Active Noise Control

Design Credits Project

Submitted by:

Rushil Ashish Shah (B20AI036)

Luyum Pegoo (B20CS031)

Under the Supervision of:

Dr. Amrita Puri

Department of Mechanical Engineering

Indian Institute of Technology Jodhpur

Semester IV (2021-22)

Motivation and Goal:

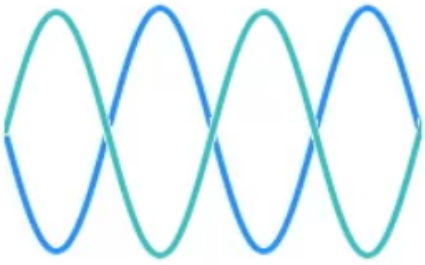
On the basis of literature studied it has been concluded that the comparative analysis of all these algorithms (Linear as well as non-linear) for different noises (Tonal, Multi tonal, Broadband chaotic noise and Impulsive noise) has not been performed and this has become the main motivation behind the present work.
We read the literature about the algorithms assigned ,Implemented them in MATLAB and did a comparative study of all these algorithms .

Introduction:

After the industrial revolution, noise related problems are becoming more prominent. The revolution has led to an increment in the number of equipment like compressors, blowers, fans etc. noise created due to mechanical vibrations also creates issues in the manufacturing industry, transportation and sometimes in household appliances also. Therefore there is a clear need for cancellation/ control of the noise .Broadly there are two noise cancellation techniques : Active Noise Cancellation and Passive noise cancellation. In this project, we have implemented the linear and non-linear techniques for Active Noise Cancellation.
Active Noise cancellation basically involves adding a complementary signal to the corresponding noise signal in order to reduce the magnitude of the noise.

Noise

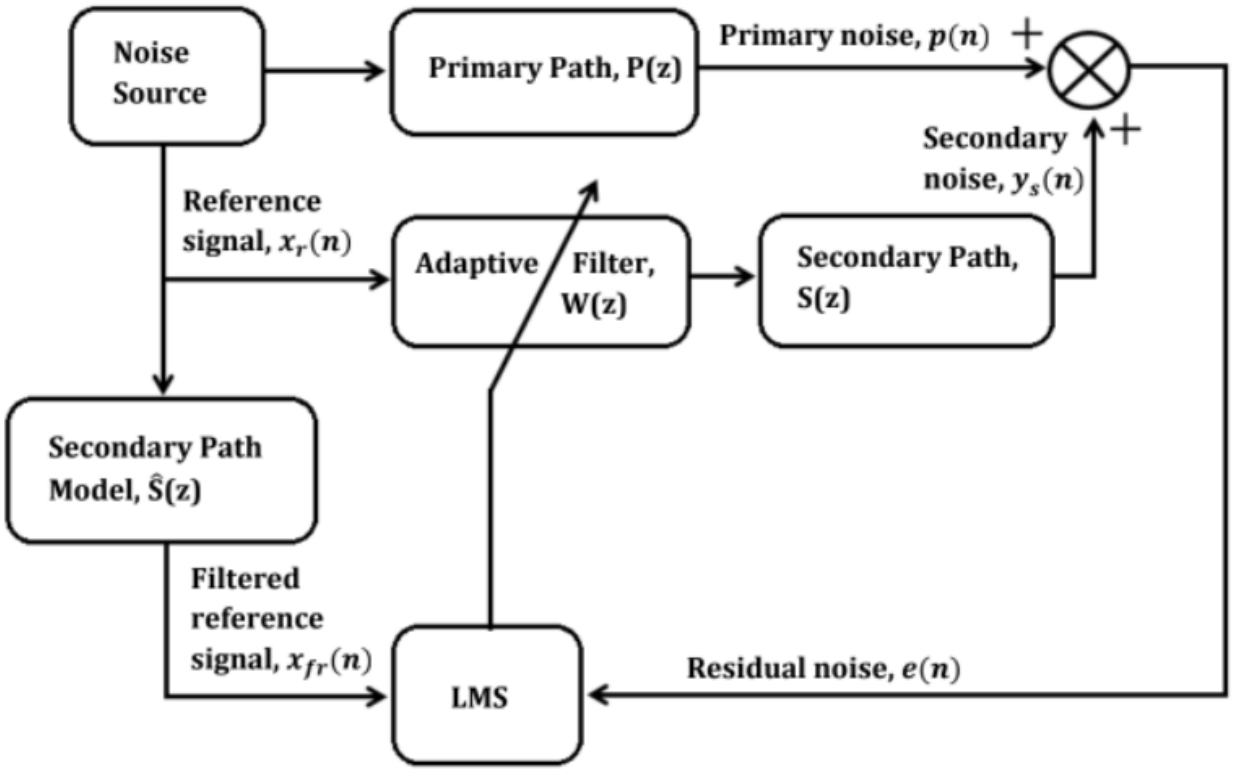
Noise Canceled



As the name suggests, in ANC, we reduce the noise in live time and deliver the best sound experience to the user/observer. The Linear techniques include the Least Mean Square (LMS) and Filtered-X Least Mean Square (Fx-LMS) algorithm, and nonlinear techniques Include several algorithms like Hammerstein FxLMS, Chebyshev FxLMS and Even Mirror fourier Non-Linear. We have presented each of the algorithms in brief in this report

FxLMS Algorithm

The FxLMS algorithm is a widely used algorithm for active noise control (ANC). The Figure below shows the block diagram of the FxLMS algorithm. Primary path is the path from the noise source to the output of the error sensor. Reference signal which has the same frequency components as that of the primary noise is fed to an adaptive filter. The adaptive filter is an FIR (finite impulse response) filter whose output drives the secondary source which generates secondary noise. Interference of the primary and the secondary noise at the location of the error sensor leads to cancellation of sound.



The residual noise (or the error signal) at the nth sample, (n), is a combination of the primary noise, p(n),and the sound generated by the secondary source, ys(n). It can be written as,

$$e(n) = p(n) + y_s(n)$$

The objective function, J(n), is minimized using the steepest descent algorithm which gives the following equation for updating the filter coefficients.

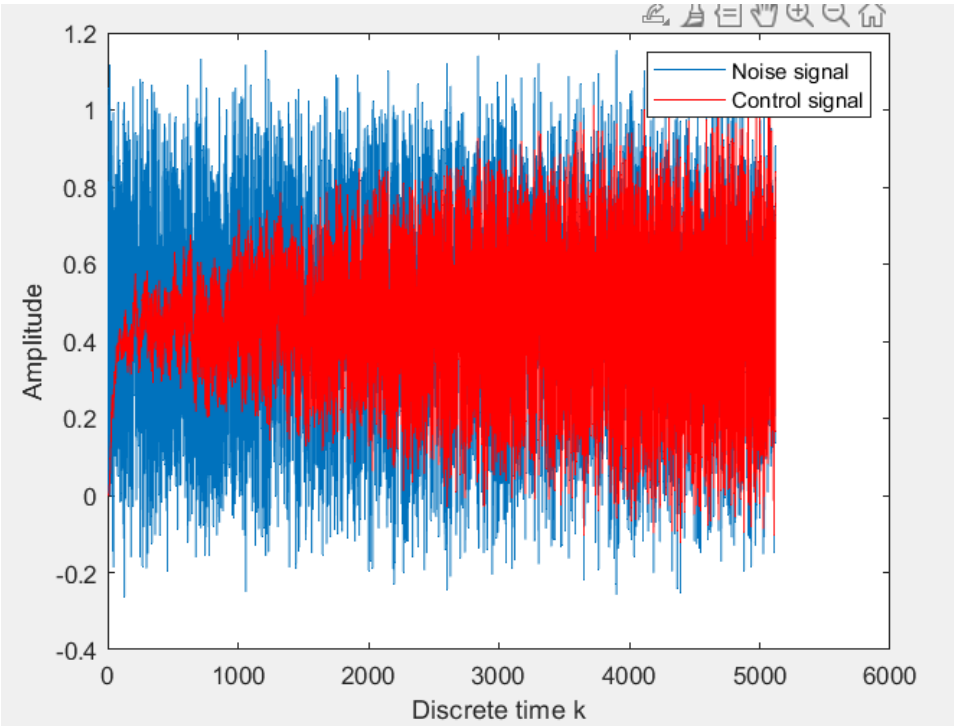
$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) - \frac{\mu}{2}\boldsymbol{\nabla J}(n)$$

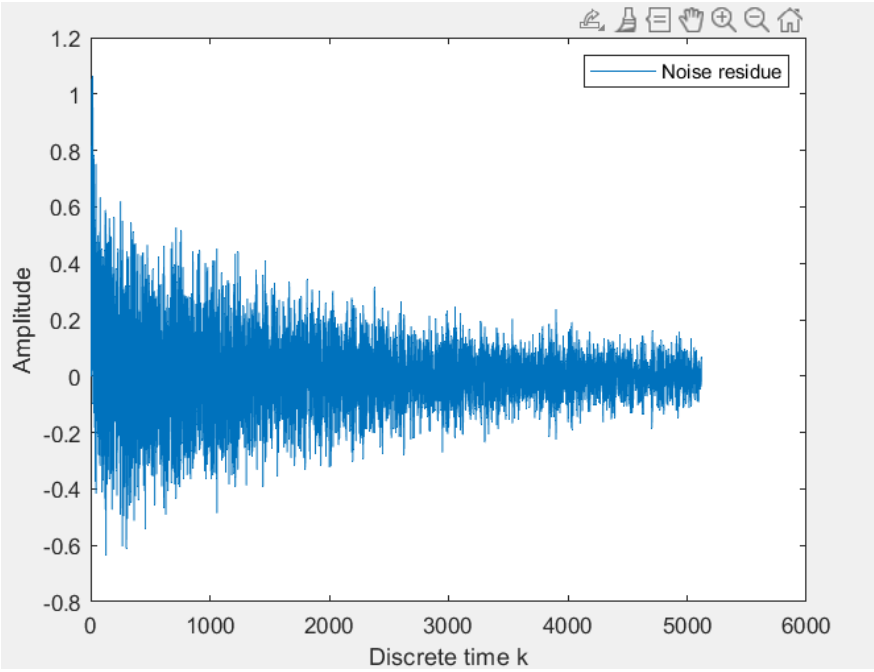
The gradient of the objective function can be written as,

$$\nabla J(n) = 2e(n)\nabla e(n)$$

```
T = 1;
N = 30;
fs = 5120;
Ts = 1/fs;
t = T*fs +1;
Tpp = 10;
Tsp = 10;
mu = 10^-2.5;
%X = sin(2*pi*500*(0:Ts:T));
X = rand(1, T*fs +1);
PP = IMPULSE1([1,-.3,0.2],[1,0,0,0,0,0,0,0],0,Ts,Tpp);
SP = IMPULSE1([1, 1.5, -1],[1,0,0,0,0],0,Ts,Tpp);
PP = PP/max(PP);
SP = SP/max(SP);

Yd = zeros(t,1);                        %Recorded noise
Ys = zeros(t,1);                        %Control Signal
e_fxlms = zeros(t,1);                       %error
Cw1 =  zeros(1, N);
Xw1 =  zeros(1, N);
Cw_sum = zeros(length(SP), 1);

del = 0.9;

for n=1:t
    for i=1:min(n, length(PP))
        Yd(n) = Yd(n) + PP(i)*X(n-i +1);
    end
    Cy = 0;
    for i=1:min(n,N)
        Cy = Cy + Cw1(i)*X(n-i+1);
    end
    Cw_sum=[Cy; Cw_sum(1: end-1)];

    Ys(n) = sum(Cw_sum.*SP);
    e_fxlms(n)=Yd(n)+Ys(n);

    temp = 0;
    for i=1:min(n, N)
        temp = temp + SP(i)*X(n-i+1);
```

```
    end
    Xw1=[temp Xw1(1:end-1)];
    Cw1 = Cw1 - mu*e_fxlms(n)*Xw1;
    disp(n);
end

figure(1);
plot(e_fxlms);
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise residue')

figure(2);
plot(Yd)
hold on
plot(Yd-e_fxlms, 'r');
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise signal', 'Control signal')
hold off

figure(5);
plot(Yd)
hold on
plot(Yd-e_fxlms, 'r')
hold on
plot(e_fxlms);
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise signal', 'Control signal','errror residual')
hold off



function sys3 = IMPULSE1(num,den,Ti,Ts,Tf)
    sys = tf(num, den, Ts);
    sys3 = impulse(sys,Ti:Ts:Tf);
end
```


Noise signal
Control signal
Amplitude
Discrete time k
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
0
1000
2000
3000
4000
5000
6000


Noise residue
Amplitude
Discrete time k
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
1000
2000
3000
4000
5000
6000

Even Mirror Fourier Nonlinear Filters

These filters are characterized by two relevant properties:
i) they are able to arbitrarily well approximate any discrete-time, time-invariant, causal, infinite-memory, continuous, nonlinear system and
ii) they are always stable according to the bounded-input-bounded-output criterion. Even though recursive models can represent many systems with fewer coefficients than their finite-memory counterparts, it is still possible to further reduce their computational complexity.

The cosine and sine functions are the basis functions of order P=2k and P=2k+1 respectively where k is a positive integer or zero.

Order 0

$$1.$$

Order 1

$$\sin(\tfrac{1}{2}\pi\xi_0), \sin(\tfrac{1}{2}\pi\xi_1), \ldots, \sin(\tfrac{1}{2}\pi\xi_{N+M}).$$

Order 2

$$\begin{gathered}\cos(\pi\xi_0), \cos(\pi\xi_1), \ldots, \cos(\pi\xi_{N+M}),\\ \sin(\tfrac{1}{2}\pi\xi_0)\sin(\tfrac{1}{2}\pi\xi_1), \ldots, \sin(\tfrac{1}{2}\pi\xi_0)\sin(\tfrac{1}{2}\pi\xi_{N+M}),\\ \sin(\tfrac{1}{2}\pi\xi_1)\sin(\tfrac{1}{2}\pi\xi_2), \ldots, \sin(\tfrac{1}{2}\pi\xi_1)\sin(\tfrac{1}{2}\pi\xi_{N+M}),\\ \vdots\\ \sin(\tfrac{1}{2}\pi\xi_{N+M-1})\sin(\tfrac{1}{2}\pi\xi_{N+M}).\end{gathered}$$

Order 3

$$\begin{gathered}\sin(\tfrac{3}{2}\pi\xi_0), \sin(\tfrac{3}{2}\pi\xi_1), \ldots, \sin(\tfrac{3}{2}\pi\xi_{N+M}),\\ \cos(\pi\xi_0)\sin(\tfrac{1}{2}\pi\xi_1), \ldots, \cos(\pi\xi_0)\sin(\tfrac{1}{2}\pi\xi_{N+M}),\\ \cos(\pi\xi_1)\sin(\tfrac{1}{2}\pi\xi_2), \ldots, \cos(\pi\xi_1)\sin(\tfrac{1}{2}\pi\xi_{N+M}),\\ \vdots\\ \cos(\pi\xi_{N+M-1})\sin(\tfrac{1}{2}\pi\xi_{N+M}),\\ \sin(\tfrac{1}{2}\pi\xi_0)\cos(\pi\xi_1), \ldots, \sin(\tfrac{1}{2}\pi\xi_0)\cos(\pi\xi_{N+M}),\\ \sin(\tfrac{1}{2}\pi\xi_1)\cos(\pi\xi_2), \ldots, \sin(\tfrac{1}{2}\pi\xi_1)\cos(\pi\xi_{N+M}),\\ \vdots\\ \sin(\tfrac{1}{2}\pi\xi_{N+M-1})\cos(\pi\xi_{N+M}),\\ \sin(\tfrac{1}{2}\pi\xi_0)\sin(\tfrac{1}{2}\pi\xi_1)\sin(\tfrac{1}{2}\pi\xi_2), \ldots,\\ \sin(\tfrac{1}{2}\pi\xi_{N+M-2})\sin(\tfrac{1}{2}\pi\xi_{N+M-1})\sin(\tfrac{1}{2}\pi\xi_{N+M}).\end{gathered}$$

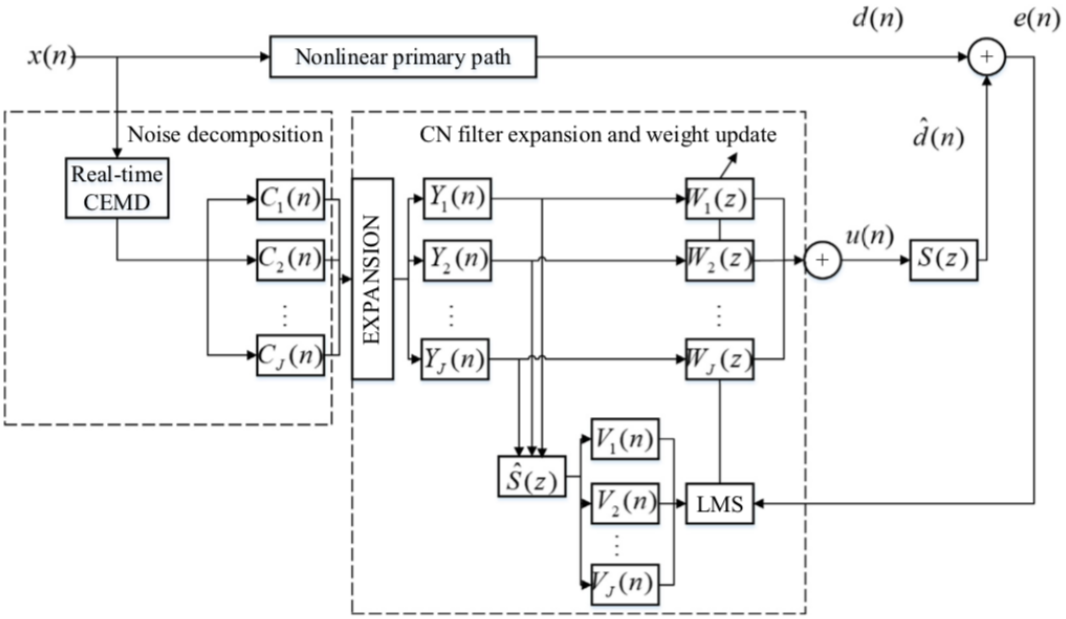
In general, the number of coefficients in an REMFN filter including all the kernels

$$N_{\mathrm{T}} = \binom{N+M+1+P}{N+M+1}.$$

of order 0 to P is equal to ,
The complexity of the filter hugely increases with the order of the filter and the number of samples and of the input and output signals, respectively.

Non-linear ANC system based on Chebyshev filter

A real-time correlated empirical mode decomposition (CEMD) is first designed to decompose the non-stationary noise into intrinsic mode functions (IMFs) and makes correlation analysis between the IMFs and primary noise. The second-order Chebyshev nonlinear (CN) filter, which has both the cross-terms and a linear term formed by the first order basis functions, is applied to expand the IMFs. Also, the controller weights of IMFs are adaptively updated by the FxLMS algorithm. Simulation results demonstrated that the proposed algorithm is robust for the non-stationary noise and nonlinear primary path.



The noise generated by rotating machinery under variable speed can be expressed as

$$x_s(n) = \sum_q B_q \cos(2\pi q f_s n + \beta_q)$$

The output signal is given by

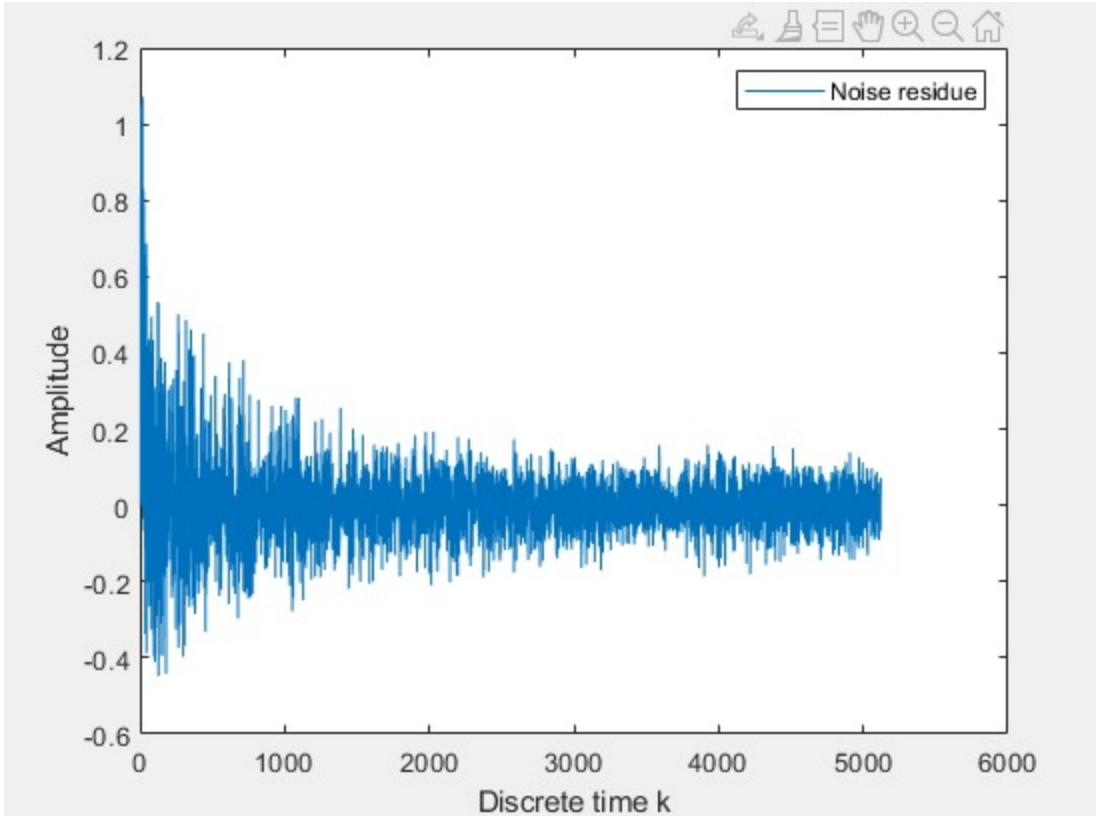
$$z(n) = \sum_{p=1}^{r} g_{k,p} \cos(2\pi \cdot p f_s \cdot n + \beta_{k,p}) + \gamma(n)$$

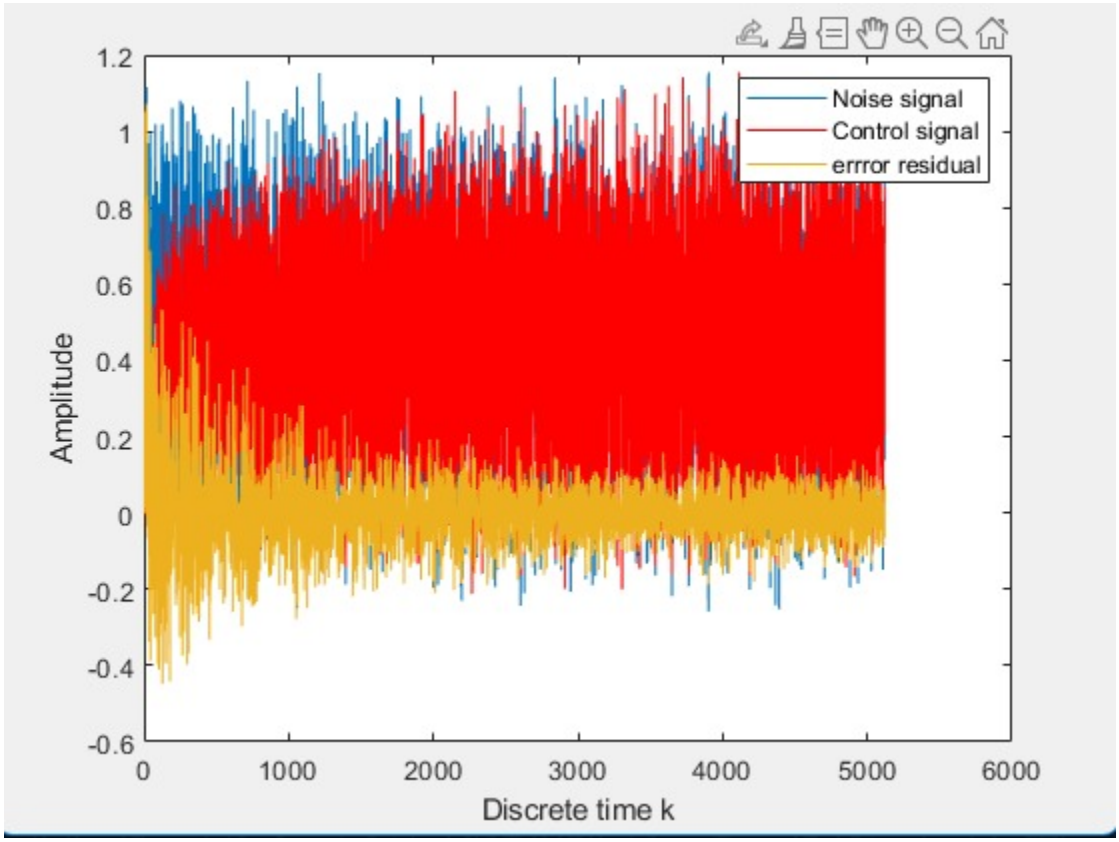
where $\gamma(n)$ is a constant term. Compared to the primary noise, $x_s(n)$, it is found that the frequency components of the output signal passed through the nonlinear primary path contain some higher order harmonics.

```
T = 1;
N1 = 30;
p = 6;
%fs = 51200;
fs = 5120;
Ts = 1/fs;
t = T*fs +1;

Tpp = 10;
Tsp = 10;

mu = 10^-2.7;
X = rand(t,1);
%x = downsample(UniversalMil.VarName2,10);

PP = IMPULSE1([1,-.3,0.2],[1,0,0,0,0,0,0,0],0,Ts,Tpp);
SP = IMPULSE1([1, 1.5, -1],[1,0,0,0,0],0,Ts,Tpp);

PP = PP/max(PP);
SP = SP/max(SP);

Yd = zeros(t,1);                        %Recorded noise
Ys = zeros(t,1);                        %Control Signal

Cw=zeros(N1, p);                            %weights
e_cxlms=zeros(t,1);                         %error

Xhx=zeros(N1, p);

Xw =  zeros(t, 1);

for n=1:t

    for i=1:min(n,length(PP))
        Yd(n) = Yd(n) + PP(i)*X(n-i +1);
    end
```

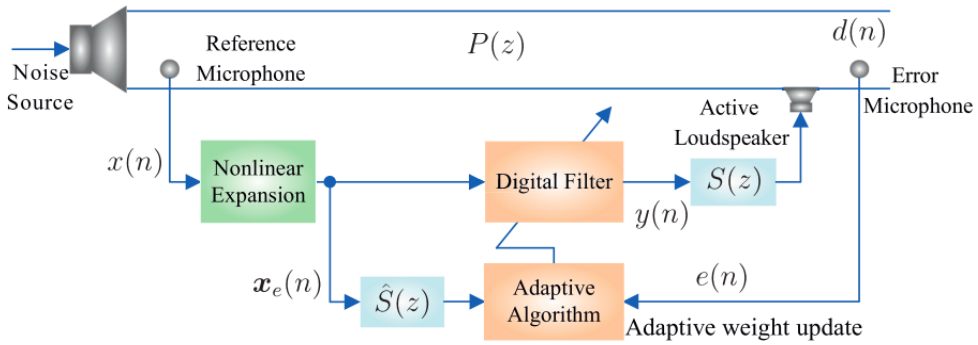
```
    for i=1:p
        for j=1:min(n,N1)
            Xw(n) = Xw(n) + Cw(j,i)*T_func(X(n-j+1), i);
        end
    end


    for i=1:min(n,N1+1)
        Ys(n) = Ys(n) + SP(i)*Xw(n-i+1);
    end

    e_cxlms(n)=Yd(n)+Ys(n);

    for i=1:p
        temp =0;
        for j=1:min(n,length(SP))
            temp = temp + SP(j)*T_func(X(n-j+1), i);
        end
        Xhx(:,i)= [temp; Xhx(1:end-1,i)];
    end

    for i=1:p
        Cw(:,i) = Cw(:,i) - mu*e_cxlms(n)*Xhx(:,i);
    end
end



figure(4);
plot(e_cxlms);
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise residue')

figure(2);
```

```
plot(Yd)
hold on
plot(Yd-e_cxlms, 'r');
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise signal', 'Control signal')

figure(3);
plot(Cw);
ylabel('Amplitude');
xlabel('Discrete time k');
legend('weights')

figure(5);
plot(Yd)
hold on
plot(Yd-e_cxlms, 'r')
hold on
plot(e_cxlms);
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise signal', 'Control signal','error residual')

function sys3 = IMPULSE1(num,den,Ti,Ts,Tf)

    sys = tf(num, den, Ts);
    sys3 = impulse(sys,Ti:Ts:Tf);

end
function res = T_func(x, n)
    if n<=1
        res = x^n;
    else
        res = 2*x*T_func(x, n-1) - T_func(x, n-2);
    end
end
```

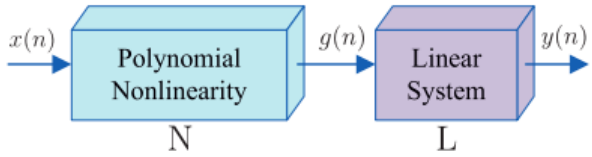

Noise residue
Amplitude
Discrete time k
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
1000
2000
3000
4000
5000
6000


Noise signal
Control signal
errror residual
Amplitude
Discrete time k
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
1000
2000
3000
4000
5000
6000

Non-linear ANC system based on Hammerstein filter



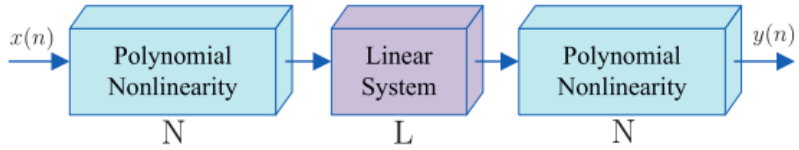
As the type of point-wise nonlinear expansions, the Hammerstein system model is regarded as the class of truncated Volterra filters. The basic Hammerstein system (N-L) model is comprised of a cascade connection of a memoryless nonlinearity in series with a linear system



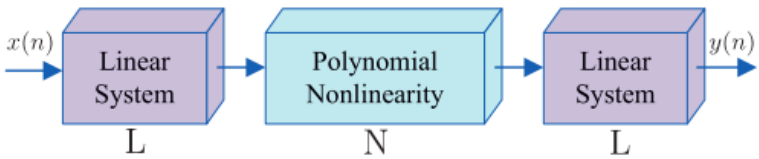
The input-output mapping of the memoryless nonlinearity can be expressed in the form of polynomial expansion

$g(n) = p_1x(n) + p_2x^2(n) + \ldots + p_Mx^M(n)$

where pj, j = 1, 2, . . ., M denotes the jth order coefficient. Combining the Wiener model (L-N), Hammerstein-Wiener model (NL-N) is closer to the actual nonlinear system than the Hammerstein model.



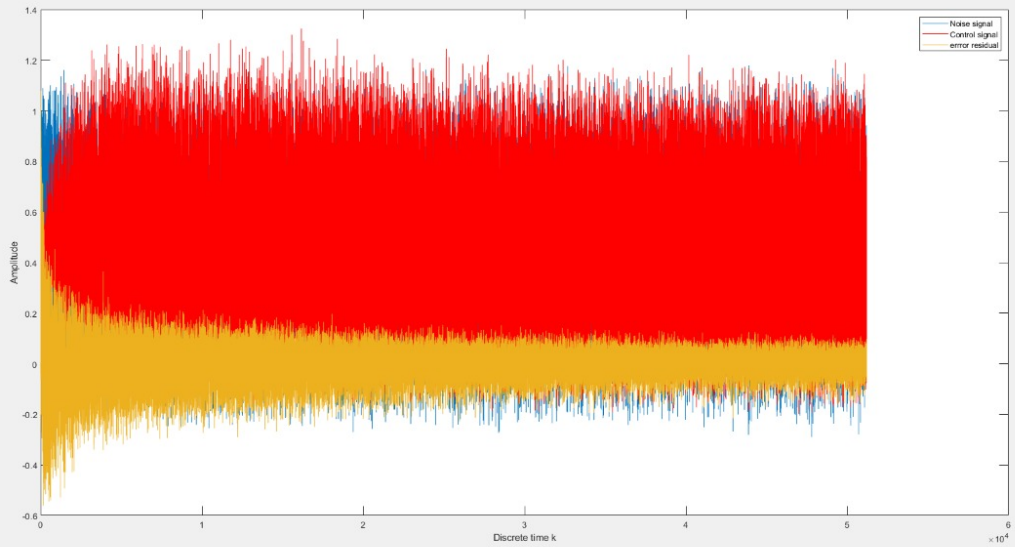
Although the Hammerstein-Wiener type nonlinear system has one more nonlinear link than Hammerstein and Wiener models, its essential feature is still a series of static nonlinear link and dynamic linear link. Changing the cascading order of the static nonlinear part and the dynamic linear part, the Wiener-Hammerstein model (L-N-L) can be obtained,

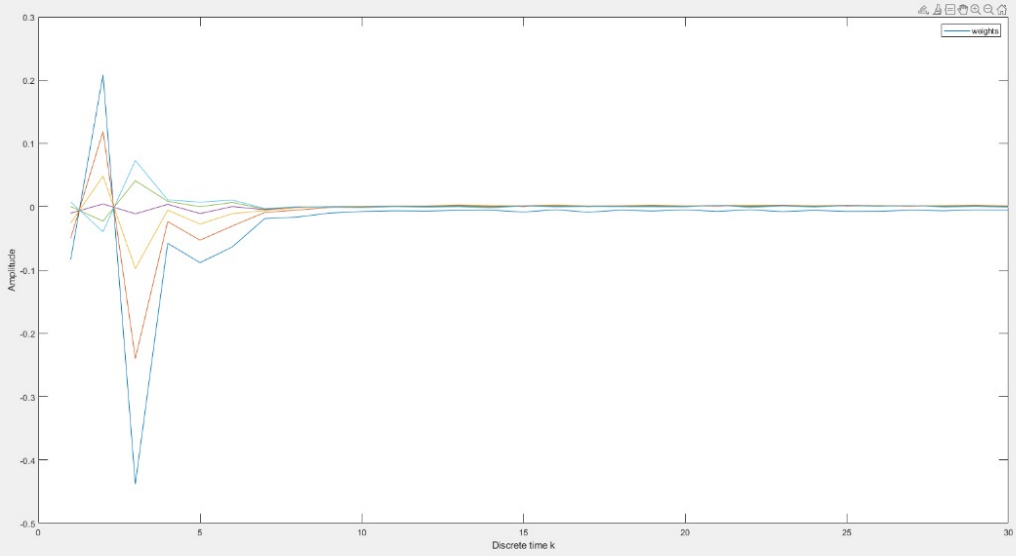


```
T = 10;
N1 = 30;
p = 6;
fs = 5120;
Ts = 1/fs;
t = T*fs+1;

Tpp = 10;
Tsp = 10;

mu = 10^-1;
%X = downsample(filling.VarName2,10);
%X = rand(t,1);
X = downsample(exp1.VarName2,10);
X = X/max(X);

PP = IMPULSE1([1,-.3,0.2],[1,0,0,0,0,0,0,0],0,Ts,Tpp);
SP = IMPULSE1([1, 1.5, -1],[1,0,0,0,0],0,Ts,Tpp);
PP = PP(1:40);
SP = SP(1:40);

PP = PP/max(PP);
SP = SP/max(SP);

Yd = zeros(t,1);                        %Recorded noise
Ys = zeros(t, 1);                        %Control Signal
e_hxlms = zeros(t,1);                      %error

Cw=zeros(N1, p);                           %weights
Xw = zeros(N1, p);

Cw_sum = zeros(length(SP), 1);

tic;
for n=1:t
```

```
    for i=1:min(n,length(PP))
        Yd(n) = Yd(n) + PP(i)*X(n-i +1);
    end

    Cy = 0;
    for i=1:p
        for j=1:min(n, N1)
               Cy = Cy + Cw(j,i)*(X(n-j+1)^i);
        end
    end


    Cw_sum=[Cy; Cw_sum(1: end-1)];
    Ys(n) = sum(Cw_sum.*SP);

    e_hxlms(n)=Yd(n)+Ys(n);

    for i=1:p
        temp =0;
        for j=1:min(n,length(SP))
            temp = temp + SP(j)*(X(n-j+1)^i);
        end
        Xw(:,i)= [temp; Xw(1:end-1,i)];
    end

    for i=1:p
        Cw(:,i) = Cw(:,i) - mu*e_hxlms(n)*Xw(:,i);
    end
end
toc;

figure(4);
plot(e_hxlms);
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise residue')
```

```
figure(2);
plot(Yd)
hold on
plot(Yd-e_hxlms, 'r');
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise signal', 'Control signal')

figure(3);
plot(Cw);
ylabel('Amplitude');
xlabel('Discrete time k');
legend('weights')

figure(5);
plot(Yd)
hold on
plot(Yd-e_hxlms, 'r')
hold on
plot(e_hxlms);
ylabel('Amplitude');
xlabel('Discrete time k');
legend('Noise signal', 'Control signal','errror residual')



function sys3 = IMPULSE1(num,den,Ti,Ts,Tf)

    sys = tf(num, den, Ts);

    sys3 = impulse(sys,Ti:Ts:Tf);

end
```


Noise signal
Control signal
error residual
Amplitude
Discrete time k
×10^4


weights
Amplitude
Discrete time k