Turkish Online Journal of Qualitative Inquiry (TOJQI)

Volume 12, Issue 7, July 2021: 2228 – 2245

Least Mean Fourth Based Constrained Adaptive Order Statistic Filters For Image Restoration

Ganeswara Padhy[a], Sudam Panda[b], Santanu Kumar Nayak[c]

[a*] Department of Electronic Science, Berhampur University, Bhanja Bihar,Ganjam,Odisha,PIN-760007

***Corresponding author:** ganeswar.padhi@gmail.com

Abstract

This paper proposes the adaptive constrained order statistic L and combination C (Ll) filter using the least mean fourth algorithm (LMF) with linear and non-linear structures at the output. Though, the LMF problem involves many stability problems due to noise variance and increase of input power. This can be avoided in Normalized Least Mean Fourth (NLMF) algorithm by normalising the weight update terms by the fourth power norm of the regressor. Here the LMF based adaptive L filter is derived with linear and non linear output of a fixed window. Whereas the adaptive C filter uses the rank order and temporal order information from the input sequence of fixed window. These filters use ordered data to remove non-Gaussian noise components, preserves the edges and details of an image. In this paper, the performance of C filter overrides the performance of the LMF-L and other LMF based filters.

Keywords: NLMF, Adaptive filters, L and C filters, Image restoration

1. Introduction

In recent years, adaptive filtering techniques have been used in many areas of signal and image processing like channel equalization, system identification, echo cancellation in telephone channels adaptive arrays and elimination of narrowband interference in wideband signals [1-6], digital image filtering, image enhancement, and edge detection. The well-known linear filters are FIR Weiner and lattice filters. These are simple for implementation and remove high-frequency noises but fail to remove non-Gaussian noises and signal-dependent noise filtering (impulse noise filtering). When the signal is non-linear and non-stationary in nature, linear filters cannot produce good results, so non-linear techniques like Voltera filter [7] functional link adaptive network (FLAN) [8], artificial neural network (ANN) models, polynomial [9-10] and order-statistic filters [11] have been developed alternate to linear techniques.

The order-statistics (OS) filters have the independence of spatial or temporal positioning within the data window. The appropriate selection of coefficients can reduce the smoothing of streaks [12] and edge jitters [13]. The termination criteria is based upon the mean square error (MSE) criterion [14-16], which is robust against the variation of signal and noise properties [14, 17-19]. It can be easily implemented in real-time models [20, 21] and suitable to restore the images.

The best-known order statistics filters are median filters, weighted median filters [6], L filter, and Ll filter [24,25], which use the concept of a certain ordering, that can be algebraically represented as median filters are well known for their edge-preserving in nature but it creates streaks and edge displacement of the images [12, 13].

There are different types of OS filters, such as linear order-statics filters (Ll or C filter), weighted median filters, and non-linear order-statistics filters (L-filter and median filter). These filterers are used in many applications of image and signal processing. Adaptive filtering techniques play an important role in statistical signal processing applications. Many of the researchers have worked out by combining adaptive filtering and nonlinear filtering techniques. When the image is embedded with noise while transmitting through nonlinear channels, noise becomes prominent, close to the edge than the smooth regions. The least mean square (LMS) [22] and constrained LMS [23] based L filters depend upon minimization of mean squared error (between the original image and restored image after adaptation) and also avoid the computational burden in comparison with the filters. The drawback of the algorithm is that the weight update occurs in a heuristic manner, and it does not depend upon the minimization of the error norm. Other techniques like the RLS algorithm, constrained LMS algorithm, normalized LMS [26] are available to design L-filters basing upon the minimization of mean square error (MSE). These algorithms are independent of input data, noise, selection of step size, and capable of tracking the signal's varying statistics.

The least mean fourth (LMF) algorithm performs [27, 28] better than the LMS in Gaussian noise environments achieving the better trade-off between transient and steady-state performances of an adaptive filter. But its application is limited due to its stability problem. The advantages of the LMF algorithm is faster in its initial convergence and lower in steady-state error than the LMS algorithm because of the fourth power of its cost function. The higher-order cost function requires the smallest step size to ensure the stable adaption [30], whereas the existence of a third-order error term in the weight updation formula causes instability at the initial stage.

To improve the stability of the LMF algorithm, the normalized version of this algorithm called Normalized LMF (NLMF) has been worked out by many scientists [31-34]. The weight vector update term of the NLMF algorithm is normalized by the fourth power of the regression [34], which improves the stability of this algorithm against the increased power and unboundedness of input disturbances. This algorithm shows that even though the input power increases, it remains stable. But the main thing is that the approximation of the value of step size has to be chosen.

Here, we have been using LMF and NLMF adaptive algorithms in the L filter and Ll (C) filter for image restoration purposes. A constrained algorithm [23] has been incorporated in weight term to achieve faster convergence and global optimum value. The proposed paper is organized as follows: Following the introduction, section-2 provides the cellular structure of window chosen for input data of distorted image and application of LMF and NLMF algorithm for L and Ll filter. Section-3 provides results achieved by those filters using the above algorithms and concludes by a conclusion in section-4.

2.1.Use of LMF and NLMF algorithm on L filter Structures:

Here, we have been considering the location invariant least mean fourth(LMF) based L filter (LMF-L) structure for non-constant signal corrupted with zero mean additive white noise/non- Gaussian noise **n**(c), which can be mathematically represented as

$$\mathbf{x}(c) = \mathbf{s}(c) + \mathbf{n}(c) \qquad (1)$$

Here $\mathbf{x}(c)$ is a corrupted image of size $p \times q$, $\mathbf{n}(c)$ is an additive noise added to pixels, and $s(c)$ is a pure (without noise) image. Where, c= (c_1, c_2) denotes the pixel co-ordinate. In image processing applications, we have taken a filter window around the neighbourhood of the pixel c. The window size is taken as $I_1 \times I_2$, where $I_1 = 2i_1 + 1$, $I_2 = 2i_2 + 1$and $i_1 = i_2 = 1,2.....$

The cellular structure of window can be represented as

$$\mathbf{x}(c) = \begin{bmatrix} x(c_1 - i_1, c_2 - i_2) & x(c_1 - i_1, c_2 - i_2 + 1) & ..... & x(c_1 - i_1, c_2 + i_2) \\ . & . & . & . \\ . & . & . & . \\ x(c_1 + i_1, c_2 - i_2) & x(c_1 + i_1, c_2 - i_2 + 1) & ..... & x(c_1 + i_1, c_2 + i_2) \end{bmatrix} \qquad (2)$$

We can arrange the window data into a row matrix format as

$$\mathbf{x}(c) = \left[x(c_1 - i_1, c_2 - i_2) \;, x(c_1 - i_1, c_2 - i_2 + 1) ... \quad x(c_1 + i_1, c_2 + i_2)\right]^T \qquad (3)$$

where, $I_1 \times I_2 = N$ is the numbers of data available within the window.Equation (3) can be written as

$$\mathbf{x}(c) = [x_1(c), x_2(c), \ldots\ldots\ldots\ldots x_N(c)]^T \quad (4)$$

One can represent the eqn.(4) in an ordered vector form as:

$$\mathbf{x}_m(c) = \left(x_{(1)}(c), x_{(2)}(c), \ldots\ldots\ldots x_{(N)}(c)\right)^T \quad (5)$$

where, $x_{(1)}(c) \le x_{(2)}(c) \le \ldots\ldots\ldots \le x_{(N)}(c)$. Here, the bold letters indicate the vectors and $(\,)^T$ denotes the transpose of a vector. The corresponding output of the L filter is equal to the weighted sum of the ordered inputs

$$u(c) = \sum_{i=1}^{N} w_i(c) x_i(c) = \mathbf{w}^T x \quad (6)$$

where, $\mathbf{w}(c) = \left(w_1(c), w_2(c), \ldots\ldots w_N(c)^T\right)$,

weight vector of L filter with constraint $\mathbf{w}^T I = I^T \mathbf{w} = 1$ and $I = [1,1\ldots\ldots 1]^T$ is the $N \times 1$ unit vector. The non-linear output at the pixel ($c=c_1, c_2$) is given as

$$y(c) = f(u(c)) = \frac{1}{1 + e^{-u(c)}} \quad (7)$$

The cost function of LMF algorithm is taken as

$$E(c) = \sum_c e^4(c) \quad (8)$$

where, $e(c) = s(c) - y(c)$, s(c) is a non-corrupted image signal at pixel c, and e(c) is the error signal at pixel c. The weight updation procedure for this algorithm can be expressed as

$$\mathbf{w}(c+1) = \mathbf{w}(c) - \eta_w \nabla E(c) \quad (9)$$

where, η is the learning rate parameter. The above equation (10) is expressed as

$$\mathbf{w}(c+1) = \mathbf{w}(c) + 4\eta e^3(c) f'(u(c)\mathbf{x}(c) \quad (10)$$

The constraint on the weight as $\mathbf{w}^T I = 1$ has been imposed, we can rearrange the weight vector and input vector as

$$\mathbf{w}(c) = (w_1(c), \ldots\ldots w_{\gamma-1}(c) \mid w_\gamma(c) \mid w_{\gamma+1}(c), \ldots\ldots w_N(c))^T, \quad 11(a)$$

$$\mathbf{x}_m(c) = (x_1(c), \ldots\ldots x_{\gamma-1}(c) \mid x_\gamma(c) \mid x_{\gamma+1}(c) \ldots\ldots x_N(c))^T, \quad 11(b)$$

where, $\gamma = \frac{N+1}{2}$.

The modified weight vectors and input vectors can be expressed as

$$\mathbf{wt}(c) = (w_1(c), w_2(c) \ldots w_{\gamma-1}(c), w_{\gamma+1}(c) \ldots\ldots w_N(c)),^T \quad 12(a)$$

$$\mathbf{xt}(c) = (x_{(1)}(c), x_{(2)}(c) \ldots\ldots x_{(N)}(c)), \quad 12(b)$$

Where, $i = (1), (2), (3), \ldots\ldots\ldots (N)$ and $i \neq \gamma$.

The updated vector as per equation (10) can be written as

$$\mathbf{wt}_i(c+1) = \mathbf{wt}_i(c) + 4\eta e^3(c) f'(u(c)) \mathbf{xt}_i(c) \quad 13(a)$$

for i=(1), (2), ……., (N) except i=γ

$$w_\gamma(c+1) = = 1 - \sum_{i=(1)}^{(N)} wt_i(c+1) \quad \text{for} \quad i \neq \gamma \qquad 13(b)$$

The deviation of weight vectors from the optimal weights **h**(c), which is responsible for the generation of noise-free image **s**(c) is **v**(c)=**h**(c)-**w**(c). The mean square deviation of the above equation can be expressed as the expectation value of $v(c)$ i.e. $\|\mathbf{v}(c)\|_2 = \sqrt{\mathbf{v}^T(c)\mathbf{v}(c)}$.

From the above equation, the learning parameter η remains between 0 and 2 i.e. $0 < \eta < 2$ [28].

The above weight updation formula of the LMF algorithm has an instability problem, which depends on the input power, noise power, and initial setting of weights. To improve this algorithm's stability, the normalized versions of the LMF algorithm have come into practice [27-30]. In NLMF algorithm the weight vector updation term (equation (13)) has been normalized by the fourth power of its regressor as [28].

$$\mathbf{wt}_i(c+1) = \mathbf{wt}_i(c) + \frac{4\eta e^3(c) f'(u(c))\mathbf{xt}_i(c)}{\|\mathbf{x}_m(c)\|^2 \left(\|\mathbf{x}_m(c)\|^2 + e^2(c)\right)} \qquad 14(a)$$

for $i = (1),(2),(3).....,(N)$, $i \neq \gamma$ and

$$wt_\gamma(c+1) = = 1 - \sum_{i=1}^{(N)} wt_i(c+1), \qquad 14(b)$$

where, $i \neq \gamma$.

At any circumstance, the input power $\|\mathbf{x}_m(c)\|^2$ becomes zero equation 14(a) diverges. This equation can be modified by adding small number 'δ' in the denominator of the equation 14(a). So the weight updation term becomes.

$$\mathbf{wt}_i(c+1) = \mathbf{wt}_i(c) + \frac{4\eta e^3(c) f'(u(c)\mathbf{xt}_i(c)}{\|\mathbf{x}_m(c)\|^2 \left(\|\mathbf{x}_m(c)\|^2 + e^2(c)\right) + \delta} \qquad 15(a)$$

for i= (1), (2), (3)…….. (N) and $i \neq \gamma$.

$$wt_\gamma(c+1) = = 1 - \sum_{i=1}^{N} wt_\gamma(c+1) \qquad 15(b)$$

The weight vector can be represented as $\mathbf{w}(c+1) = \left(wt_1(c+1), wt_2(c+1), ...wt_{\gamma-1}(c+1), wt_\gamma(c+1), wt_{\gamma+1}(c+1), ..wt_N(c+1)\right)^T$ or

$\mathbf{w}(c) = (w_1(c), w_2(c), ...w_{\gamma-1}(c), w_\gamma(c).w_{\gamma+1}(c), ...w_N(c))^T$. 15(c)

The value of δ is chosen such that $\delta << E\left(\|X(c)^4\|\right)$.

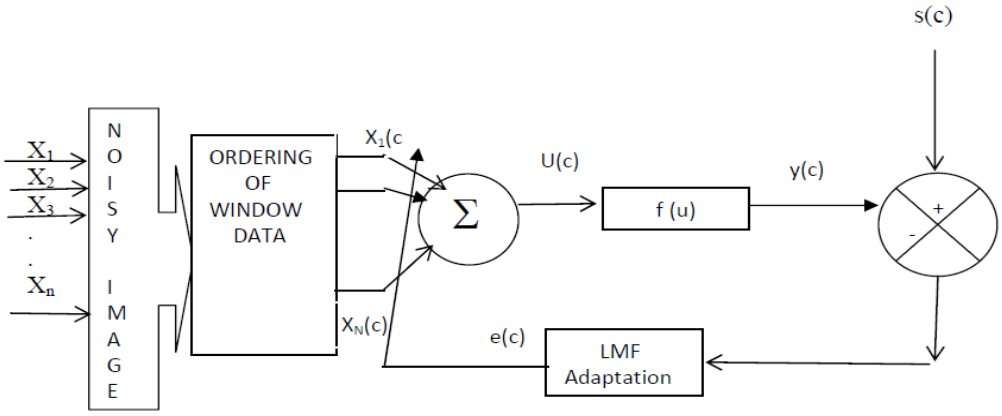


Fig.1.Block diagram for Adaptive-L order-statistic filter.

The pseudo-code for this non-linear NLMF based L filter is presented here.

2.2. Algorithm for LMF based L-filter:

Step-1: Let the input image size be $p \times q$, where p and q are numbers of rows and columns.

Step-2: The weight matrix w and step size parameter ɳ are to be initialized. The Initial cost function has been taken to be zero i.e. E=0.

Step-3: The original image s is corrupted with Gaussian or non-Gaussian noise 'n(c)' on each pixel. So the intensity of corrupted image becomes $\mathbf{x(c) = s(c) + n(c)}$.

Step-4: By choosing a small cellular window of the image to be input for the proposed algorithm. The size of the window be $I_1 \times I_2$.

Where, $I_1 = 2 \times i_1 + 1$, $I_2 = 2 \times i_2 + 1$ and $i_1 = i_2 = 1,2,3....$

Step-5: The cellular window data is to be converted for the input vector as N×1.Where,

N=I1×I2 as $\mathbf{x}(c) = [x_1(c), x_2(c), \ldots\ldots\ldots\ldots x_N(c)]^T$ and c= (c1, c2).

Step-6: $\mathbf{x}(c)$ vector is arranged in ascending order as

$$\mathbf{x}_m = \left\lfloor x_{(1)}(c) < x_{(2)}(c) \ldots\ldots\ldots\ldots\ldots < x_{(N)}(c) \right\rfloor,$$

Input vector x is ordered for L filter as

$$\mathbf{x}_m(c) = \left\lfloor x_{(1)}(c), x_{(2)}(c) \ldots\ldots x_{(\gamma-1)}(c) \,|\, x_{(\gamma)}(c) \,|\, x_{(\gamma+1)}(c) \ldots\ldots\ldots < x_{(N)}(c) \right\rfloor$$ Weight vector w(c) for L filter is as:

$$\mathbf{w}(c) = \left\lfloor w_1(c), w_2(c) \ldots\ldots w_{\gamma-1}(c) \,|\, w_\gamma(c) \,|\, w_{\gamma+1}(c) \ldots\ldots\ldots w_{(N)}(c) \right\rfloor$$

Step-7: The modified input vector and weight vector can be expressed a

$$\mathbf{xt}(c) = \left(x_{(1)}(c), x_{(2)}(c), x_{(3)}(c) \ldots\ldots x_{\gamma-1}(c), x_\gamma(c), x_{\gamma+1}(c), \ldots x_{(N)}(c)\right)^T$$

$$\mathbf{wt}(c) = \left(w_{(1)}(c), w_{(2)}(c), w_{(3)}(c) \ldots\ldots w_{\gamma-1}(c), w_{\gamma+1}(c), w_{\gamma+1}(c), \ldots\ldots w_{(N)}(c)^T\right)$$

Step-8: Numbers of iteration is to be specified (say max).

Step-9: For each pixel, a cellular window is framed, the number of windows for the whole image is $R = p \times q$.

Step-10: For the training process, the evaluation is carried out at each pixel 'c' until the Rth of pixels and means forth error is calculated.

The following Parameters are calculated as:

$$u(c) = \sum_{i=1}^{N} w_i(c) x_i(c) = \mathbf{w}^T \mathbf{x}_m$$

$$y(c) = f(u(c)) = \frac{1}{1 + e^{-u(c)}}$$

$$e(c) = d(c) - y(c)$$

$$E = E + e^4(c)$$

Step-10: Weight vector is updated according to formulas for adaptive LMF-L filter as $\mathbf{wt}_i(c+1) = \mathbf{wt}_i(c) + 4\eta e^3(c)\mathbf{xt}_i(c)$ and

$$\mathbf{wt}_i(c+1) = \mathbf{wt}_i(c) + \frac{4\eta e^3(c) f'(u(c))\mathbf{xt}_i(c)}{\|\mathbf{x}_m(c)\|^2 \left(\|\mathbf{x}_m(c)\|^2 + e^2(c)\right) + \delta}$$ for NLMF filter. For i=1,2,……,N except i=γ,

$$wt_\gamma(c+1) = 1 - \sum_{i=1}^{N} wt_i(c+1)$$.

Hence the vector can be framed as per equation 16(c)

$$\mathbf{w}(c+1) = [w_1(c+1), w_2(c+1), \ldots\ldots w_{\gamma-1}(c+1), w_\gamma(c+1), w_{\gamma+1}(c+1), \ldots\ldots w_N(c+1)]^T$$

Step-11: If the mean fourth error for total numbers of pixels are calculated, than go to step-12.

Step-12: If the MFE satisfies the termination criteria, then proceed to step-13, else go to step-5.

Step-13: The following parameters for the evaluation of images, like MFE, MSE, PSNR, and SSIM are to be calculated

$$MFE = \frac{E}{p \times q}, \quad MSE = \left(\frac{sqr}{p \times q}\right), \quad PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right)$$

$$SSIM = \left(\frac{(2\mu_x\mu_y + \tilde{c}_1)(2\sigma_{xy} + \tilde{c}_2}{(\mu_x^2 + \mu_y^2 + \tilde{c}_1)(\sigma_x^2 + \sigma_y^2 + \tilde{c}_2)}\right)$$

Step-14: Restored image is to be obtained by convolving updated weight vector on a noisy image.

Step-15: The graph between mean fourth errors versus number of iteration is plotted.

2.3. Use of LMF and NLMF algorithm on C (Ll) filter

The combination C (Ll) filter has been developed for the restoration of 1-D and 2-D signals corrupted by Gaussian noise and non-Gaussian noise. It has rank order and temporal information of the observation window and yields the measured output. It is a combination of FIR filter and non-linear filter of order-statistics type. It helps to preserve the edges, and details of noisy-image under restoration [14, 17, 18, 25]. The C and generalized C filter produce an improved performance in preserving the edges, details, and improvement in PSNR than the linear filter. The rank order filter is also useful to remove the abrupt changes and transients in signal levels. The

output of the C filter in a finite window is based on the weighting of rank-ordered temporal data, which is responsible for filtering frequency selective type of non-stationary signals. Such types of filters are called as temporal rank order statistics (TROS) filters [25]. The proper design of the C filter is very useful for retaining the desired signal frequency, preserving edges better than linear filters, and reducing the impulsive noise. Taking smaller windows of a noisy image, the input vector and the ordered vector can be expressed as in equation (4) & (5)

$$\mathbf{x}(c) = \left[x_1(c), x_2(c), \ldots\ldots\ldots\ldots x_N(c)\right]^T \text{ And } \mathbf{x}_m(c) = \left[x_{(1)}(c), x_{(2)}(c), \ldots\ldots\ldots\ldots x_{(N)}(c)\right]^T .$$

Where, c is the pixel position i.e. c_1^{th} row and c_2^{th} column pixel. The output of the C-filter for the small chosen window is

$$u(c) = \sum_{i=1}^{N} \mathbf{w}_r(rank(x_i), i)x_i \ . \tag{16}$$

Where $\mathbf{w}_r$ is the rank-ordered weight of dimension $N \times N$ rank vector $\mathbf{r}$ for input signal can be written as

$$\mathbf{r} = (r_1, r_{2,} \ldots\ldots\ldots\ldots r_N) \ . \tag{17}$$

The normalized output u(c) at the pixel 'c' can be written as

$$u(c) = \frac{\sum_{i=1}^{N} \mathbf{w}_r(rank(x_i), i)x_i}{\sum_{i=1}^{N} \mathbf{w}_r(rank(x_i), i)} \ . \tag{18}$$

The non-linear output at the node 'c' for the window, considering equation (16) and (17) can be written as: The error signal at the output for desired signal s(c) at pixel c is e(c) =s(c)-y(c), and the cost function for the LMF algorithm in equation (8) procedure, the change of $y(c) = f(u(c)) = \frac{1}{1+e^{-u(c)}}$ is . Using the method of steepest descent weights for the equation can be expressed as

$$E(c) = \sum_c e^4(c) \quad \mathbf{w}_r(c+1) = \mathbf{w}_r(c) - \eta \frac{\partial E(c)}{\partial \mathbf{w}_r(c)} \ . \tag{19}$$

The above equation can be reduced to

$$\mathbf{w}_r(c+1) = \mathbf{w}_r(c) - 4\eta e^3 f'(u(c))\mathbf{x}(c) \ . \tag{20}$$

Here, δ is a small constant to avoid divergence, when the first term of the denominator becomes zero.

Similarly, the weight updation for rank-order using the normalized LMF(NLMF-C) algorithm can be written as

$$\mathbf{w}_r(c+1) = \mathbf{w}_r(c) + \frac{4\eta e^3 f'(u(c))\mathbf{x}(c)}{\left\|\mathbf{x}_m(c)\right\|^2 \left(\left\|\mathbf{x}_m(c)\right\|^2 + e^2(c)\right) + \delta} . \tag{21}$$

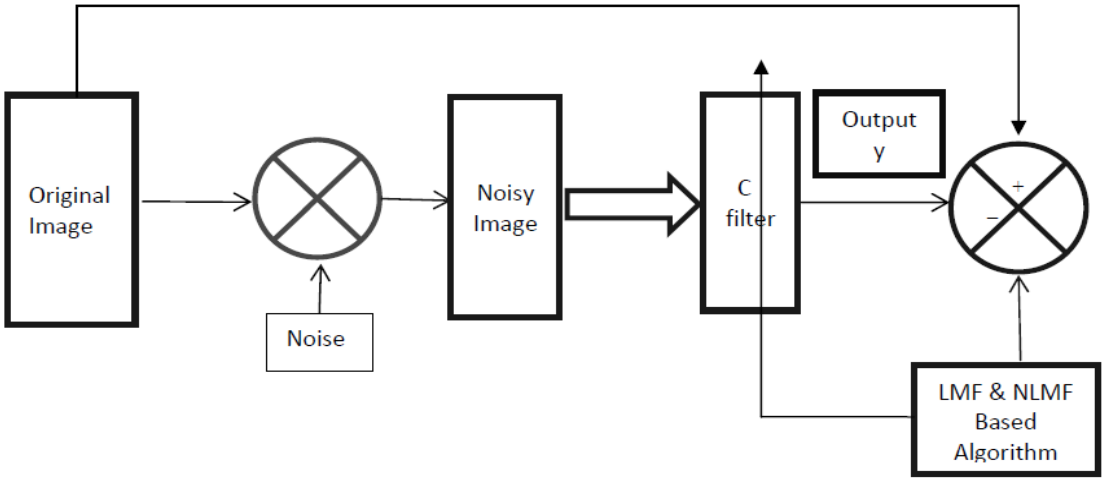


Fig.1.2. Block diagram for adaptive-C order-statistic filters

2.4. Algorithm for LMF based Ll or C filter

Step-1: Let the size of input image be $p \times q$, where p and q are numbers of rows and columns.

Step-2: The weight matrix w and step size and step size parameter ɳ are to be initialized. Initialize the initial cost function E=0.

Step-3: The original image's' is corrupted with Gaussian or non-Gaussian noise n on each pixel. So the intensity of corrupted image becomes $\mathbf{x}(c) = \mathbf{s(c)} + \mathbf{n(c)}$.

Step-4: By choosing a small cellular window of the image to be input for the proposed algorithm. The size of the window be $I_1 \times I_2$, where, $I_1 = 2i_1 + 1$, $I_2 = 2i_2 + 1$, and $i_1 = i_2 = 1,2,3....$

Step-5: The cellular window data is to be converted for the input vector $N \times 1$

$$\mathbf{x}(c) = \left[x_1(c), x_2(c), \ldots\ldots\ldots x_s(c) \ldots\ldots\ldots x_N(c)\right]^T .$$

Step-6: $\mathbf{x}(c)$ vector is arranged in ascending order as $\mathbf{x}_m = \left[x_{(1)}(c) < x_{(2)}(c) \ldots\ldots\ldots\ldots\ldots\ldots < x_{(n)}(c)\right]$

where, c=1 to p×q. The input vector $\mathbf{x}$ is ordered for the C filter as:

$\mathbf{x}(c) = \left[x_{(1)}(c), x_{(2)}(c) \ldots\ldots x_{(\gamma-1)}(c) \,|\, x_{(\gamma)}(c) \,|\, x_{(\gamma+1)} \ldots\ldots\ldots\ldots, x_{(N)}(c)\right]$Step-7: The Rank vector $\mathbf{r}$ for the input signal is to be obtained

$$\mathbf{r} = (r_1, r_2, \ldots\ldots\ldots\ldots r_n)$$

The ranked Order vector $\mathbf{w}_r(c)$ is to be obtained from w matrix

$$\mathbf{w}_r(c) = \mathbf{w}(R(x_i), i)$$

Step-8: The number of iteration is specified (say max).

Step-9: For each pixel, a cellular window is formed, the number of windows for the whole images is p× q for the training process, and the evaluation is carried out at each pixel and the means fourth error is calculated using the following formulae

$$u(c) = \sum_{i=1}^{N} \mathbf{w}_r(c) x_i(c) = \mathbf{w}_r{}^T \mathbf{x}$$

$$u\ (c) = \frac{\sum_{i=1}^{N} \mathbf{w}_r(r(x_i), i) x_i}{\sum_{i=1}^{N} \mathbf{w}(r(x_i, i)}$$

$$y(c) = f(u(c)) = \frac{1}{1+e^{-u(c)}}$$

$$e(c) = d(c) - y(c)$$

$$E = E + e^4(c)$$

Step-10: Weight vector is to be updated according to the formula for the LMF-C filter

$$\mathbf{w}_r(c+1) = \mathbf{w}_r(c) - 4\eta e^3 f'(u(c))\mathbf{x}(c)$$

and for the Normalized LMF-C filter

$$\mathbf{w}_r(c+1) = \mathbf{w}_r(c) + \frac{4\eta e^3 f'(u(c))\mathbf{x}(c)}{\|\mathbf{x}(c)\|^2 (\|\mathbf{x}(c)\|^2 + e^2(c)) + \delta}$$

Step-11: Mean fourth error for total numbers of pixels are calculated than go to step-12

Step-12: If the termination criteria is met than goto step-13 else goto step-9

Step-13: The following parameters for evaluation of images like MFE, PSNR, MSE, and SSIM are to be calculated as

$$MFE = \frac{E}{p \times q}$$

$$MSE = \left(\frac{sqr}{p \times q} \right)$$

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE} \right)$$

$$SSIM = \left(\frac{(2\mu_x\mu_y + \tilde{c}_1)(2\sigma_{xy} + \tilde{c}_2}{(\mu_x^2 + \mu_y^2 + \tilde{c}_1)(\sigma_x^2 + \sigma_y^2 + \tilde{c}_2)} \right)$$

Step-14: Restored image is to be obtained by convolving updated weight vector on a noisy image.

Step-15: Mean fourth errors versus the number of iterations are to be plotted.

3.1. Simulation parameters:

Digital images are corrupted by different types of noises like Gaussian, random, salt pepper (impulse) noise during acquisition, processing, compression, storage, transmission, restoration, and display. The noises are filtered by linear and non-linear filters. The performances of these filters are evaluated with the help of specific parameters like mean square error (MSE), peak signal to noise ratio (PSNR), and structural quality assessment

procedure [28] called structural quality index measurement (SSIM). The simple and widely used image quality measurement method is mean square error, which is computed by averaging the squared intensity of difference of restored and reference image of pixels. This can be represented as

$$MSE = \frac{1}{p \times q} \sum_{c=1}^{p \times q} \left(s(c) - y(c) \right)^2 . \qquad (22(a))$$

Similarly, the mean fourth error can be represented as

$$MFE = \frac{1}{p \times q} \sum_{c=1}^{p \times q} \left(s(c) - y(c) \right)^4 . \qquad (22(b))$$

Where s(c), and y(c) are reference pixel intensity and filtered pixel output at c= (c_1, c2) and p×q is the numbers of pixel points. The peak signal to noise ratio can be written as

$$PSNR = 10 \log_{10} \frac{255^2}{MSE}, \qquad (23)$$

Where, 255 is the maximum value of pixel, normally taken as 255 for gray scale images. The perceptual image quality is evaluated taken in terms of SSIM [35] from the reference image s(c) and restored image y(c) taking the values of window size for adaptation of images

$y(c) = (y_1(c), y_2(c), \ldots\ldots\ldots\ldots y_N(c))^T$ and $s(c) = [s_1(c), s_2(c) \ldots\ldots\ldots s_N(c)]^T$. Then estimating the luminance of each windowed signal of references and observed image, we can take as the mean intensity as $\mu_{\tilde{s}} = \sum_{i=1}^{N} s_i$, $\mu_{\tilde{y}} = \sum_{i=1}^{N} y_i$ and luminance comparison $l(\bar{s}, \bar{y})$ of two functions μ_s and μ_y are considered as $l(\tilde{s}, \tilde{y}) = \frac{2\mu_{\tilde{s}} \mu_{\tilde{y}} + \tilde{c}_1}{\mu_{\tilde{s}}^2 + \mu_{\tilde{y}}^2 + \tilde{c}_1}$, where c_1 is the constant introduced to avoid instability with $\mu_{\tilde{s}}^2 + \mu_{\tilde{y}}^2 = 0$ and $\tilde{c}_1 = (k_1 \times \max)^2$. Where, $k_1 << 1$ (a small constant) and max is the dynamic range pixel (255 for gray scale). The contrast of two images is derived from the standard deviation of the window, which are expressed as $\sigma_{\tilde{s}} = \frac{1}{N} \sum_{i=1}^{N} (\tilde{s}_i - \tilde{\mu}_s)^2$ and $C(\tilde{s}, \tilde{y}) = \frac{2\sigma_{\tilde{s}} \sigma_{\tilde{y}} + \tilde{c}_2}{\sigma_{\tilde{s}}^2 + \sigma_{\tilde{y}}^2 + \tilde{c}_2}$. Where, $\tilde{c}_2$ is a non-negative constant as $\tilde{c}_2 = (k_2 \times \max)^2$ and $k_2 << 1$. The structural similarity of the images is derived from the standard deviation as $O(s, y) = \frac{\sigma_{\tilde{s}} \sigma_{\tilde{y}} + \tilde{c}_3}{\sigma_{\tilde{s}}^2 + \sigma_{\tilde{y}}^2 + \tilde{c}_3}$. Where the covariance between $\tilde{s}$, and $\tilde{y}$ can be estimated as

$$\sigma_{\tilde{s}\tilde{y}} = \frac{1}{N-1} \sum_{i=1}^{N} \left(\tilde{s}_i - \mu_{\tilde{s}} \right) \left(\tilde{y}_i - \mu_{\tilde{s}} \right) \qquad (24)$$

The structural similarity index (SSIM) between two patches of images can be expressed as

$$SSIM(\tilde{s}, \tilde{y}) = \left[l(\tilde{s}, \tilde{y}) \right]^{\alpha} \left[c(\tilde{s}, \tilde{y}) \right]^{\beta} \left[o(\tilde{s}, \tilde{y}) \right]^{\gamma} \quad (25)$$

Where $\alpha > 0, \beta > 0, \gamma > 0$ are used as the relative importance of the above three components. The SSIM measures the nature of similarity between two images. When SSIM=1, it perfectly matches and SSIM<1, it is bounded rule; it gives the Order of matching with the reference image.

3.2 Simulation results discussion:

In this paper, the simulation is carried out on the Lena image, with the LMF based L and C filters, where the reference image is available. The image of Lena is distorted by salt and pepper noise, Gaussian noise, and speckle noise of different strengths. The filter coefficients are initialized randomly such that their sum is unity. The entire noise image is used to train the adaptive filter using the LMF and the NLMF algorithms, which are stable as compared to LMS, NLMS filters. Here, the learning parameters to the above algorithms remain the same in all experiments.

3.3. Image corrupted with salt and pepper Noise:

For three different sets of noise images, the Lena image is corrupted by 10dB, 5dB, and 0dB of salt & pepper noise separately. A small portion of the noise image at the edges and a homogeneous part of the image are taken for training purposes. The cost function for training purposes is taken as the fourth power of the error i.e. the LMF algorithm. As the LMF algorithm provides better performance than the LMS algorithm in the case of transient and steady-state performance of the adaptive filter. But LMF algorithm has several stability problems. So the use of this algorithm limits the application in signal and image processing. The normalized LMF (NLMF) algorithm gives the global minimum in the least mean fourth of its error and provides the remedy for its stability case. In the NLMF algorithm, the weight vector updation is normalized by dividing by the fourth power of norm of the regressor or by the product of the second power of the regressor and sum of the second power of the regressor and square of the error term. From Table-1, it has been observed that in the case of LMS and NLMS filters for different step sizes (η values), the MSE diverges, and PSNR produces low values. But when or less than 0.1, MFE, PSNR, and SSIM, performs better than and. Noisy images and restored images using LMS and LMF algorithms are shown in Fig.2. LMF-L and LMF-C filters provide better results in comparison to LMF filters, as shown in Fig.7 and Table.2. Fig.3 shows that MSE for LMS algorithm diverges, whereas, for NLMS, LMF, LMF-L, and LMF-C filters, it converges. Figures (4 to 6) show that the MFE characteristics for the normalized version of LMF, LMF-L, and LMF-C algorithms. From the convergences of the curve, we may conclude about the stability of the proposed algorithms.

Table-1. Mean square error for different values step-size parameter for Adaptive-LMS and NLMS filter

Filter type	η=0.1		η=0.5		η=0.7	
	MSE	PSNR	MSE	PSNR	MSE	PSNR
LMS	0.0015	28.040	0.0072	21.380	NAN	NAN
NLMS	0.001	28.01	0.005	21.82	0.004	21.07

LMF-L, and LMF-C provide better results than LMF filters as shown in the Fig.4 and Table 2. While NLMF performs better stability with respect to η values as shown in Table-2 and output in Fig.5. For step size η=0.1, better values of MFE, PSNR and SSIM are obtained, in comparison to η=0.5, 1 and 1.9 for LMF filter. Whereas in case of NLMF for η=0.1 to 1.9 MFE, PSNR, and SSIM values remain closer but in some cases, these values remain the same values which reflect the stability of normalized version of L, and C of the LMF filters as the values are tabulated in Table-2. Fig.7 shows that noisy image can be restored with better visuality using NLMF-C filter as PSNR=39.79dB, and SSIM=0.92 compared to NLMF-L filter, as it gives PSNR=38.12dB and SSIM=0.90, where as in case of NLMF PSNR=32.04 and SSIM=0.79. When, the noise increases in the distorted images, PSNR values decrease as seen from Table 2. It can be observed that for increasing order of step size parameter η=0.1 to 1.9, PSNR and SSIM slightly decrease with increase in MFE but variation is very small which reflects stability of the algorithm. For NLMF-L filter similar performance are observed but, in case of NLMF-C filter it has been observed better performance than LMF-L filter.

3.4. Image corrupted with Gaussian and speckle noise

In Gaussian noise with noise strength of SNR=10dB, LMF-L filter restores image with PSNR=34.37dB, and SSIM=0.84, while LMF-C filter restores with a better quality of vision with PSNR=35.11dB, and SSIM=0.84. Simulations are carried out for Gaussian and speckle noises for different strengths of noises. For different step

sizes (η), it has been observed better visuality in the case of normalized LMF (NLMF) version of C-filter than L filter, and those perform better in case of Gaussian noise. NLMF-L filter restores image with PSNR=35.52 and SSIM=0.86, whereas NLMF-C filter with PSNR=36.54 and SSIM=0.87, for Gaussian noise of strength, SNR=10dB (Table-3). For stronger noise strengths such as 5dB and 0dB of SNR, the output obtained with less PSNR and SSIM is tabulated in Table-3. Similarly, in the case of speckle noise, for normalized LMF-L (NLMF-L) filter, the noisy image is restored with PSNR=35.52 and SSIM=0.86, and for normalized LMF-C filter, it is observed as PSNR=36.54dB and SSIM= 0.87 with noise strength SNR=10dB as shown in Table-4.

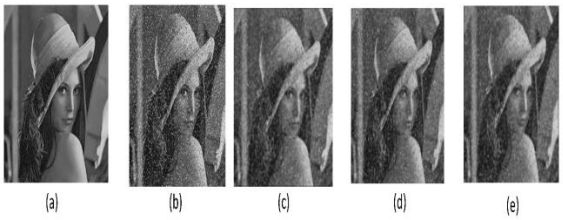


Fig.2.(a) Pure image, (b) noisy image, (c) restored using LMS filter,(d) restored using NLMS filter, (e) restored using LMF filter.

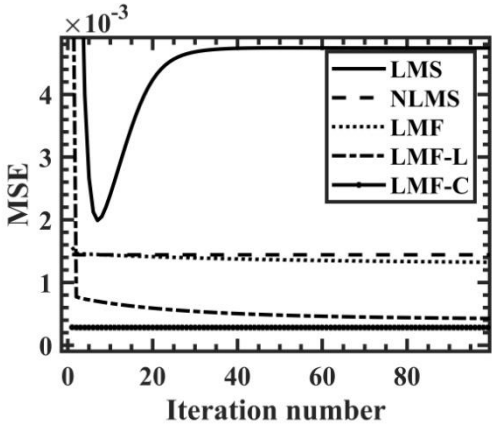


Fig.3. Mean square error (MSE) Characteristics for LMS, NLMS, LMF, LMF-L, LMF-C filters

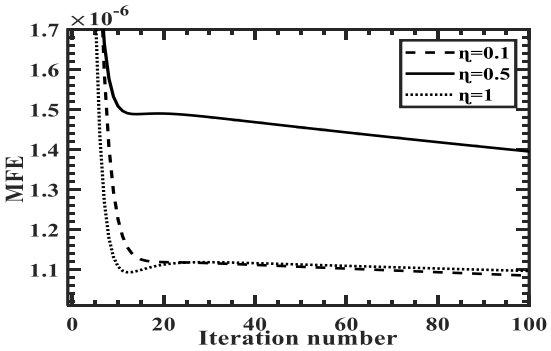


Fig.4 Mean fourth error (MFE) characteristics for Normalized LMF filter for different values of step size (η).

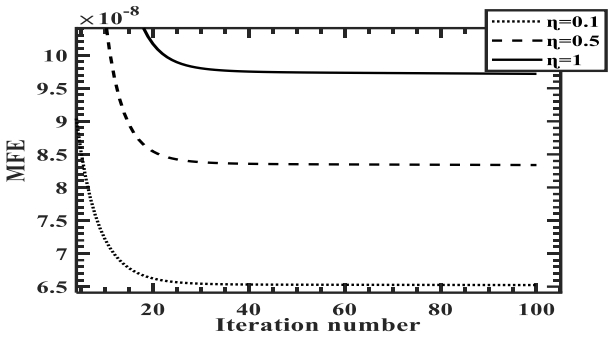


Fig.5. Mean fourth error (MFE) characteristics for Normalized LMF-L filter for different value of step size (η).

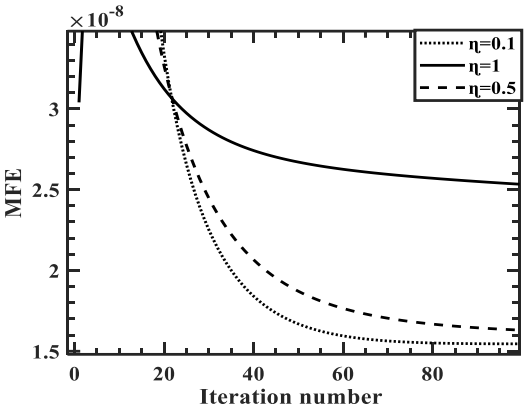


Fig.6. Mean fourth error (MFE) characteristics for Normalized LMF-C filter for different value of step size (η).

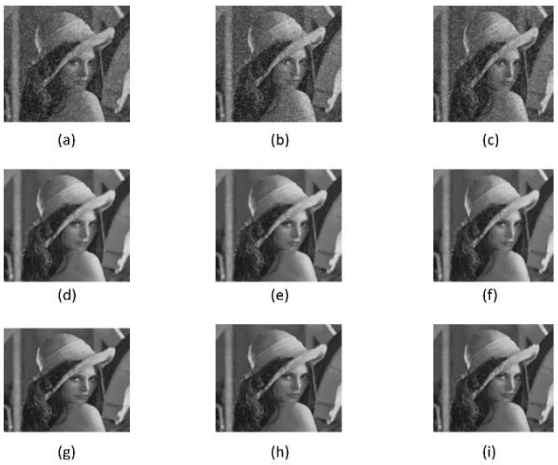


Fig.7. Image restored using (a) NLMF filter with η=0.1, (b) NLMF filter with η=0.5, (c) NLMF filter with η=1, (d) NLMF-L filter with η=0.1, (e) NLMF-L filter with η=0.5, (f) NLMF-L filter with η=1, (g) NLMF-C filter with η=0.5, (h) NLMF-C filter with η=1.

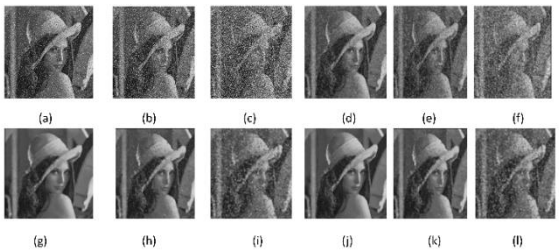


Fig.8.Noisy and restored images: Salt and pepper noisy images with noise strength (a) SNR=10dB, (b) SNR=5dB, (c) SNR=0dB. Restored images (d) for SNR=10db using NLMF, (e) for SNR=5dB using NLMF, (f) for SNR=0dB using NLMF, (g) for SNR=10db using NLMF-L, (h) for SNR=5dB using NLMF-L, (i) for SNR=0dB using NLMF-L, (j) for SNR=10dB using NLMF-C, (k) for SNR=5dB using NLMF-C, (l) for SNR=0dB using NLMF-C, filters.

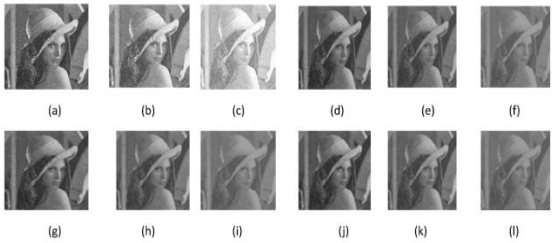
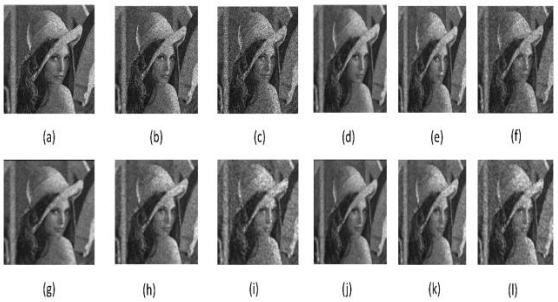


Fig.9.Noisy and restored images: Gaussian noisy images with noise strength (a) SNR=10dB, (b) SNR=5dB, (c) SNR=0dB. Restored images (d) for SNR=10db using NLMF, (e) for SNR=5dB using NLMF, (f) for SNR=0dB using NLMF, (g) for SNR=10dB using NLMF-L, (h) for SNR=5dB using NLMF-L, (i) for SNR=0dB using NLMF-L, (j) for SNR=10dB using NLMF-C, (k) for SNR=5dB using NLMF-C, (l) for SNR=0dB using NLMF-C, filters



. Noisy and restored images: Speckle noisy images with noise strength (a) SNR=10dB, (b) SNR=5dB, (c) SNR=0dB. Restored images (d) for SNR=10db using NLMF, (e) for SNR=5dB using NLMF, (f) for SNR=0dB using NLMF, (g) for SNR=10db using NLMF-L, (h) for SNR=5dB using NLMF-L, (i)for SNR=0dB using NLMF-L, (j) for SNR=10db using NLMF-C, (k) for SNR=5dB using NLMF-C, (l) for SNR=0dB using NLMF-C, filters.

4. Conclusion:

This paper reflects a comparison of performance LMS, NLMS, LMF, NLMF, LMF-L, NLMF-L, LMF-C, NLMF-C filters performance in image restoration and stability of the algorithms in terms of minimum mean fourth of error. In the LMS algorithm, the restored image is inferior as compared with the algorithms like LMF, NLMF. When, the images are corrupted with different types of noises like Gaussian, salt & pepper, and speckle noise of different strengths, SNR values 10dB, 5dB, 0dB. The weight updation criterion is based upon the least mean fourth algorithm. By taking a small patch of pixels near to the edge and homogeneous region, the LMF based C-filters provide better evaluation parameters like MFE, PSNR, and SSIM in comparison to filters using least mean square and least mean fourth algorithms, when trained under similar conditions of identical initial weights and step size. In these cases, the weight vector is constrained to 1 as in the case of the median filter. The restored image provides better vision in the case of the NLMF-C filter when compared with other restored images.

Tabl-2. Different simulated parameters like MFE, PSNR, and SSIM for efficient evaluation of filters at different learning rates(η)when the image is corrupted with salt & pepper noise of different strength (10dB, 5dB, 0dB).

Filter type	Value Of η	SNR=10dB			SNR=5dB			SNR=0dB		
Salt & pepper Noise		MFE	PSNR	SSIM	MFE	PSNR	SSIM	MFE	PSNR	SSIM
LMF	0.1	1.7*10^(-6)	30.53	0.78	8.3*10^(-6)	27.02	0.69	4.9*10^(-5)	21.65	0.34
	0.5	1..8*10^(-6)	30.25	0.78	9.8*10^(-6)	26.37	0.67	9.9*10^(-4)	18.64	0.28
	1	6.1*10^(-6)	27.46	0.75	9.9*10^(-6)	25.97	0.59	1.4*10^(-4)	18.12	0.27
	1.9	7.8*10^(-6)	26.93	0.72	9.5*10^(-6)	25.13	0.57	1.8*10^(-4)	18.01	0.27
LMF-L	0.1	1.7*10^(-6)	30.53	0.78	8.3*10^(-6)	27.02	0.69	4.9*10^(-5)	21.65	0.34
	0.5	1..8*10^(-6)	30.25	0.78	9.8*10^(-6)	26.37	0.67	9.9*10^(-4)	18.64	0.28
	1	6.1*10^(-6)	27.46	0.75	9.9*10^(-6)	25.97	0.59	1.4*10^(-4)	18.12	0.27
	1.9	7.8*10^(-6)	26.93	0.72	9.5*10^(-6)	25.13	0.57	1.8*10^(-4)	18.01	0.27
LMF-C	0.1	8.0*10^(-8)	38.01	0.90	3.9*10^(-7)	34.42	0.76	3.9*10^(-5)	23.21	0.41
	0.5	1.0*10^(-7)	37.70	0.89	6.0*10^(-7)	33.18	0.73	4.4*10^(-5)	22.45	0.39
	1	1.7*10^(-7)	36.53	0.88	4.3*10^(-7)	33.91	0.73	4.2*10^(-5)	22.64	0.39
	1.9	1.7*10^(-7)	36.51	0.88	5.5*10^(-7)	33.43	0.72	4.8*10^(-5)	22..01	0.37
NLMF	0.1	1.08*10^(-6)	32.04	0.79	8.1*10^(-6)	27.21	0.68	4.8*10^(-5)	21.68	0.34
	0.5	1.39*10^(-6)	32.02	0.78	8.6*10^(-6)	27.10	0.67	4.9*10^(-5)	21.53	0.33
	1	1.09*10^(-6)	32.03	0.79	8.8*10^(-6)	27.04	0.67	4.7*^10(-5)	21.52	0.32
	1.9	1.38*10^(-6)	32.03	0.78	8.8*10^(-6)	27.01	0.66	4.7*^10(-5)	21.56	0.32
NLMF-L	0.1	6.5*10^(-8)	38.12	0.90	2.8*10^(-7)	34.70	0.78	1.8*10^(-5)	24.19	0.43
	0.5	8.3*10^(-8)	37.81	0.88	3.1*10^(-7)	34.25	0.77	2.0*10^(-5)	24.11	0.42
	1	9.7*10^(-8)	37.03	0.87	3.3*10^(-7)	34.07	0.76	2.1*10^(-5)	24.26	0.42
	1.9	9.9*10^(-8)	37.01	0.86	3.3*10^(-7)	34.03	0.76	2.4*10^(-5)	24.12	0.41
NLMF-C	0.1	1.5*10^(-8)	39.79	0.92	2.4*10^(-7)	34.85	0.79	1.5*10^(-5)	25.02	0.43
	0.5	1.63*10^(-8)	39.60	0.92	2.4*10^(-7)	34.81	0.79	1.8*10^(-5)	24.24	0.43
	1	2.53*10^(-8)	38.42	0.90	2.6*10^(-7)	34.76	0.78	1.8*10^(-5)	24.21	0.42
	1.9	2.67*10^(-8)	38.37	0.90	2.5*10^(-7)	34.74	0.78	1.9*10^(-5)	24.12	0.41

Table-3. Different simulated parameters like MFE, PSNR, and SSIM for efficient evaluation of filters at different learning rates (η) when the image is corrupted with Gaussian noise of different strength (10dB, 5dB, 0dB).

Filter type	Value Of η	SNR=10dB			SNR=5dB			SNR=0dB		
Gaussian noise		MFE	PSNR	SSIM	MFE	PSNR	SSIM	MFE	PSNR	SSIM
LMF	0.1	7.5*10^(-7)	33.17	0.81	1.2*10^(-6)	31.75	0.78	4.6*10^(-6)	29.52	0.71
	0.5	7.9*10^(-7)	33.13	0.80	1.4*10^(-6)	31.69	0.77	6.3*10^(-6)	28.37	0.67
	1	9.0*10^(-7)	32.27	0.78	1.7*10^(-6)	30.90	0.77	3.4*10^(-6)	29.30	0.68
	1.9	9.8*10^(-7)	31.34	0.76	2.8*10^(-6)	29.16	0.76	9.4*10^(-6)	27.61	0.63
LMF-L	0.1	2.8*10^(-7)	34.37	0.84	5.1*10^(-7)	32.92	0.81	1.4*10^(-6)	30.12	0.72
	0.5	4.2*10^(-7)	33.82	0.82	9.2*10^(-7)	31.47	0.80	2.7*10^(-6)	29.74	0.74
	1	5.9*10^(-7)	33.55	0.81	9.8*10^(-7)	31.24	0.78	3.8*10^(-6)	28.45	0.73
	1.9	6.4*10^(-7)	32.96	0.81	1.4*10^(-6)	30.65	0.77	4.1*10^(-6)	28.43	0.71
LMF-C	0.1	2.2*10^(-7)	35.11	0.84	4.9*10^(-7)	32.97	0.81	1.1*10^(-6)	31.92	0.72
	0.5	2.9*10^(-7)	34.97	0.84	7.7*10^(-7)	33.10	0.81	1.9*10^(-6)	31.09	0.70
	1	3.5*10^(-7)	34.62	0.83	7.9*10^(-7)	33.02	0.80	2.2*10^(-6)	29.37	0.70
	1.9	4.7*10^(-7)	33.26	0.81	9.3*10^(-7)	31.78	0.79	2.8*10^(-6)	29.12	0.68
NLMF	0.1	4.3*10^(-7)	33.71	0.81	1.0*10^(-6)	31.89	0.78	3.7*10^(-6)	29.92	0.72
	0.5	4.5*10^(-7)	33.05	0.80	1.2*10^(-6)	31.72	0.78	3.5*10^(-6)	29.98	0.72
	1	4.5*10^(-7)	33.02	0.80	1.1*10^(-6)	33.50	0.79	3.7*10^(-6)	28.86	0.71
	1.9	8.7*10^(-7)	32.92	0.79	1.1*10^(-6)	33.50	0.79	3.9*10^(-6)	28.97	0.70
NLMF-L	0.1	1.9*10^(-7)	35.52	0.86	3.1*10^(-7)	34.86	0.81	8.0*10^(-7)	32.50	0.78
	0.5	2.1*10^(-7)	34.97	0.85	3.7*10^(-7)	33.90	0.79	7.8*10^(-7)	32.52	0.78
	1	2.5*10^(-7)	34.68	0.84	4.1*10^(-7)	32.06	0.78	8.6*10^(-7)	31.25	0.77
	1.9	2.5*10^(-7)	34.63	0.84	4.2*10^(-7)	32.01	0.78	8.8*10^(-7)	31.21	0.75
NLMF-C	0.1	1.4*10^(-7)	36.54	0.87	2.7*10^(-7)	34.73	0.83	5.3*10^(-7)	33.28	0.81
	0.5	1.7*10^(-7)	36.25	0.85	2.9*10^(-7)	34.41	0.83	5.5*10^(-7)	33.71	0.80
	1	2.1*10^(-7)	35.74	0.84	3.3*10^(-7)	33.77	0.82	6.3*10^(-7)	32.43	0.79
	1.9	2.0*10^(-7)	35.71	0.84	3.4*10^(-7)	33.71	0.81	6.1*10^(-7)	32.36	0.79

Table-4. Different simulated parameters like MFE, PSNR, and SSIM for efficient evaluation of filters at different learning rates(η) when the image is corrupted with speckle noise of different strength (10dB, 5dB, 0dB).

FILTER TYPE	Value Of η	SNR=10dB			SNR=5dB			SNR=0dB	
Speckle Noise		MFE	PSNR	SSIM	MFE	PSNR	SSIM	MFE	PSNR
LMF	0.1	1.5*10^(-6)	30.42	0.73	5.9*10^(-6)	27.72	0.69	2.3*10^(-5)	24.08
	0.5	4.6*10^(-6)	29.12	0.71	6.5*10^(-6)	27.23	0.64	4.7*10^(-5)	21.81
	1	7.4*10^(-6)	27.89	0.67	1.6*10^(-5)	25.69	0.59	8.3*10^(-5)	21..12
	1.9	7.9*10^(-6)	27.19	0.66	1.9*10^(-5)	25.34	0.59	8.8*10^(-5)	21.34
LMF-L	0.1	6.7*10^(-7)	32.49	0.78	7.8*10^(-6)	27.12	0.67	1.1*10^(-5)	25.21
	0.5	2.4*10^(-6)	29.70	0.72	9.8*10^(-6)	26.58	0.63	1.0*10^(-5)	25.97
	1	9.0*10^(-7)	32.13	0.77	8.0*10^(-6)	27.01	0.64	5.0*10^(-5)	22.97
	1.9	4.6*10^(-6)	29.06	0.72	9.6*10^(-6)	26.82	0.63	5.9*10^(-5)	22.23
LMF-C	0.1	3.9*10^(-7)	34.10	0.81	1.4*10^(-6)	30.99	0.72	1.0*10^(-5)	26.54
	0.5	6.2*10^(-7)	33.62	0.79	6.7*10^(-6)	28.66	0.67	1.5*10^(-5)	25.69
	1	6.3*10^(-7)	32.94	0.80	7.3*10^(-6)	27.32	0.67	3.8*10^(-5)	23.96
	1.9	7.2*10^(-7)	32.07	0.77	7.1*10^(-6)	27.28	0.66	4.4*10^(-5)	23.63
NLMF	0.1	1.3*10^(-6)	30.78	0.74	4.3*10^(-6)	28.38	0.71	2.1*10^(-5)	25.01
	0.5	1.5*10^(-6)	30.70	0.74	4.5*10^(-6)	28.21	0.70	2.4*10^(-5)	24.84
	1	1.8*10^(-6)	30.12	0.73	4.2*10^(-6)	28.34	0.70	2.5*10^(-5)	24.76
	1.9	1.8*10^(-6)	30.07	0.73	4.2*10^(-6)	28.34	0.70	2.9*10^(-5)	23.67
NLMF-L	0.1	5.8*10^(-7)	33.82	0.80	1.2*10^(-6)	31.04	0.72	1.1*10^(-5)	26.22
	0.5	5.9*10^(-7)	33.69	0.79	1.4*10^(-6)	30.96	0.72	1.3*10^(-5)	26.01
	1	6.1*10^(-7)	32.55	0.78	1.8*10^(-6)	29.98	0.71	1.2*10^(-5)	25.96
	1.9	6.3*10^(-7)	32.47	0.78	1.8*10^(-6)	29.98	0.71	1.2*10^(-5)	25.90
NLMF-C	0.1	2.4*10^(-7)	35.15	0.84	1.01*10^(-6)	31.43	0.73	1.0*10^(-5)	26.57
	0.5	2.6*10^(-7)	35.08	0.83	1.05*10^(-6)	31.22	0.73	1.0*10^(-5)	26.43
	1	3.9*10^(-7)	34.92	0.82	1.1*10^(-6)	31.16	0.72	1.1*10^(-5)	25.27
	1.9	3.9*10^(-7)	34.87	0.82	1.2*10^(-6)	31.06	0.72	1.7*10^(-5)	25.23

References

[1] A. H. Sayed, Fundamentals of Adaptive Filtering. John Wiley & Sons, 2003.

[2] S. S. Haykin, Adaptive Filter Theory. Pearson Education India, 2008.

[3] M. Bellanger, Adaptive digital filters and signal analysis. New York: M. Dekker, 1987.

[4] B. Widrow and S. D. Stearns, Adaptive signal processing. Englewood Cliffs, N.J.: Prentice-Hall, 1985.

[5] N.Kalouptsidis and S.Theodoridis, EdsAdaptive system Identification and signal processing Algorithm London, U.K:Printice-Hall Int, 1993.

[6] I. Pitas and A. N. Venetsanopoulos, Nonlinear Digital Filters: Principles and Applications. Springer Science & Business Media, 2013.

[7] M. Rathod, V. Patel, and N. V. George, "Generalized spline nonlinear adaptive filters," Expert Syst. Appl., vol. 83, pp. 122–130, Oct. 2017, doi: 10.1016/j.eswa.2017.04.043.

[8] V. J. Mathews, "Adaptive polynomial filters," IEEE Signal Process. Mag., vol. 8, no. 3, pp. 10–26, Jul. 1991, doi: 10.1109/79.127998.

[9] F. Palmieri, "A backpropagation algorithm for multilayer hybrid order statistic filters," in International Conference on Acoustics, Speech, and Signal Processing, May 1989, pp. 1179–1182 vol.2, doi: 10.1109/ICASSP.1989.266644.

[10] I. Pitas and S. Vougioukas, "LMS order statistic filters adaptation by backpropagation," Signal Process., vol. 25, no. 3, pp. 319–335, Dec. 1991, doi: 10.1016/0165- 1684(91)90117-2.

[11] Y.- Zhou, R. Chellappa, A. Vaid, and B. K. Jenkins, "Image restoration using a neural network," IEEE Trans. Acoust. Speech Signal Process., vol. 36, no. 7, pp. 1141–1151, Jul. 1988, doi: 10.1109/29.1641.

[12] A. Bovik, "Streaking in median filtered images," IEEE Trans. Acoust. Speech Signal Process., vol. 35, no. 4, pp. 493–503, Apr. 1987, doi: 10.1109/TASSP.1987.1165153.

[13] A. C. Bovik, T. S. Huang, and D. C. Munson, "The Effect of Median Filtering on Edge Estimation and Detection," IEEE Trans. Pattern Anal. Mach. Intell., vol. PAMI-9, no. 2, pp. 181–194, Mar. 1987, doi: 10.1109/TPAMI.1987.4767894.

[14] A. Bovik, T. Huang, and D. Munson, "A generalization of median filtering using linear combinations of order statistics," IEEE Trans. Acoust. Speech Signal Process., vol. 31, no. 6, pp. 1342–1350, Dec. 1983, doi: 10.1109/TASSP.1983.1164247.

[15] F.Palmieri and C.G Boncelet, "Optimal MSE liner combination of order statistics for restoration of Markov processing," Proc. 20th Ann.Conf.info.Sci.syst, Princeton, NJ, Mar.1986.

[16] C.G.Boncelet, "The efficient design of order statistic filters for image restoration,"In Proc.24th Ann. Allrston conf.commun, contr. ,comput, Monticello, Det.1986.

[17] A. Restrepo and A. C. Bovik, "Adaptive trimmed mean filters for image restoration," IEEE Trans. Acoust. Speech Signal Process., vol. 36, no. 8, pp. 1326–1337, Aug. 1988, doi: 10.1109/29.1660.

[18] Yong Lee and S. Kassam, "Generalized median filtering and related nonlinear filtering techniques," IEEE Trans. Acoust. Speech Signal Process., vol. 33, no. 3, pp. 672–683, Jun. 1985, doi: 10.1109/TASSP.1985.1164591.

[19] A.C Barik and L.Naaman, "Least square signal estimation using Order statistics filters," in Proc.20th Ann.Con.Info.Sci.Syst, Princeton, NJ, Mar.1986.

[20] T. Huang, G. Yang, and G. Tang, "A fast two-dimensional median filtering algorithm," IEEE Trans. Acoust. Speech Signal Process., vol. 27, no. 1, pp. 13–18, Feb. 1979, doi: 10.1109/TASSP.1979.1163188.

[21] J. Fitch, E. Coyle, and N. Gallagher, "Median filtering by threshold decomposition," IEEE Trans. Acoust. Speech Signal Process., vol. 32, no. 6, pp. 1183–1188, Dec. 1984, doi: 10.1109/TASSP.1984.1164468.

[22] C. Kotropoulos and I. Pitas, "Adaptive LMS L-filters for noise suppression in images," IEEE Trans. Image Process. Publ. IEEE Signal Process. Soc., vol. 5, no. 12, pp. 1596– 1609, 1996, doi: 10.1109/83.544568.

[23] C. Kotropoulos and I. Pitas, "Constrained adaptive LMS L-filters," Signal Process., vol. 26, no. 3, pp. 335–358, Mar. 1992, doi: 10.1016/0165-1684(92)90119-H.

[24] F. Palmieri and C. G. Boncelet, "Ll-filters-a new class of order statistic filters," IEEE Trans. Acoust. Speech Signal Process., vol. 37, no. 5, pp. 691–701, May 1989, doi:10.1109/29.17561.

[25] P. P. Gandhi and S. A. Kassam, "Design and performance of combination filters for signal restoration," IEEE Trans. Signal Process., vol. 39, no. 7, pp. 1524–1540, Jul. 1991, doi: 10.1109/78.134392.

[26] I. Pitas and A. N. Venetsanopoulos, "Adaptive filters based on order statistics," IEEE Trans. Signal Process., vol. 39, no. 2, pp. 518–522, Feb. 1991, doi: 10.1109/78.80845.

[27] E. Walach and B. Widrow, "The least mean fourth (LMF) adaptive algorithm and its family," IEEE Trans. Inf. Theory, vol. 30, no. 2, pp. 275–283, Mar. 1984, doi: 10.1109/TIT.1984.1056886.

[28] E. Eweda, "Global Stabilization of the Least Mean Fourth Algorithm," IEEE Trans.

[29] Signal Process., vol. 60, no. 3, pp. 1473–1477, Mar. 2012, doi:10.1109/TSP.2011.2177976.

[30] E. Eweda and A. Zerguine, "New insights into the normalization of the least mean fourth algorithm," Signal Image Video Process., vol. 7, no. 2, pp. 255–262, Mar. 2013, doi: 10.1007/s11760-011-0231-y.

[31] S.Koike, "Stability Condition for adaptive algorithms with non-quadratic errors criteria," EUSIPCO-2000, Tempere, Finland pp.131-134(2000).

[32] A. Zerguine, "Convergence behavior of the normalized least mean fourth algorithm," in Conference Record of the Thirty-Fourth Asilomar Conference on Signals, Systems and Computers (Cat. No.00CH37154), Oct. 2000, vol. 1, pp. 275–278 vol.1, doi: 10.1109/ACSSC.2000.910958.

[33] M. K. Chan and C. F. N. Cowan, "Using a normalised LMF algorithm for channel equalisation with co-channel interference," in 2002, 11th European Signal Processing Conference, Sep. 2002, pp. 1–4.

[34] A. Zerguine, "Convergence behavior of the normalized least mean fourth algorithm," in Conference Record of the Thirty-Fourth Asilomar Conference on Signals, Systems and Computers (Cat. No.00CH37154), Oct. 2000, vol. 1, pp. 275–278 vol.1, doi: 10.1109/ACSSC.2000.910958.

[35] E.Ewda and A zerguine, "New insights into the normalization of the least mean fourth algorithm," signal, Image, videoProcess., 2011DOI:10:1007/S 11760-011-0231-y.

[36] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Trans. Image Process., vol. 13, no. 4, pp. 600–612, Apr. 2004, doi: 10.1109/TIP.2003.819861.