Average Filtering: Theory, Design and Implementation

Javier Gonzalez-Barajas
Honeywell
57 PUBLICATIONS **60** CITATIONS
SEE PROFILE

Davis Montenegro
Electric Power Research Institute
56 PUBLICATIONS **527** CITATIONS
SEE PROFILE

Chapter

AVERAGE FILTERING: THEORY, DESIGN AND IMPLEMENTATION

***Davis Montenegro**[*] **and Javier Gonzalez**[†]*
Universidad Santo Tomás, Bogotá, Cundinamarca, Columbia

ABSTRACT

Digital filtering is a set of algorithms based on differential equations. The simplest algorithm within this set is the Finite Impulse Response (FIR) filter. This only requires the input samples to generate the filtered output, avoiding the feedback loops. FIR filters can be designed and implemented according to the need, and because of their simplicity have found application on many fields including Real-Time systems. A special implementation of a low pass algorithm is the averaging filter. It calculates the output sample using the average from a finite number of input samples. The averaging filter is used in situations where is necessary to smooth data that carrying high frequency distortion. The main aim of this chapter is the exposition of the theory, implementation and application of the average filtering. This chapter also presents the application of the average filteringin two study cases: electrophysiological signals and electrical power signals.

Keywords: average, digital signal processing, filtering, low order, real-time

[*] E-mail: davismontenegro@usantotomas.edu.co (Davis Montenegro).
[†] E-mail: javiergonzalezb@usantotomas.edu.co (Javier Gonzalez).

INTRODUCTION

Nowadays data acquisition using digital interfaces is the preferred technique for describing and analyzing physical phenomenon. Depending on the primary element (sensor) and the environmental conditions a wide range of high frequency noise can be part of the acquired waveform.

This situation demands a cleaning stage where the involved noise gets reduced after the acquisition process. With this process it is expected that further stages will receive clean data for performing analysis. As result, the accuracy of the obtained information for describing the phenomenon gets improved.

But not all applications look for the same goal. There are fields of application where it is required to eliminate signal components which are harmonics of the desired waveform. Once the undesired signal components are separated these can be used for performing control actions for eliminate them.

The cases mentioned above can be found in instruments, signal coditioning equipment, control equipment, among others. These applications have a common request: Fast algorithms for implementation with several hardware architectures for real-time operation.

These kind of problems can be addressed by using digital filtering. Digital filtering is a set of algorithms based on differential equations. The simplest algorithm within this set is the Finite Impulse Response (FIR) filter. This requires only the input samples to generate the filtered output, avoiding the feedback loops. FIR filters can be designed and implemented according to the need, and because of their simplicity have found application on many fields including Real-Time systems.

A special implementation of a low pass filters is the averaging filter. It calculates the output sample using the average value from a finite number of input samples. The averaging filter is used in situations where data carrying high frequency distortion need to be smoothed.

The aim of this chapter is the exposition of the theory, implementation and application of the average filtering. This chapter also presents the application of the average filtering in two study cases: electrophysiological signals and electrical power signals.

THE AVERAGING FILTER

Theoretical Background

A digital filter is a discrete system designed for processing data stored in arrays. Mathematically, a digital filter can be described using a differential equation as shown in (1).

$$\sum_{k=0}^{L-1} V_k y(n-k) = \sum_{k=0}^{L-1} W_k x\,(n-k) \tag{1}$$

An Alternative form is the Finite Impulse Response (FIR) filter. This uses only the actual and previous input for performing the filtering action. This form avoids the uses of previous outputs, thus reducing the computational burden when calculating the output y(n) as shown in (2).

$$y[n] = \sum_{k=0}^{L-1} W_k x(n-k) \tag{2}$$

A digital FIR structure with a particular function is the average filter. This Filter attenuates higher frequency components and smooths the signal. The average filter is shownin (3), where the variable L is de order of average filter.

$$y[n] = \frac{1}{L}\sum_{k=0}^{L-1} x(n-k) \tag{3}$$

These filters have the following characteristics [1]:

1. The transfer function only has one constant term. The other terms are always the coefficients of the shift registers where the inputs are stored. Additionally, all the poles are in the origin of the Z plane. This guarantees the stability of the filter because the impulse response counts with a finite number of terms.
2. The operations of the FIR filter involves only multiply the inputs by its coefficients for being accumulated. The implementation of this filter is very simple.
3. Given the format of the FIR filter its coefficients can be found to obtain the desired response. This filter can be configured as low-pass, band-pass and high pass.

There are several methods for obtaining the coefficients of this filter. These methods use the Fast Fourier Transform, time windows, frequency sampling among others [2]. Some of them and their practical application are shown as follows.

Design Algorithms

The frequency response of the average filter depends of the L value according to (2) and (3). Using the following code in MATLAB, it is possible estimate the frequency response.

```
l = [2 4 8 16 32 64];
N = length (l);
for i = 1:N
L = l (i);
B = (1/L)* ones (1, L);
[H, F] = freqz (B, 1, 200, Fs);
A (i,:) = abs (H);
end;
```

With this algorithm it is possible to calculate the response of the filter FIR in the frequency domain for different values of L. In Figure 1 the response for L = 4, 8 and 16 are shown.

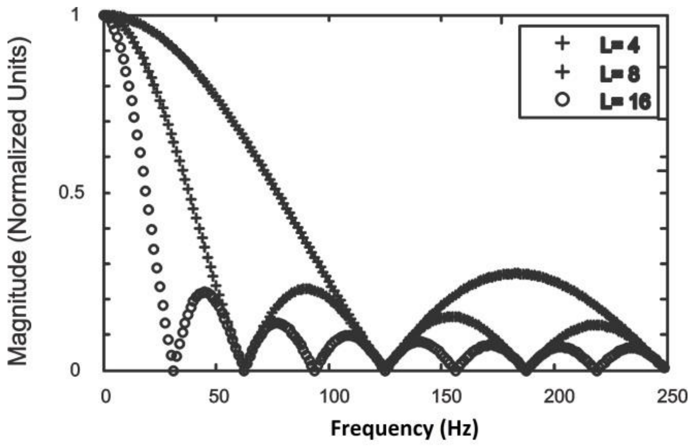

Figure 1. Response of the FIR filter for L = 4, 8 and 16.

In Figure 1 the operational characteristics of the filter can be appreciated such as the Gibbs oscillations, the ripple at the cut off frequency, among others. Then, for testing the performance of the filter with different values of L a waveform is built using a sample frequency of 500 Hz. This waveform is composed by 4 frequencies as shown in the following code:

```
Fs = 500;
Ts = 1/Fs;
n = 1:500;
t = (n-1)*Ts;
s = cos (2*pi*5*n*Ts);
r = cos (2*pi*13*n*Ts) + cos (2*pi*72*n*Ts) + cos (2*pi*130*n*Ts);
x = s + r;
```

As result, the signal x (n) is composed by two parts: the clean and desired waveform and the noise. The clean waveform is s(n) and the noise is represented by r(n). Both components are shown in Figure 2.

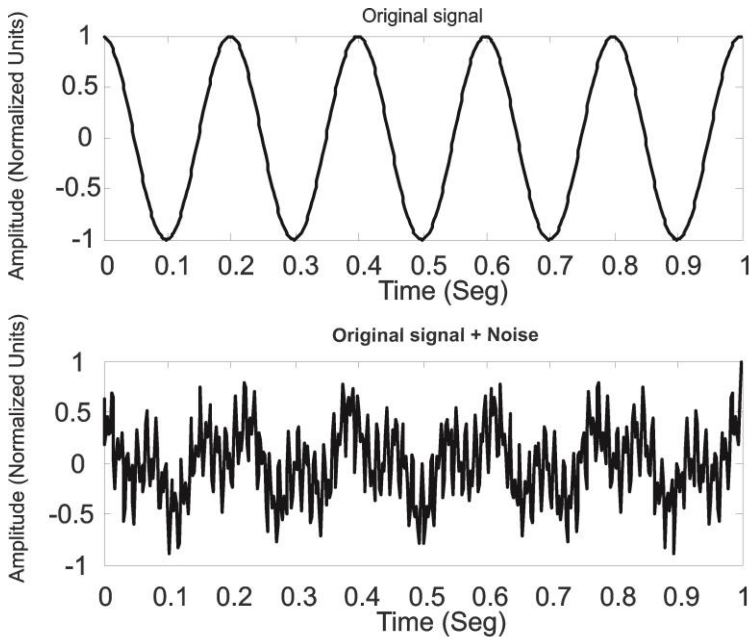

Figure 2. Clean waveform (Up) and Contaminated waveform (Bottom).

The contaminated waveform x(n) will be entered to the filter and the output signal y(n) will be plotted. For this example 4, 8, 16 and 32 coefficients (L) are proposed for designing 4 different filters. This procedure is made using the following code:

```
l= [4 8 16 32];
N = length (l);
for i = 1:N
L = l(i);
b = (1/L)* ones (1, L);
y (i,:) = filter(b, 1, x);
end;
```

The obtained outputs for each filter are shown on Figure 3. In this Figure is notable the low-pass effect of the filter and the smooth reached depending on the order of the filter. Additionally, it is evident that when the order of the filter gets increase the performance of the filter is better. However, the increment in the filter order means a higher computational burden. It is then when a balance must be found between computational burden and the filter response. Another consideration for the FIR filter is the phase shifting when increasing the order of the filter.

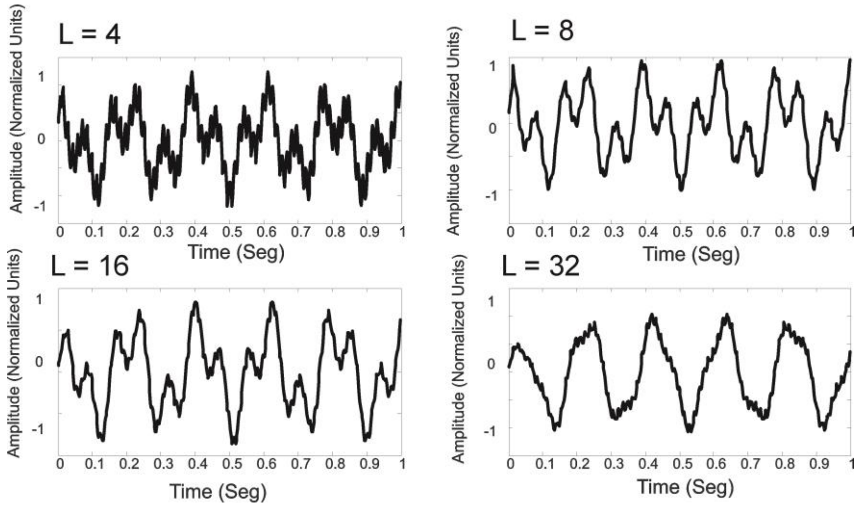

Figure 3. Signal y(n) obtained from different versions of the average filter.

For implementing this filter in different languages a simple coding is required. One technique for a simple implementation is using the concept of convolution [3]. According to its definition a convolution can be described as shown in (4).

$$y(n) = x(n) * h(n) \tag{4}$$

In this case h(n) is a discrete function and its coefficients corresponds to the value of L. This discrete function is described in (5) where it is an array of size L filled with ones divided by L.

$$h(n) = \left(\frac{1}{L}\right)[1_0 1_2 \ldots \ldots . . 1_{L-1}] \tag{5}$$

This implementation can be performed using C++ in a simple way. The following code can be used for implementing a simple average filter in a regular computer. The input samples are stored in the shift array x and the result of the convolution loop is placed in the variable c.

```
for (n = L + 1; n <= N; n ++)
{c = 0;
For (k = 1; k <= L, k ++)
c = c + x(n-k);
y (n) = c/L;}
```

But sometimes the time becomes a major concern. In these cases users would like to implement this kind of filters using hardware for covering a wider spectrum. The preferred technologies for this are FPGAs. Figure 4 shows the implementation of an average filter based on shift registers which can be described in hardware using VHDL.

In Figure 4, the shift registers Rn are used for storing the samples taken with the analog to digital converter (ADC) in time. The register B0 is used for storing the result of operating these samples according to (5) using bit right shifting. As result, the filtered signal will be found after operate all the coefficients of the filter corresponding to the shift register length.

This structure can be implemented using VHDL with the following code:

```
identity FILTER is
Port (X: in STD_LOGIC_VECTOR (7 downto 0);
C: in STD_LOGIC;
Y: out STD_LOGIC_VECTOR (7 downto 0);
end tarjeta;

architecture Behavioral of FILTER is
signal R0:std_logic_vector (7 downto 0);
signal R1:std_logic_vector (7 downto 0);
signal R2:std_logic_vector (7 downto 0);
signal R3:std_logic_vector (7 downto 0);
B0:std_logic_vector (7 downto 0);
B1:std_logic_vector (7 downto 0);

begin
process (C)
begin
if C = '0' and C'event then
X < = not A;
R0 <= not (A);
R1 <= R0;
R2 <= R1;
R3 <= R2;
end if;

end process;
B0 <= R0 + R1 + R2 + R3;
B1 (7) <= '0';
B1 (6) <='0';
B1 (5) <= B0 (7);
B1 (4) <= B0 (6);
B1 (3) <= B0 (5);
B1 (2) <= B0 (4);
B1 (1) <= B0 (3);
B1 (0) <= B0 (2);
Y <= B1;
end Behavioral;
```

This code can be used for reproducing the proposed filter within a FPGA. This device can then be used for several purposes.

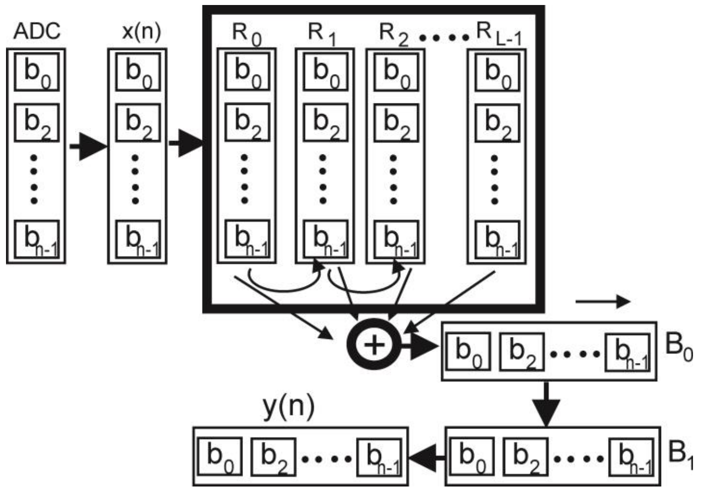

Figure 4. Average filter based on shift registers.

Applications in the Time and Frequency Domains

Many applications of the average filter can be found in the time and frequency domains. Sometimes, the input signals have baseline components for which modern technology is not immune. The problem with this kind of components is that they contaminate the low-frequency components of the studied signal [4]. In the Followings lines of code for MATLAB, it is possible simulate a discrete signal with a base line component.

```
fs = 500;
Ts = 0.001;
n = 1:1000;
t = (n-1)*Ts;
s1 = sin (2*pi*10*n*(1/fs));
s2 = sin (2*pi*15*n*(1/fs));
s3 = sin (2*pi*20*n*(1/fs));
s4 = sin (2*pi*25*n*(1/fs));
s5 = sin (2*pi*30*n*(1/fs));
x = s1 + s2 + s3 + s4 + s5;
v = 2*sin (2*pi*0.2*n*(1/fs));
s = x + v;
```

This code will generate a waveform based on sinusoidal signals. The baseline of a waveform can be described as a background that surrounds the important data of the composed signal [5]. This background represents a source of noise that can affect the accuracy of the information contained in the time window.

The waveform generated with the code presented above is shown in Figure 5. In this Figure it is possible to appreciate the concepts of baseline mentioned previously.

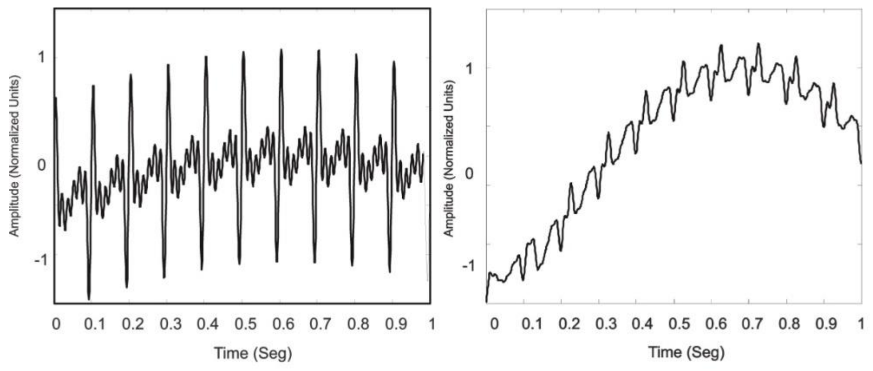

Figure 5. Waveform generated containing baseline components (left) and the baseline (right).

For isolating the baseline from the composed waveform an average filter can be implemented using 128 memory blocks (L). The code for implement this filter and extract the baseline as show in Figure 5 (right) is as follows:

```
L = 128;
b = (1/L)*ones (1, L);
y = filter (b, 1, s);
```

This simple implementation allows by generating a group of coefficients with the same value to characterize the useless data. In this particular case this modeled noise can be used for removing it from the original signal before perform extract the desired information. In Figure 5 if the signal at the right is algebraically subtracted from the signal at the left the base line components gets eliminated. The resulting waveform is shown in Figure 6.

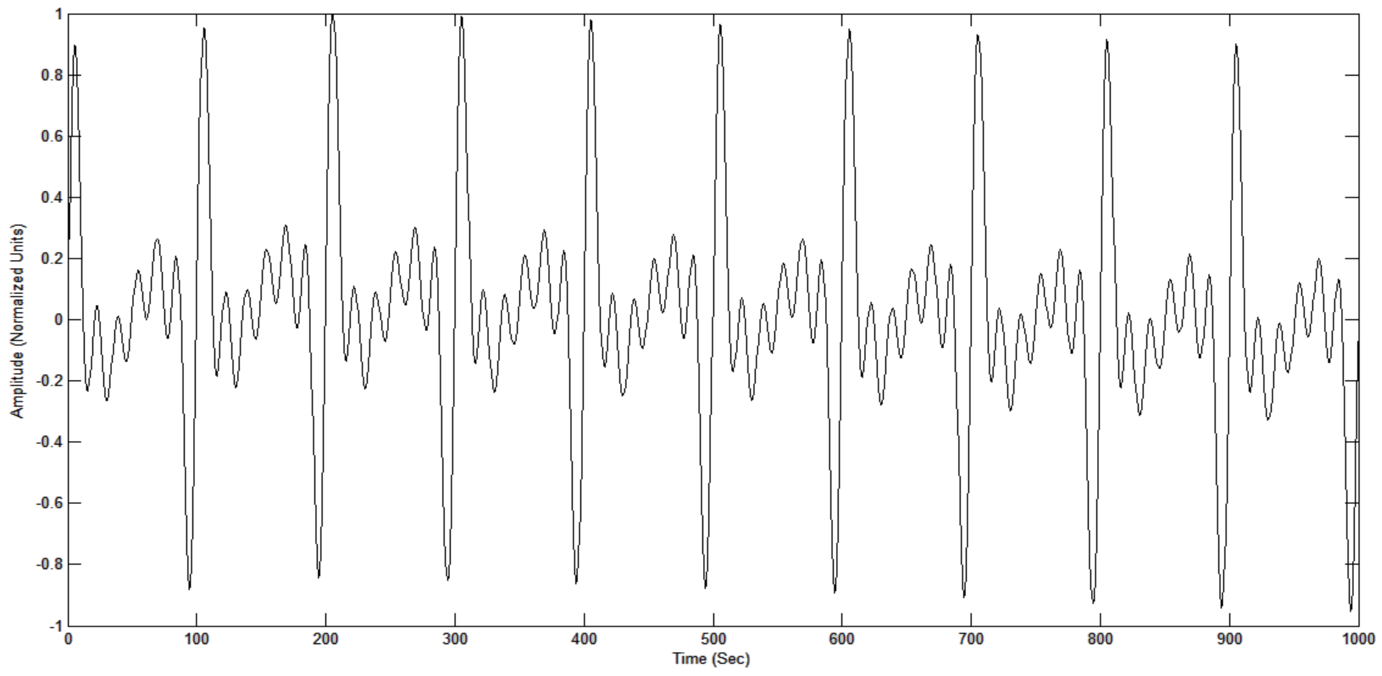

Figure 6. Waveform obtained after the baseline removal.

These procedures are common in several applications. Two examples of these applications are going to be presented as follows.

Study Case: Electrophysiological Signals

The electrocardiogram is an electrophysiological signal that describes the electrical behavior of the human heart. The electrocardiographic signal (ECG) requires several measurements from different parts of the patient's body (biomedical instrumentation). However, the presence of electrical equipment adds disturbances that affect the instrumentation system.

The ECG's amplitude is around 1mV and its bandwidth is between 0.5 and 100 Hz. Figure 9 shows an electrocardiographic signal taken from the signals data base Physionet [6]. The proposed signal and its components in the frequency domain are shown in Figure 7.

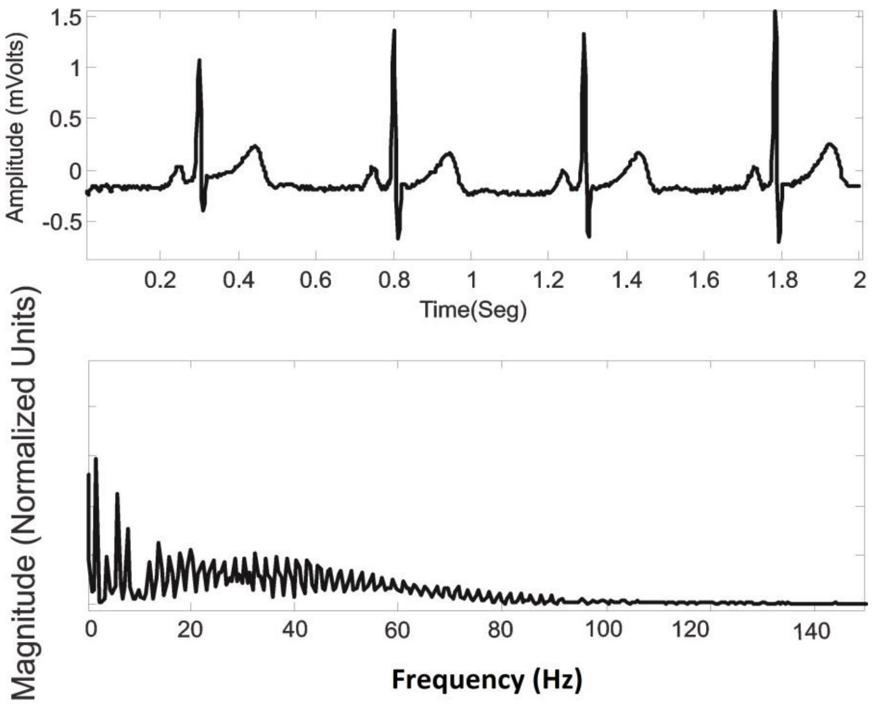

Figure 7. Proposed ECG signal and its spectrum.

This signal is going to be altered by adding noise corresponding to the noise generated by the proximity of power lines, which is common in ECG signals once they are acquired. The MATLAB code used for generate the new waveform is presented as follows:

```
load signal
fs = 500;
Ts = 1/fs;
N = 1000;
n = 1:N;
t= (n-1)/fs;
r = 0.5*sin (2*pi*60*n*Ts);
x1 = s + r;
```

The obtained waveform is shown in Figure 8. This new version of the original signal is closer to a real scenario where different factors, such as power lines, can add undesired data to the time window.

Then following the procedure presented above the average filter can be obtained by using a simple code in MATLAB. In this case a low order is selected for implement the filter (L = 8); the aim is to verify how with low orders the filter reach a good performance for this kind of tasks. The implemented code is as follows:

```
L = 8;
b = (1/L)*ones (1, L);
y = filter (b, 1, x1);
y = y/max (y);
```

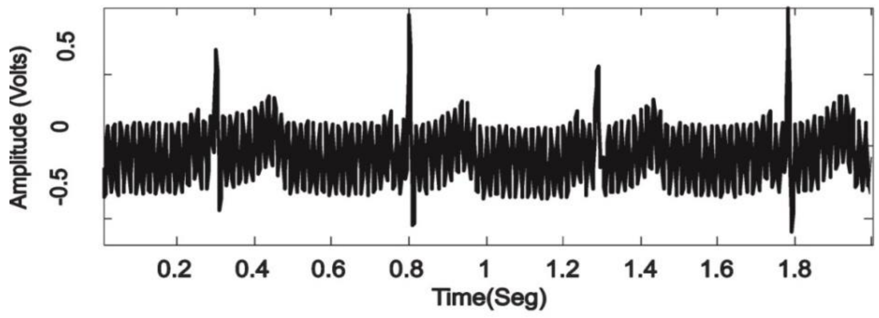

Figure 8. Distorted ECG signal by the proximity of power lines.

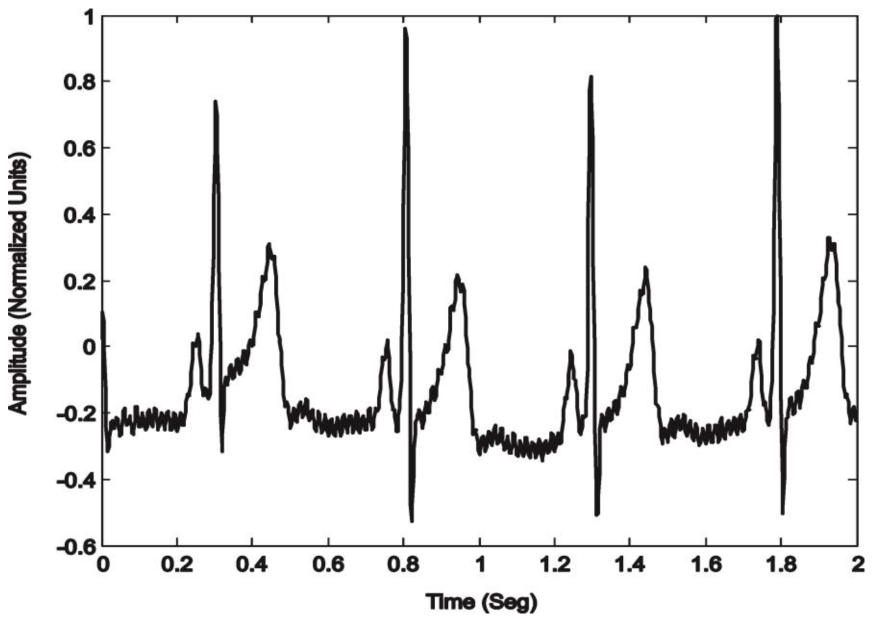

Figure 9. Filtered ECG signal using average filtering with L = 8.

As result the noise added initially becomes reduced significantly as shown in Figure 9. This demonstrates that the average filter results highly efficient for working as low-pass filter and eliminate the undesired components of the studied signal. Additionally the computational burden is not demanding for Real-time application.

Study Case: Active Filters for Power Conditioning

In power systems the connection of non-conventional loads such as power inverters, electronic power sources, Electric Vehicles, among others, has led a big workforce interested in power quality issues [7, 8]. The power quality disturbances are already identified and documented in international standards [9, 10]; these standards define the characteristics of the disturbances for monitoring and control.

Harmonic distortion is a very common disturbance and there is a big interest in reduce it in power systems [11, 12] for building the smart grid. Using active power filters the harmonics can be modeled and eliminated from the power system by injecting or draining energy. Figure 10 shows the general schematic of an Active filter for power conditioning connected in shunt.

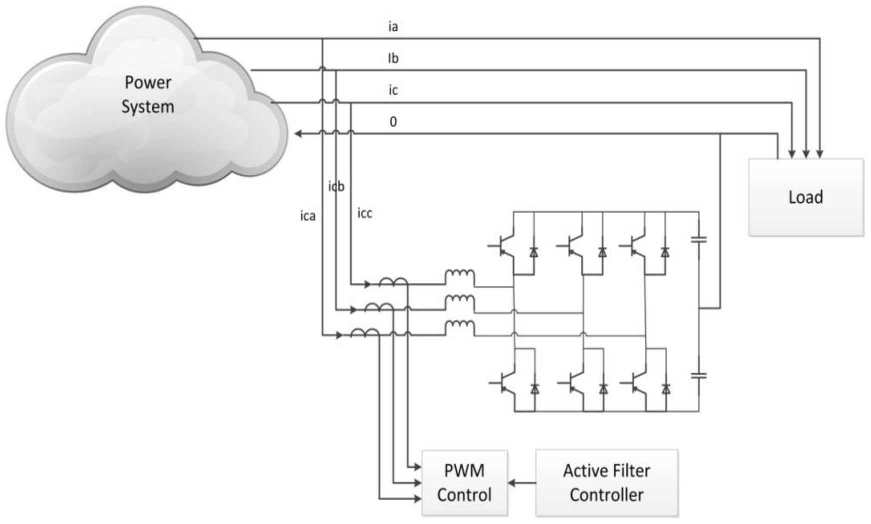

Figure 10. General schematic of active filters for power conditioning.

The control of the semiconductors for injecting the correcting signal is performed by the active filter controller. This structure was proposed by Akagi et al. [13] for applying the instantaneous power theory and eliminating the harmonics present in the power lines [14].

The interesting about this application is that works with only one sample for infer the operation of the semiconductor bridge. As result, energy can be dragged and stored in the capacitors or injected when it is necessary.

In this special application the use of average filtering results adequate. First the voltage and current signal samples are acquired and transformed from the *a*, *b*, *c* plane to the *α, β, 0* plane using Clark's transformation as shown in (6).

$$\begin{bmatrix} I_{ca} \\ I_{cb} \\ I_{cc} \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1/\sqrt{2} & 1 & 0 \\ 1/\sqrt{2} & -1/2 & \sqrt{3}/2 \\ 1/\sqrt{2} & -1/2 & -\sqrt{3}/2 \end{bmatrix} \begin{bmatrix} -i_0 \\ i_{c\alpha} \\ i_{c\beta} \end{bmatrix} \tag{6}$$

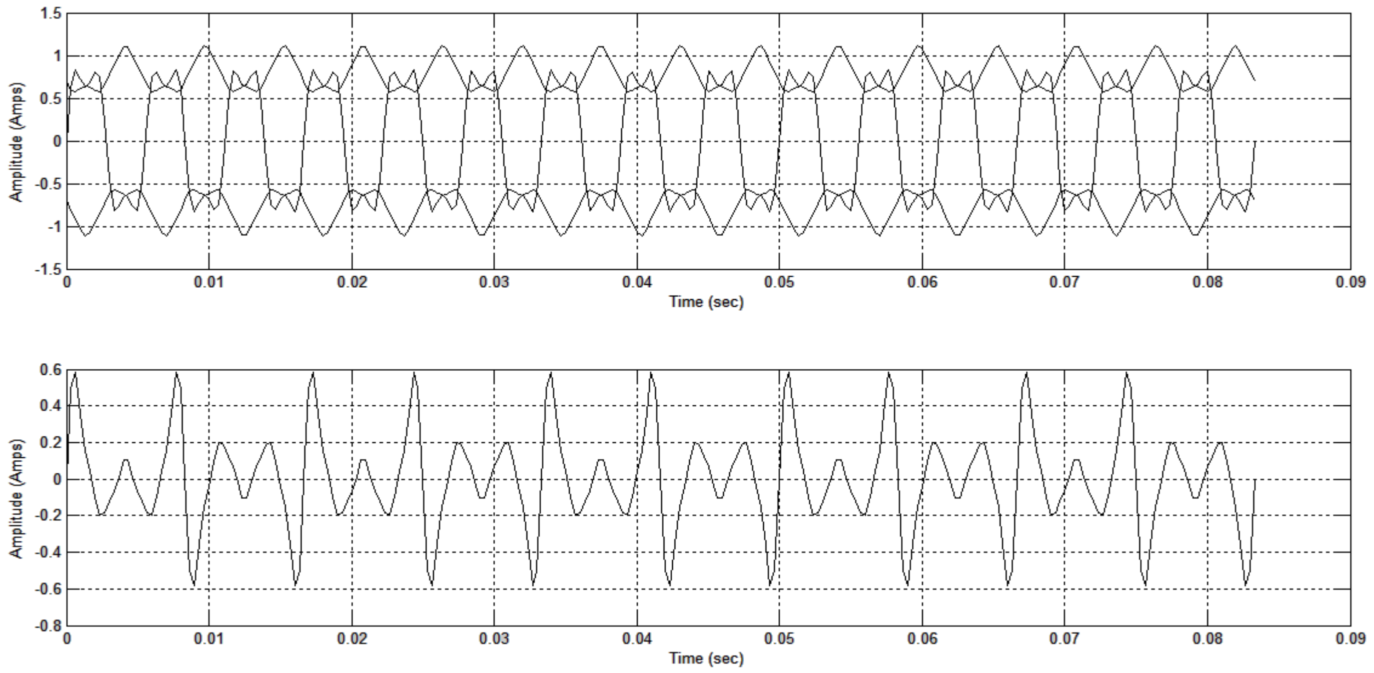

Figure 11. Distorted current waveforms (top) and filtered non active current (bottom).

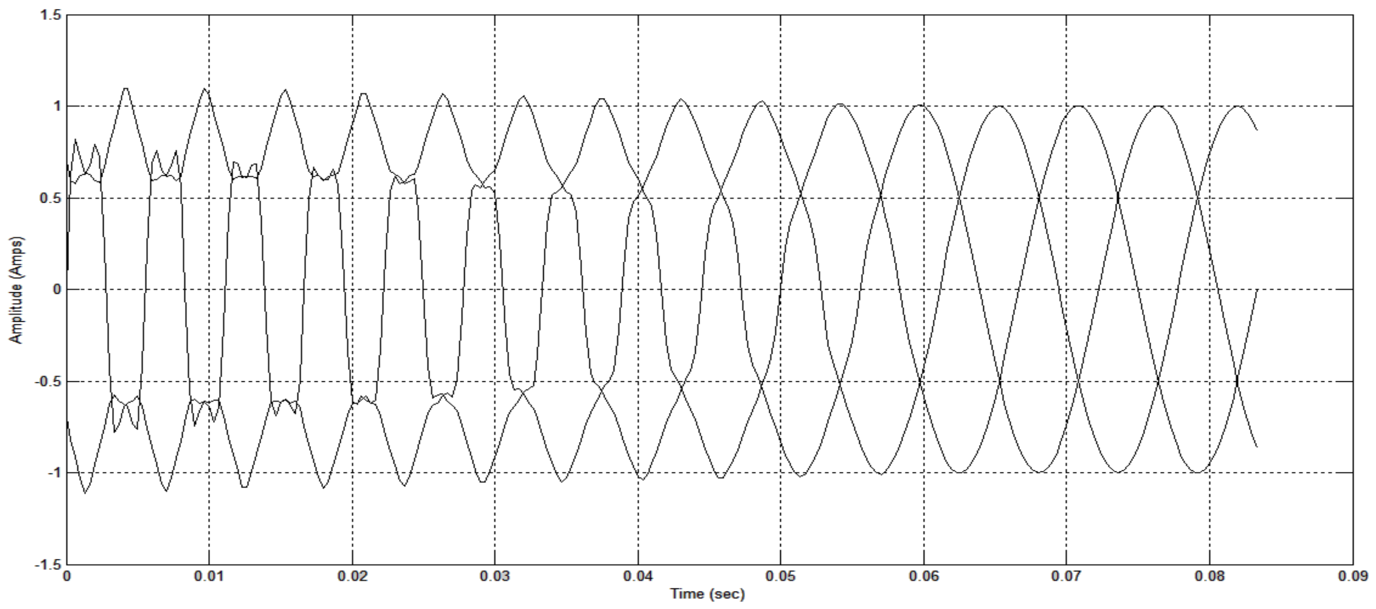

Figure 12. Obtained current signal (phase a) when the filter is operating.

Then, terms associated with 0 sequence (ground) can be discarded because they do not contribute to the non-active power [15]. With this reduces vector the reference frame can change from *α, β, 0* to the *p*, *q* plane. This new frame separates the active power from the non-active power, which represents the undesired components of the power signal (harmonics). This transformation is performed as follows:

$$\begin{bmatrix} i_{c\alpha} \\ i_{c\beta} \end{bmatrix} = \frac{1}{V_\alpha^2 + V_\beta^2} \begin{bmatrix} V_\alpha & -V_\beta \\ V_\beta & V_\alpha \end{bmatrix} \begin{bmatrix} -\tilde{p} + \Delta\bar{p} \\ -\bar{q} - \tilde{q} \end{bmatrix} \tag{7}$$

In (7) V_α and V_β were obtained using the same procedure presented in (6). These equations are invertible making possible to go from different reference frames for control purposes. After equation (7) is operated the composed power signal $-\tilde{p} + \Delta\bar{p}$ can be filtered using the current sample.

This consideration means that the action of the filter is performed after a number of iteration equal to the length of the filter's shift register. So if the filter has a length of 8 registers means that only after 8 iterations the filter will be initialized. This initialization will guarantee that all the coefficients of the filter are different to 0 and the filtering action will be performed.

After the filtering the new term *p* is transformed in inverse for determining the current value for injecting/drain from the power system. The injection/drain sequence of the semiconductor bridge is made by applying a hysteresis algorithm that delivers the triggering sequence for each phase.

The following MATLAB code generates the current waveforms of a three phase 6 pulse inverter; this device can be found in some Electric Vehicles. The X/R ratio of the system is quite low making the total harmonic distortion (THD) of the voltage signal lower than 1.

```
% Manitudes of odd harmonics and fundamental (Inverter IEEE 519)
Harm = [1 0.1378 0.30 0.12 0.089 0.056 0.044];
Fs = 3000; %Sample Frequency S/s
FFund = 60; %Fundamental Freq (60 Hz)
Per = 5; %Periods
Long = (Fs/FFund)*Per + 1; %length of the vectors
N = 0:Long-1; %Samples equivalent to "Per" periods
theta = 2*pi/3; %Phase angle
Ic = zeros (3, Long);
Tam = size (Harm);
% Generate the 3 phase current waveforms
```

```
for Fr = 1:Tam(2), %Tam(2),
Ic = Ic + [Harm (Fr)*sin((Fr*2-1)*2*pi*FFund*n/Fs);
Harm (Fr)*sin ((Fr*2-1)*(2*pi*FFund*n/Fs + theta));
Harm (Fr)*sin ((Fr*2-1)*(2*pi*FFund*n/Fs-theta))];
end;
t = n/Fs;
plot (t, Ic);
grid on;
xlabel ('Time (sec)');
ylabel('Amplitude (Amps)');
```

The generated waveforms are shown in Figure 11 (up). The magnitude of these signals is normalized and represents a demand big enough for impact negatively the power system. After performing the plane transformation of the current and voltage waveforms the average filter is applied. Then the non-active power is transformed into the harmonic current shown in Figure 11 (bottom). The resulting current waveform when the filter operates is shown in Figure 12.

CONCLUSION

This chapter has presented an introduction to the theory, design and implementation of average filters. The characteristics of these filters make of them easy for designing and implement. They are widely applied in several fields due their suitability for Real-Time execution. Two of these applications have been presented and also some codes for simulating them.

REFERENCES

[1] Tan, L. (2007). *Digital Signal Processing: Fundamentals and Applications*: Elsevier Science.

[2] Proakis, J. G., & Ingle, V. (1997). *Digital Signal Processing Using MATLAB V4.0*. Boston, United States of America: PWS Publishing Company.

[3] Proakis, J. G., & Manolakis, D. G. (1995). *Digital Signal Processing*. New Jersey, U.S.A.: Prentice- Hall, Inc.

[4] DeLuca, C. J., Gilmore, L. D., Kuznetsov, M., & Roy, S. H. (2010). Filtering the surface EMG signal: Movement artifact and baseline noise contamination. *Journal of Biomechanics, 43*, 1573-1579. doi: 10.1016/j. jbiomech.2010.01.027.

[5] Chouhan, V. S., & Mehta, S. S. (2007, 5-7 March 2007). *Total Removal of Baseline Drift from ECG Signal.* Paper presented at the International Conference on Computing: Theory and Applications, 2007. ICCTA '07.

[6] NIBIB. (2015). *Physionet: The research resource for complex physiological signals*. Retrieved January, 2015, from http://www. physionet.org/.

[7] Arritt, R. F., & Dugan, R. C. (2011). Distribution System Analysis and the Future Smart Grid. *IEEE Transactions on Industry Applications, 47*(6), 2343-2350. doi: 10.1109/TIA.2011.2168932.

[8] Zavoda, F. (2010). *Advanced Distribution Automation (ADA) Applications and Power Quality in Smart Grids.* Paper presented at the China International Conference on Electricity Distribution, Shanghai.

[9] IEEE Recommended Practice for Monitoring Electric Power Quality. (2009) *IEEE Std 1159-2009 (Revision of IEEE Std 1159-1995)* (pp. c1-81).

[10] IEEE Recommended Practices and Requirements for Harmonic Control in Electrical Power Systems. (1993). *IEEE Std 519-1992*, 0_1. doi: 10.1109/IEEESTD.1993.114370.

[11] Aredes, M., & Watanabe, E. H. (1995). New control algorithms for series and shunt three-phase four-wire active power filters. *IEEE Transactions on Power Delivery, 10*(3), 1649-1656. doi:10.1109/61. 400952.

[12] Bollen, M. H. J., Gu, I. Y. H., Santoso, S., McGranaghan, M. F., Crossley, P. A., Ribeiro, M. V., & Ribeiro, P. F. (2009). Bridging the gap between signal and power. *Signal Processing Magazine, IEEE, 26*(4), 12-31. doi: 10.1109/MSP.2009.932706.

[13] Akagi, H., Kanazawa, Y., & Nabae, A. (1984). Instantaneous Reactive Power Compensators Comprising Switching Devices without Energy Storage Components. *IEEE Transactions on Industry Applications, IA-20*(3), 625-630. doi: 10.1109/TIA.1984.4504460.

[14] Akagi, H., Watanabe, E. H., & Aredes, M. (2007). *Instantaneous Power Theory and Applications to Power Conditioning*: Wiley.

[15] Watanabe, E. H., Stephan, R. M., & Aredes, M. (1993). New concepts of instantaneous active and reactive powers in electrical systems with generic loads. *IEEE Transactions on Power Delivery, 8*(2), 697-703. doi: 10.1109/61.216877.

P. K.