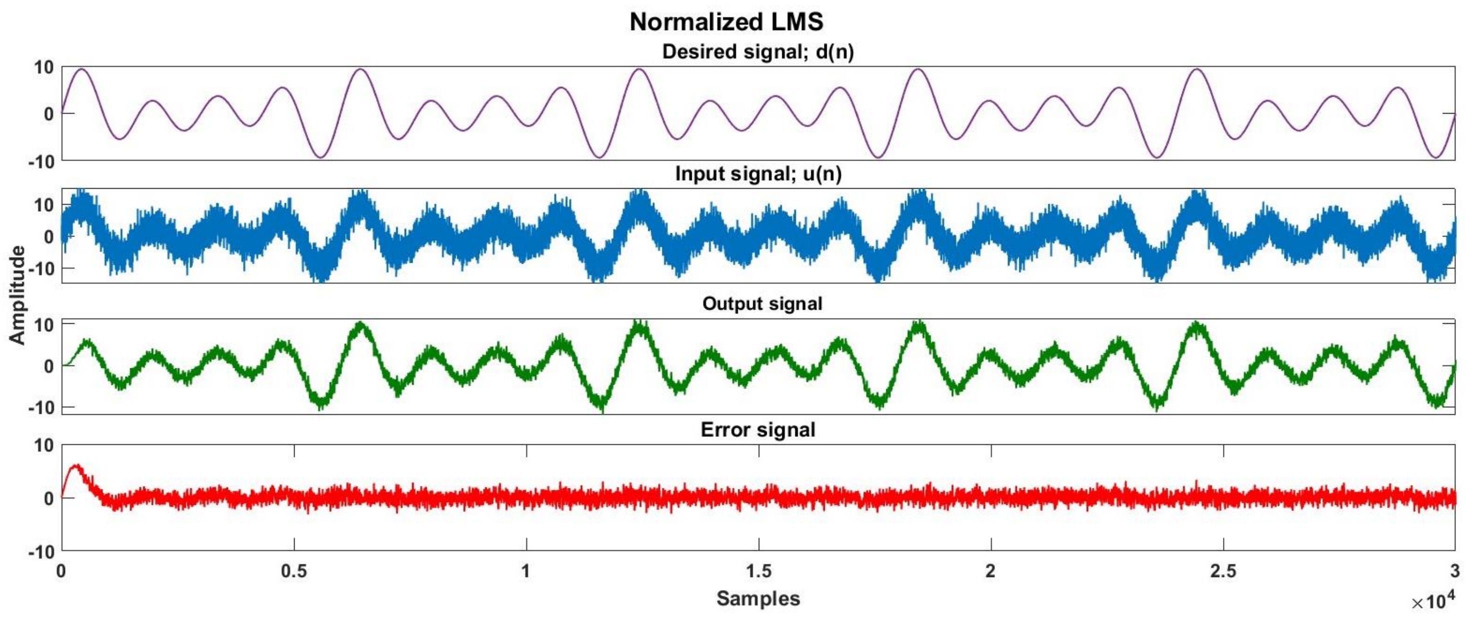

Figure 4: The figure shows input signal, input signal with added noise, result from Normalized Least mean square algorithm and the error signal for **M=8** & **μ=0.0001**.

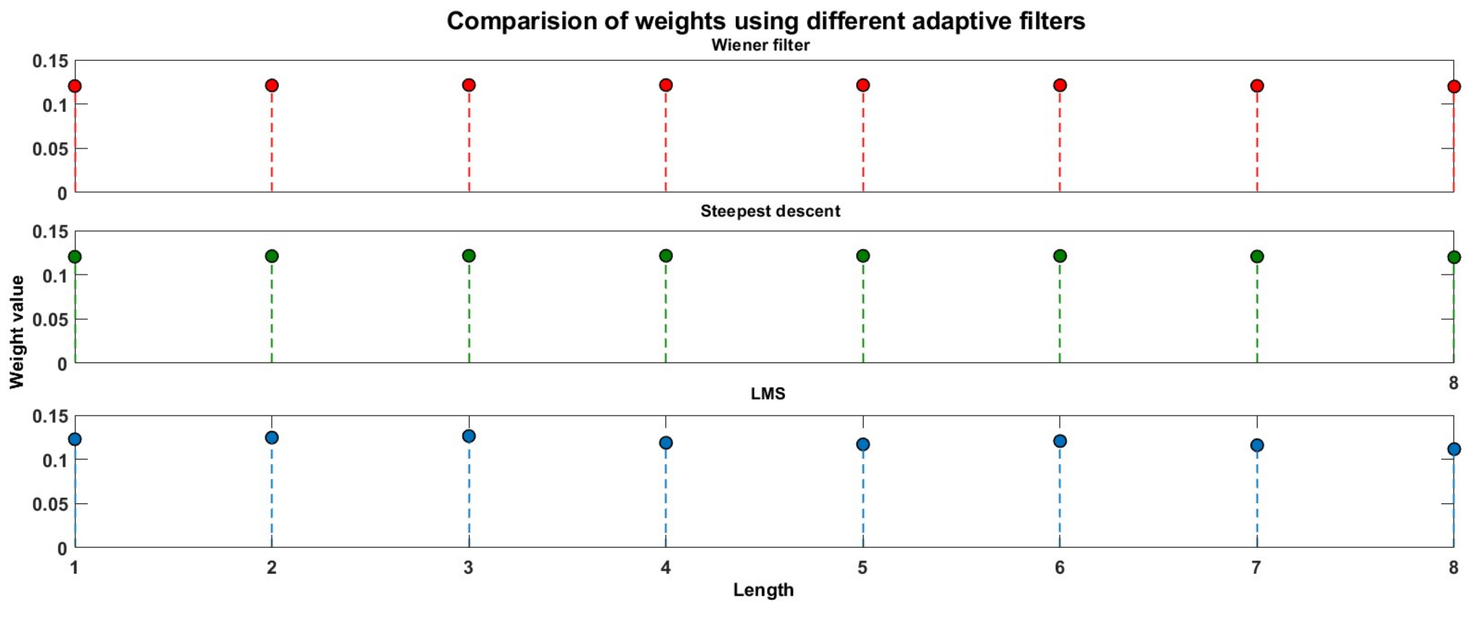

Figure 5: The figure shows weights from different adaptive filter methods for **M=8** & **μ=0.0001**.

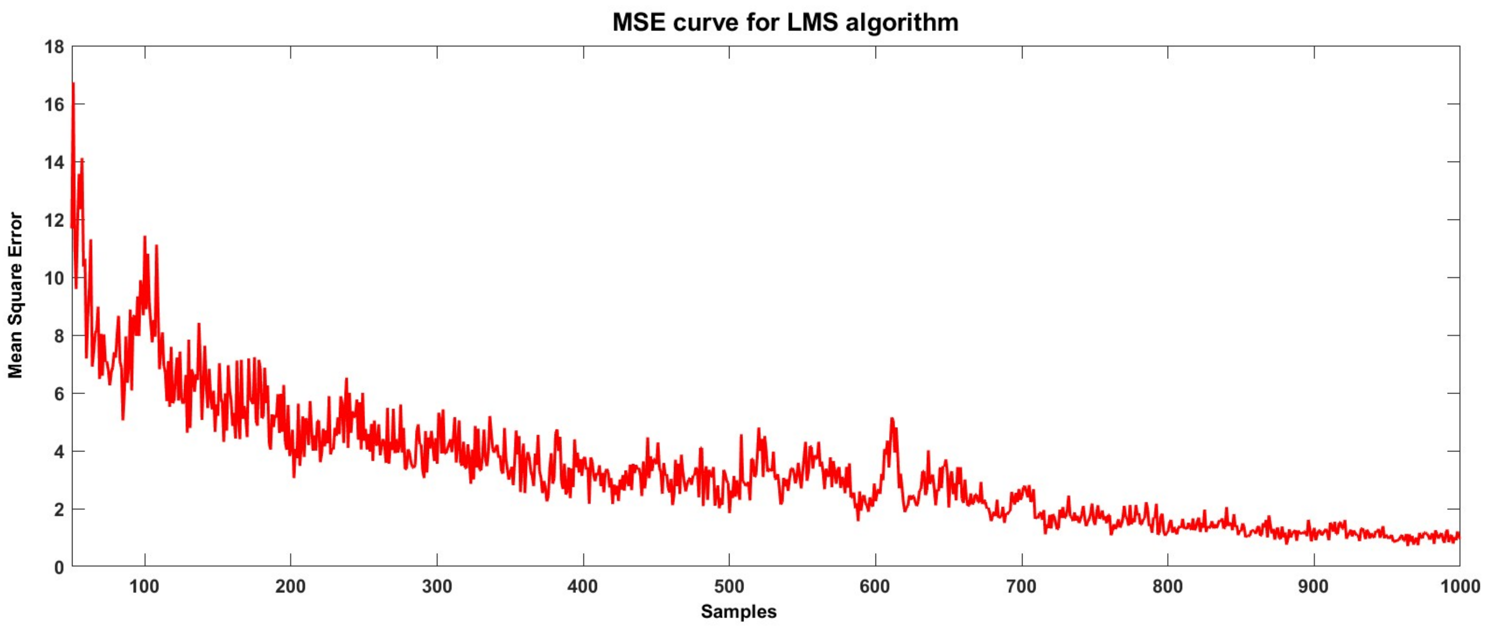

Figure 5: The figure shows the Mean Square Error of simple LMS algorithm for **M=8** & **μ=0.0001**.

Procedure:

- **Steepest Descent method**

1. Take any arbitrary signal *d(n)*
2. Add some random white noise with zero mean $v(n)$ to the $d(n)$ then we get noisy signal **u(n)**.
3. Obtain autocorrelation matrix $\mathbf{R}=\mathbf{u(n)u^H(n)}$ and cross-correlation matrix $\mathbf{P}=\mathbf{u(n)}d(n)$.
4. Define value of step-size (for e.g., $\mu = 0.01$).
5. Initialize weight matrix **w(n)** with some random/zeros values.
6. Update the weight matrix $\mathbf{w(n+1)=w(n)}+\mu(\mathbf{P-Rw(n)})$.
7. Perform filtering of $d(n)$ with updated weight matrix and we get estimate of desired signal $\hat{d}(n) = \mathbf{w(n)*u(n)}$.
8. Obtain the error signal $e(n) = d(n) - \hat{d}(n)$.

- **Least Mean Square (LMS):**

1. Take any arbitrary signal *d(n)*.

2. Add some random white noise with zero mean $v(n)$ to the then we get noisy signal $\mathbf{u(n)}$.
3. Obtain autocorrelation matrix $\mathbf{R=u(n)u^H(n)}$ and cross-correlation matrix $\mathbf{P=u(n)}d(n)$.
4. Define value of step-size (for e.g., $\mu = 0.01$).
5. Initialize weight matrix $\mathbf{w(n)}$ with some random values.
6. For Simple LMS Update the weight matrix $\mathbf{w(n+1)=w(n)+}\mu\mathbf{u(n)}e^*(n)$.
7. Different type of LMS algorithm has different way of updating weight matrix and the corresponding estimate of desired signal for every n^{th} index is obtained by $\hat{d}(n) = \mathbf{w(n)*u(n)}$.

   A. Sign-Sign LMS

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu sign(\mathbf{u(n)}) sign(e^*(n))$$

   B. Sign-Regressor LMS

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu e^*(n) sign(\mathbf{u(n)})$$

   C. Sign-Error LMS

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu \mathbf{u(n)} sign(e^*(n))$$

   D. Normalized LMS

$$\mathbf{w(n+1)} = \mathbf{w(n)} + \mu \frac{\mathbf{u(n)}e^*(n)}{\|\mathbf{u(n)}\|^2}$$

8. Obtain the error signal $e(n) = d(n) - \hat{d}(n)$ for different type of LMS.