Journal of Computer Science

Original Research Paper

Software-based Control Algorithm for Providing Convergence Region with an Increasing Polynomial under Windowed FIR Filters

[1]Sergio Bimbi Junior, [2,3,4]Vitor Chaves De Oliveira, [1,5]Agenor De Toledo Fleury and [6]Ronaldo Ruas

[1]Technological Research Institute ("IPT"), Building 31 - Av. Prof. Almeida Prado, 532 - Butantã, São Paulo - SP, 05508-901, Brazil
[2]Pontifical Catholic University of Campinas – PUCC, R. Prof. Dr. Euryclides de Jesus Zerbini, 1516 – Pq. Rural F. Santa Cândida - Campinas - SP, 13087-571, Brazil
[3]Federal Institute of Education, Science and Technology of São Paulo – IFSP, R. Rio Branco, 1780 - Vila Teixeira, Salto - SP, 13320-270, Brazil
[4]Mackenzie Presbyterian University – UPM, R. da Consolação, 930 - Consolação, São Paulo - SP, 01302-907, Brazil
[5]Mechanical, Mechatronics and Marine Engineering of USP's Polytechnic School, Av. Professor Mello Moraes, 2231-Butantã, São Paulo - SP, 05508-030, Brazil
[6]FATEC Pref. Hirant Sanazar (FATEC Osasco), R. Pedro Rissato, 30 - Vila dos Remédios, Osasco - SP, 06296-220, Brazil

Article history
Received: 25-03-2018
Revised: 17-06-2018
Accepted: 10-07-2018

Corresponding Author:
Vitor Chaves De Oliveira
Pontifical Catholic University of Campinas – PUCC, R. Prof. Dr. Euryclides de Jesus Zerbini, 1516 – Pq. Rural F. Santa Cândida - Campinas - SP, 13087-571, Brazil
Email: vitor_chaves_oliveira@yahoo.com.br

Abstract: This work developed a control algorithm for dynamical systems to high performance manner, in a way to produce an optimum output applying a modification in digital filters, capable of increasing the convergence zone within the desired cut -off frequency. Such task focused on consistently improving the challenge present in digital filters, which are time-invariant linear systems that are able to modify connected input signal characteristics, where only a specific signal component of the frequency is able to reach the filter output. In dynamic systems, digital filters are applied to optimize system measurements with respect to performance and stability. This work, through C and C# coding demonstrates a modification to windowed low-pass filters within the π sample space. The π sample space is divided, and small parts of the equation are added into a polynomial of degree *n*; this technique removes unwanted frequency components in small angular frequency windows, causing acceleration of the signal with respect to the windowed low-pass filter. This feature is very important in dynamic measurement systems because the system achieves an increased number in values closer to the target to obtain an average that indicates values with a high level of accuracy and repeatability. This work also conducted a series of experiments to verify and validate the control algorithm, this software was fully implemented in embedded industrial systems, which support low-level C coding language that is the industry standard.

Keywords: Control Algorithm, Software Development, Digital Windowed Filter, Digital Signal Processing, Low-level Programming

Introduction

The unceasing challenge to improve digital signals accuracy and performance requires a software and processing time capabilities that are met only by costly hardware employment. Therefore, it brings several practical financial business limitations to most of industries from telecommunications, to cars, to machines present manufacturing in industrial sites and any other system that demands digital signal processing within better boundaries and in less time. This work shows a software-based control algorithm solution that can be implemented requiring low- cost hardware and that can be embedded to almost every device with a digital filter that processes signals aimed to increase precision, reduce processing time and increase productivity with a 50% improvement in the cut-off frequency convergence in FIR filter type low pass. This algorithm creates, to the best of our knowledge, a comprehensive new solution that can be applied to solve, in a satisfactory manner, the defies that lie ahead the digital signal processing task, i.e. filters. The work was developed using C low-level programming language and C# to computationally achieve the objective. It is alto interesting to point out that using this programmable technique an increase of 48.8% in standard deviation gain.

Filters are time-invariant linear systems that are able to modify input signals, where only the frequency component of a specific signal is able to reach the filter output. The consideration of analog signals *x* (*t*) and *y* (*t*) and a filter designed as a response function for impulse *h* (*t*) is shown in (1):

$$y(1) = h(t) * x(t) \tag{1}$$

In the frequency domain, this can be equated as shown in (2):

$$Y(j\omega) = H(j\omega) * X(j\omega) \tag{2}$$

Considering that the digital filter implemented is a Digital Signal Processor (DSP) and that the purpose is to process an analog signal *x* (*t*) sampled by an analog into a Digital Converter (ADC) and an output *y* (*t*) generated by a digital into an analog converter (DAC), then it is possible to characterize a digital filtering (Bimbi Jr. *et al*., 2016a; 2016b) system as shown in Fig. 1 (Butterweck, 1975).

If the signals being processed are digital, then the diagram can be summarized as shown in Fig. 2 (Butterweck, 1975).

To implement a digital and Linear Invariant Time (LTI) filter, a DSP is required; the computational algorithms of the DSP will be described. The algorithms can be represented as block diagrams using basic structures, such as unit delays, gains and feedbacks. The structures of the block diagrams are similar to the differential equation of the filter, also known as the canonical structure (Adams and Willson Jr., 1983):

Signals and Systems

Signals are present in various electronic systems for measurement and analysis. A signal can generally be specified as a function that carries information. For analysis and decisions regarding the carried information, we apply the Digital Signal Processing (DSP) technique. In general, signals are generated by physical processes and require a transducer that translates these physical measurements into an electrical signal in order to be analyzed for digital processing (Adams and Willson Jr., 1984). A fundamental aspect of this implementation is the quantization and sampling of the signal; this aspect is directly connected to the quantity of samples. To proceed with the analysis, a continuous-time signal (or a discrete-time signal) is converted into a sample sequence. After the digital signal processing, this sequence may be converted back into a continuous-time signal (Ahmed *et al*., 1974). Digital signal processing can be described as when an analog signal is acquired and digitalized through sampling and when a filter is applied to obtain only the part that is desired during processing (Akansu and Medley, 1999). Signals can be characterized into categories, which are dependent on time, frequency and amplitude values. Continuous-time or analog signals are defined over any value of time that can be assumed in a continuous range. Thus, these signals can be represented by a function of continuous variables. Discrete-time signals are defined only at specific time values (Antoniou, 1982). They are represented mathematically by a sequence *x* of complex real numbers, where the n^{th} number of this sequence is denoted $x[n]$, as shown in (3):

$$X = \{x[n]\}, -\infty < n < \infty \tag{3}$$

where, *n* is an integer and samplings are acquired through a periodic and time-invariant process of sampling the analog signal. Thus, the numerical value of the n^{th} number of this sampled sequence will equal the analog value *xa* (*t*) in discrete time *nT* as (4) demonstrates:

$$[n] = xa(nT), -\infty < n < \infty \tag{4}$$

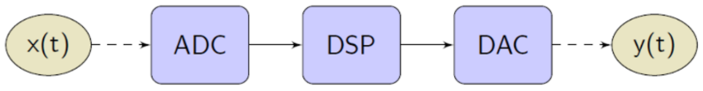


Fig. 1: Architecture of a digital filter

Fig. 2: Summary of the architecture of a digital filter

Continuous- or discrete-time signals can assume any amplitude value within finite or infinite space (i.e., they are continuous with respect to the amplitude values). Digital signals are discrete with respect to both time and amplitude. Thus, it can assume values within a finite set of possible values (Avenhaus, 1972).

Signals can be deterministic or random; any signal that can be specified by a mathematical equation, data table or well- defined rule is called a deterministic signal. However, in practical applications, signals cannot be accurately represented by mathematical equations or complex descriptions. This behavior indicates signals that are unpredictable in time, which are called random signals (Belevitch, 1968).

Linear Time-Invariant (LTI) Systems as Frequency Selector Filters

The term filter is used to describe a device or function that discriminates the property or attribute that will be accepted as a function of an input object (i.e., what passes through it) (Boyd and Vandenberghe, 2004). An LTI system fits this description by filtering or discriminating frequency components as its input. The method of filtering is defined by the frequency response H (ω), which is entirely dependent on the parameterization of the system. Accordingly, it is possible to design filters that are capable of performing the selection of frequency components in some bands and attenuating frequency component signals in other bands (Antoniou, 1993). An LTI system modifies the spectrum of the input signal $X(w)$ according to the frequency response $H(w)$, which leads to an output signal with a spectrum $Y(w) = H(w)X(w)$. In general, H (w) acts as a weight function for different frequency components of the input signal. Thus, an LTI system can be viewed as a filter, although it does not entirely block any frequency component of the input signal. Accordingly, one can conclude that a filter is a process used in digital signal processing that aims to perform a frequency selection and can be applied to noise removals, spectral analysis, and other applications. Filters are classified according to their features within the frequency domain; examples of filters are high-pass, low-pass, band- pass and band-stop (Antoniou, 2006).

Sequences through Fourier Transform

Similar to continuous-time signals, discrete-time signals can be demonstrated in different ways. One possible and applicable technique is the transformation of the time domain signal into a frequency domain through the Fourier transform technique, as (5) demonstrates (Antoniou and Rezk, 1977):

$$x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X\left(e^{j\omega}\right)e^{j\omega}d\omega \tag{5}$$

where, $X(e^{j\omega})$ is described by (6):

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega} \tag{6}$$

The Fourier transform is a complex function of ω. Its frequency response can be expressed by X (ω), as shown by (7) or (8) in polar form:

$$X(e^{j\omega}) = X_R(e^{j\omega}) + jX_i(e^{j\omega}) \tag{7}$$

$$X(e^{j\omega}) = \left|X(e^{j\omega})\right|e^{j<e^{j\omega}} \tag{8}$$

Equation 11 describes the magnitude and phase of the Fourier transform. There are cases where we can prove that the Fourier transform is not able to converge; in these cases, the Z-transform can be applied. Considering that (5) and (6) are inverse, one should consider (9) (Benvenuto *et al.*, 1984):

$$\frac{1}{2\pi}\int_{-\pi}^{\pi}\left(\sum_{m=-\infty}^{\infty} x[m]e^{-j\omega m}\right)e^{j\omega m}d\omega = \overline{x[n]} \tag{9}$$

By executing the inversion of the integration operation with the summation operation, we obtain (10):

$$\overline{x[n]} = \sum_{m=-\infty}^{\infty} x[m]\left(\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{j\omega(n-m)}\right)d\omega \tag{10}$$

By applying the integral within the parentheses, (11) is obtained:

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{j\omega(n-m)}d\omega = \frac{\sin(\pi(n-m))}{(\pi(n-m))}\begin{cases}1, m=n\\ 0, m\neq n\\ =\delta[m-m]\end{cases} \tag{11}$$

Accordingly, (12) can be determined:

$$\overline{x[n]} = \sum_{m=-\infty}^{\infty} x[m](\delta[n-m]) = x[n] \tag{12}$$

Z-Transform

The Z-transform is a mathematical tool that applies to signal and system analyses. The Z-transform characterizes the discrete form of the Laplace transform. By applying the Fourier transform, a sequence can be characterized by (13) (Bomar, 1989):

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n](e^{-j\omega}) \tag{13}$$

Defining z as $e^{j\omega}$, one can characterize the Z-transform as shown in (14):

$$X(z) = \sum_{n=-\infty}^{\infty} x[n](z^{-n}) \tag{14}$$

Equation 14 is also called a bilateral Z-transform; the unilateral Z-transform is described by (15), where the boundaries of the summation are limited by *n* starting at zero, thus determining its unilateral nature:

$$X(z) = \sum_{n=0}^{\infty} x[n](z^{-n}) \tag{15}$$

A relationship between the Fourier transform and the Z-transform can be observed. If we determine that *z* is a complex variable, it can be described by $e^{j\omega} = \cos(w) + jsen(w)$; thus, the Z-transform becomes the Fourier transform. For $z = r^{ej\omega}$, the circumference of the imaginary plane (i.e., the Z-plane) is determined, which is a circle with a unit radius equal to 1. Accordingly, the Z-transform when calculated for a circle with a unit radius is equal to the Fourier transform (Bomar and Joseph, 1987).

Sampling Theorem

In general, discrete-time signals can be conceived in different ways. The most common and applicable way is to carry out a representation of the continuous-time signals.

This occurs when digital signal processing is carried out through a sequence obtained within a sample space. Within this sample space, readings are performed (i.e., the sample acquisition); the quantity of samples, based on the frequency that the signal resonates at, determines the precision and repeatability of its reconstitution. Accordingly, in a sample space, we obtain a sequence *x*[*n*] that is obtained from continuous-time signal (Bomar *et al*., 1997).

To carry out such a process, it is necessary to implement an ADC converter. An ADC converter is a very important element: it samples and quantizes the values of a continuous- time signal (TAGHOUTI, 2010). Accordingly, it is necessary for evaluating parameters, such as the quantization of the output, linearity, resolution and sampling frequency. During ADC conversion, it is necessary to choose a discrete number of samples from the continuous signal to be processed. The main problem in this field is the choice of the number of samples; a reduced number of samples different from the demand may result in a distorted representation of the signal (Boyd and Vandenberghe, 2004). Accordingly, criterion when choosing the reconstitution frequency can be determined by using the Nyquist-Shannon theorem, where the sampling frequency is at least twice the fundamental frequency of the discrete-time signal. In practical terms, due to the sampling time (Senthilkum and Natarajan, 2008), a frequency that extrapolates the fundamental frequency of the signal is often applied, thus generating points between smaller intervals and ensuring a reliable reconstitution (Burrus and Parks, 1970).

Digital Filters

A discrete-time system is mathematically defined as the execution of a transformation that maps an input sequence *x*[*n*] into an output sequence *y*[*n*] (PANICH, 2010). During digital signal processing, this procedure is generally intended for the manipulation of a signal; as an example, one can apply a system for checking weight in a dynamic form, where the variable that is important to the system is the weight, which is a DC component represented by a load cell. However, such acquisition requires the removal of unwanted components, such as vibrations generated by motors, rollers, bearings and conveyors, as well as electrical and electromagnetic noise that is present in the process (Butterweck, 1975).

FIR Filters

The architecture of an FIR filter (finite impulse response filter) (AMIN, 2011) has a general and regular behavior, since its coefficients are defined. These coefficients are defined by the choice of behavior required for the process or application. In a measurement system, as mentioned above, the DC component is interesting for the weight reconstitution: it allows low-pass type filters to be applicable, where the pass-band is limited as a low-order frequency that tends towards the DC level (with a frequency equal to 0) (Butterweck *et al*., 1984). The architecture of an FIR filter can be seen in Fig. 3.

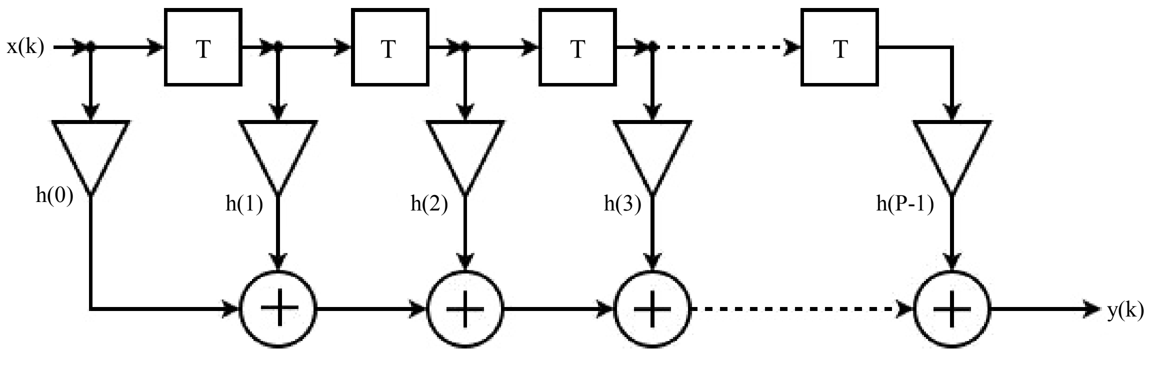


Fig. 3: Architecture of an FIR filter

FIR filters can implement different functions by simply changing their coefficients and performing as low-pass, high-pass, band-pass or band-stop filters. For practical applications, filters do not have ideal characteristics, and their responses may contain ripples in a certain transition band. Accordingly, it can be said that even with the implementation of filters, an ideal response is not obtained when analyzing their spectra.

Because the system attempts to measure only DC components that represent weights with frequencies tending towards zero, an interesting application is designing windows; in these windows, we can select frequencies appropriate for the desired behavior, which truncates the impulse response and obtains a causal and non-linear phase FIR filter. A windowed filter is characterized by a linear phase during its pass-band and a zero response during the stopband. An ideal low-pass filter with a $w_c < \pi$ bandwidth is described by (16):

$$H_d(e^{j\omega})\begin{cases} 1e^{-j\propto\omega}, w| \le w_c \\ 0, w_{c<|w|\le\pi} \end{cases} \tag{16}$$

where, w_c is the cutoff frequency and $\propto$ is the sample delay. The filter response to infinite duration is given by (17) and (18):

$$h_d[n] = \delta^{-1}[H_d(e^{j\omega})] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})e^{j\omega}dw \tag{17}$$

$$h_d[n] = \frac{1}{2\pi}\int_{-wc}^{wc} 1(e^{j\omega})e^{j\omega}dw = \frac{\sin[wc(n-\propto)]}{\pi(n-\propto)} \tag{18}$$

To get an FIR filter from $h_d[n]$, we need to truncate the values of $h_d[n]$ on both sides. Thus, we determine a linear-phase causal FIR filter in $h[n]$ of length M, shown by (19):

$$h[n] = \begin{cases} h_d[n], 0 \le n \le M-1 \\ 0, \ otherwise \end{cases} \tag{19}$$

With this, we obtain $\propto = (M - 1)/2$. This operation is called windowing, where a symmetric function is obtained with respect to the ranges of $\propto$ over $0 \le n \le M - 1$, where zero is outside of the range.

Convergence Region with an Increasing Polynomial in Windowed FIR Filters

The distortion reduction within a window carried out by a filter is an important factor regarding the rejection of undesirable points in present lobes, which are not acceptable beyond the main lobe; this occurs because it represents frequencies that are not required for that filter. As the number of M coefficients grows, the width of each side lobe decreases, but the area over the lobes remains constant (Cabezas and Diniz, 1990). This causes oscillations and ripples to experience peaks near the band edges. This fact is called the "Gibbs phenomenon." To carry out the proposal of reducing distortion within a window from the many applicable windows, it is necessary to adapt the window equation. In this study, reference windows are calculated by the Hamming window (20), the Hann window (21) and the Blackman window (22), where n is equal to the current coefficient of the filter and M is the total number of coefficients:

$$w[n] = 0{,}54 - 0{,}46\cos\left(\frac{2\pi n}{M}\right) \tag{20}$$

$$w[n] = 0{,}50 - 0{,}50\cos\left(\frac{2\pi n}{M}\right) \tag{21}$$

$$w[n] = 0{,}42 - 0{,}50\cos\left(\frac{2\pi n}{M}\right) + 0{,}08\cos\left(\frac{4\pi n}{M}\right) \tag{22}$$

The factor of 2π represents the entire spectrum of the radian axes. This technique enables the insertion of new points during the windowing process; this is due to the division of the 2π radians spectrum into smaller points to exert a more efficient filter. This occurs with the implementation of a polynomial of degree n applied to 2π, in addition to the constant factor, in order to maintain linearity. In Figs. 4 and 5, the full radian axis and the total division of the spectrum can be observed, respectively.

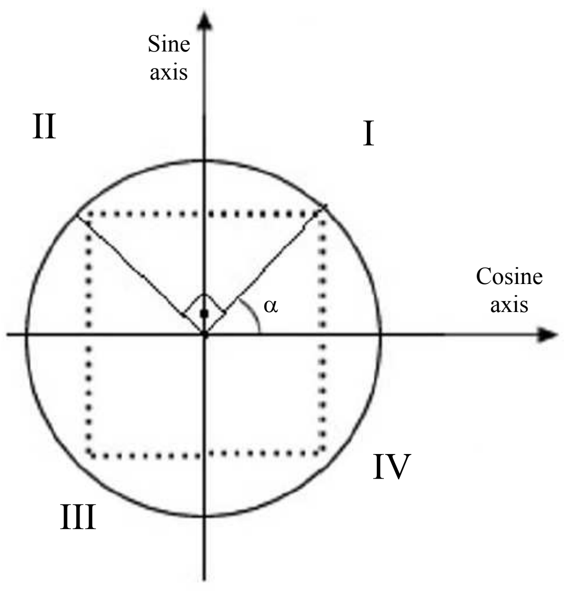


Fig. 4: Full 2π radian spectrum

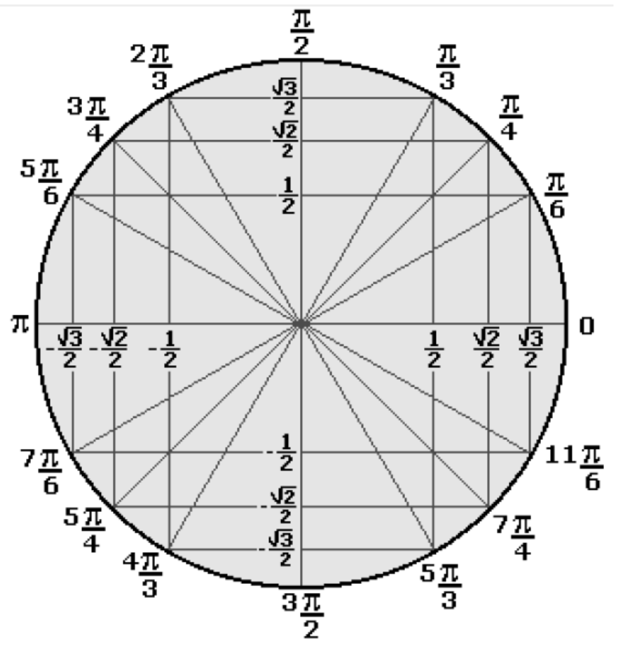


Fig. 5: Full 2π radian spectrum divided into smaller points

The central focus of this process is minimizing the effect of smaller lobes that are not included in the cut-off frequency determined for the system; this consists of the sum of window $w[n]$ with a polynomial of degree nx-1 for $nx \geq 0$. The described implementation can be visualized in (23) (i.e., the Bimbi polynomial):

$$w[n] = \left[\sum_{y=0}^{y=M} F1 - F2\cos\left(\frac{2^{(n-y)}\pi n}{M} \right) \right] G \tag{23}$$

where, G is described by (24):

$$G = \left[\sum_{n=0}^{n=M} \left(\frac{1}{w[n]} \right) \right] \tag{24}$$

Equation 25 describes the sum the polynomial applied to $w[n]$ using a windowed filter, where $F1$ is the window constant factor, $F2$ is the portion applied to the cosine, n is the current filter coefficient, M is the number of filter coefficients and G is the gain.

Implementation and Results

For development and execution of this work, the following software and equipment were used: the Visual Studio compiler (C# platform), a library with the purpose of mathematical/graphical development, and proofing by comparing the state-of-the-art techniques available (i.e., Hamming, Hann and Blackman) with the methods proposed in this study. For the application of the Visual Studio compiler with the DSPLab library, a tool is developed that is capable of carrying out digital filters, which provides a greater number of tests and comparisons by applying deterministic signals that have control of the following parameters:

a) Analog source
   - Sine
b) Amplitude (V)
c) Signal frequency (Hz)
d) Sampling frequency (Hz)
e) Digital analog converter resolution (bits)
f) Level (Vpp)
g) Choice of digital filter (e.g., a state-of-the-art filter or a filter with application of the polynomial)
   - Hamming-windowed low-pass filter
   - Hann-windowed low-pass filter
   - Blackman-windowed low-pass filter
h) Selection of the cutoff frequency applied to the digital filter
i) Digital filter dimension (M)
j) Visualization of the response in the time domain
k) Visualization of the response in the frequency domain
l) Impulse response

For the graphic results, the "IPT_Filter_Design" tool was developed based on the DSPLab library. With this tool, it was possible to run tests in the time and frequency domains with sine-type signals that had the following characteristics:

- Amplitude (5 Vdc)
- Frequency (5 Hz)
- Sampling frequency (10 kHz)
- ADC resolution (16 bit)
- Peak-to-peak voltage (5 Vdc)
- Coefficients (101)

Figure 6 shows the sine-type signal as a function of the mentioned characteristics, where tests with the Hamming, Hann and Blackman windows with a 1 Hz cut-off frequency are implemented.

Figures 7 and 8 compare the results in the time domain between a Hamming filter and Hamming filter with polynomial alteration respectively.

Figure 9 shows the code to calculates the coefficients for Hamming window in art state and polynomial form.

Figure 16 shows the code to calculates the coefficients for Hann window in art state and polynomial form.

Figures 10 and 11 compare the frequency domain results between the Hamming filter and the Hamming filter with polynomial alteration respectively.

Figures 12 and 13 compare the results of the impulse response between the Hamming filter and the Hamming filter with polynomial alteration respectively.

Figures 14 and 15 compare the results in the time domain between a Hann filter and Hann filter with polynomial alteration respectively.

Figures 17 and 18 compare the frequency domain results between the Hann filter and the Hann filter with polynomial alteration respectively.

Figures 19 and 20 compare the results of the impulse response between the Hann filter and the Hann filter with polynomial alteration respectively.

Figures 21 and 22 compare the results in the time domain between a Blackman filter and Blackman filter with polynomial alteration respectively.

Figure 23 shows the code to calculates the coefficients for Blackman window in art state and polynomial form.

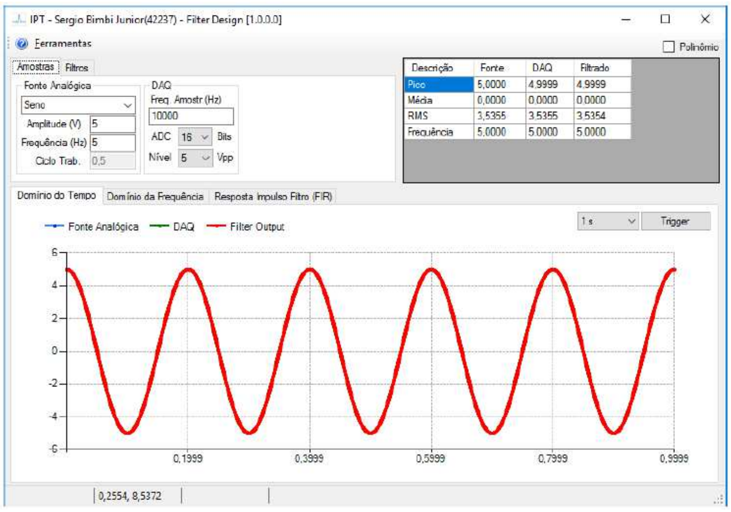
Figures 24 and 25 compare the frequency domain results between the Blackman filter and the Blackman filter with polynomial alteration respectively.

Figures 26 and 27 compare the results of the impulse response between the Blackman filter and the Blackman filter with polynomial alteration respectively.

Figures 28 and 29 compare the results of the poles and zeros between the Hamming filter and the Hamming filter with polynomial alteration (with 20 coefficients) respectively.

Figures 30 and 31 compare the results of the poles and zeros between the Hamming filter and the Hamming filter with polynomial alteration (with 100 coefficients) respectively.

Figure 32 shows the code to apply calculated coefficients in acquired samples for all cases in FIR filters.



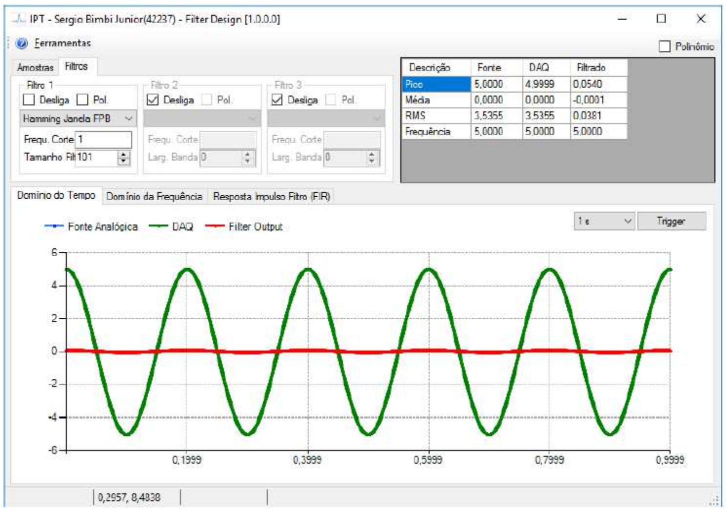
Fonte Analógica: Seno
Amplitude (V) 5
Frequência (Hz) 5
Ciclo Trab. 0,5

DAQ
Freq. Amostr (Hz) 10000
ADC 16 Bits
Nível 5 Vpp

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4.9999	4.9999
Média	0,0000	0.0000	0.0000
RMS	3,5355	3.5355	3.5354
Frequência	5,0000	5.0000	5.0000

0,2554, 8,5372

Fig. 6: Sine-type base signal



Filtro 1: ☐ Desliga ☐ Pol.
Hamming Janela FPB
Frequ. Corte 1
Tamanho Filt 101

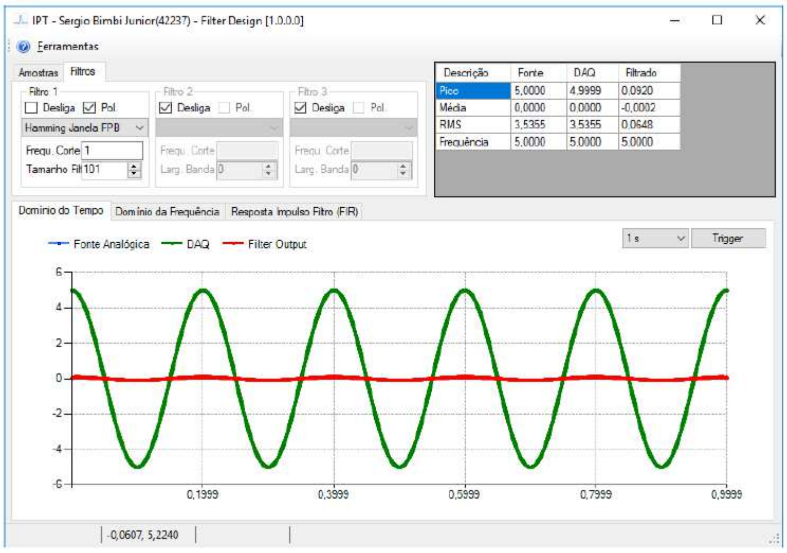
Filtro 2: ☑ Desliga ☐ Pol.
Frequ. Corte
Larg. Banda 0

Filtro 3: ☑ Desliga ☐ Pol.
Frequ. Corte
Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4.9999	0,0540
Média	0,0000	0.0000	-0,0001
RMS	3,5355	3.5355	0.0381
Frequência	5,0000	5.0000	5.0000

0,2957, 8,4838

Fig. 7: Hamming window response in the time domain (1 Hz cut-off frequency)



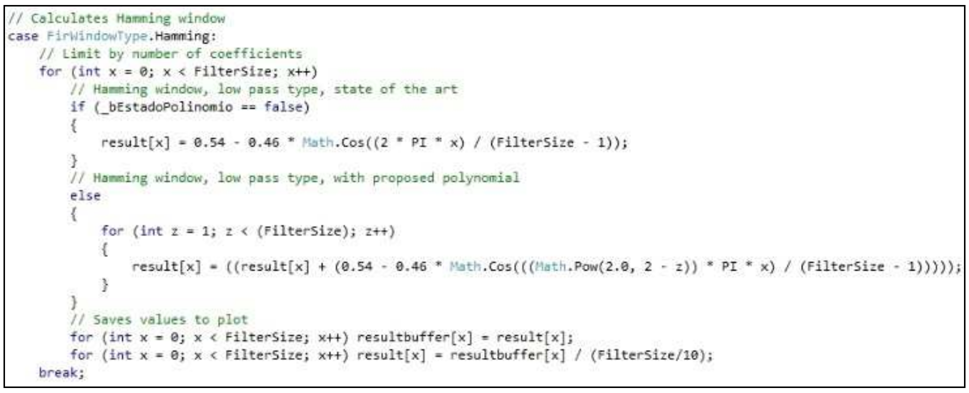
Filtro 1: ☐ Desliga ☑ Pol
Hamming Janela FPB
Frequ. Corte 1
Tamanho Filt 101

Filtro 2: ☑ Desliga ☐ Pol
Frequ. Corte
Larg. Banda 0

Filtro 3: ☑ Desliga ☐ Pol
Frequ. Corte
Larg. Banda 0

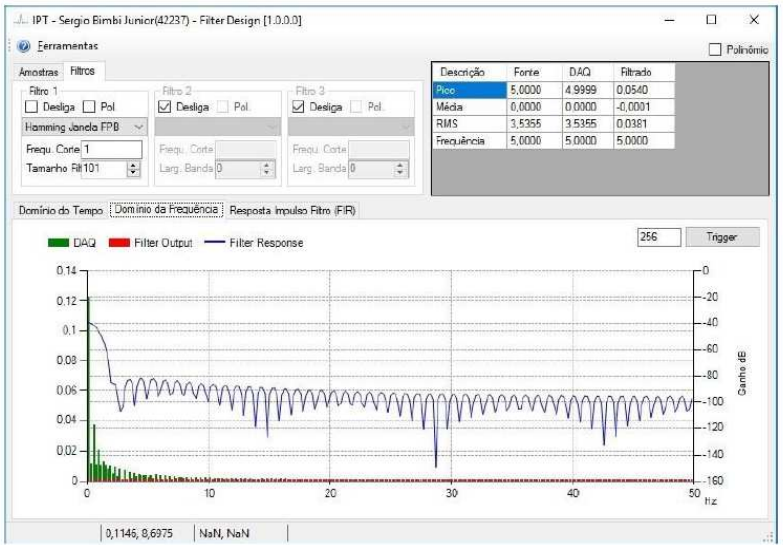
Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0920
Média	0,0000	0,0000	-0,0002
RMS	3,5355	3,5355	0,0648
Frequência	5,0000	5,0000	5,0000

Fig. 8: Hamming window response in the time domain (1 Hz cut-off frequency, changed by the polynomial)



```
// Calculates Hamming window
case FirWindowType.Hamming:
    // Limit by number of coefficients
    for (int x = 0; x < FilterSize; x++)
        // Hamming window, low pass type, state of the art
        if (_bEstadoPolinomio == false)
        {
            result[x] = 0.54 - 0.46 * Math.Cos((2 * PI * x) / (FilterSize - 1));
        }
        // Hamming window, low pass type, with proposed polynomial
        else
        {
            for (int z = 1; z < (FilterSize); z++)
            {
                result[x] = ((result[x] + (0.54 - 0.46 * Math.Cos(((Math.Pow(2.0, 2 - z)) * PI * x) / (FilterSize - 1)))));
            }
        }
        // Saves values to plot
        for (int x = 0; x < FilterSize; x++) resultbuffer[x] = result[x];
        for (int x = 0; x < FilterSize; x++) result[x] = resultbuffer[x] / (FilterSize/10);
    break;
```

Fig. 9: Hamming window code calculates the coefficients in art state and polynomial form



☐ Polinômio

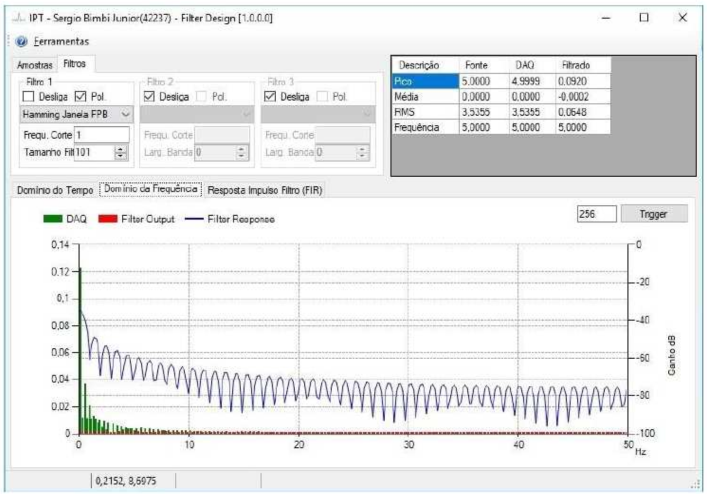
Filtro 1: ☐ Desliga ☐ Pol
Hamming Janela FPB
Frequ. Corte 1
Tamanho Filt 101

Filtro 2: ☑ Desliga ☐ Pol
Frequ. Corte
Larg. Banda 0

Filtro 3: ☑ Desliga ☐ Pol
Frequ. Corte
Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0540
Média	0,0000	0,0000	-0,0001
RMS	3,5355	3,5355	0,0381
Frequência	5,0000	5,0000	5,0000

Fig. 10: Hamming window response in the frequency domain (1 Hz cut-off frequency)



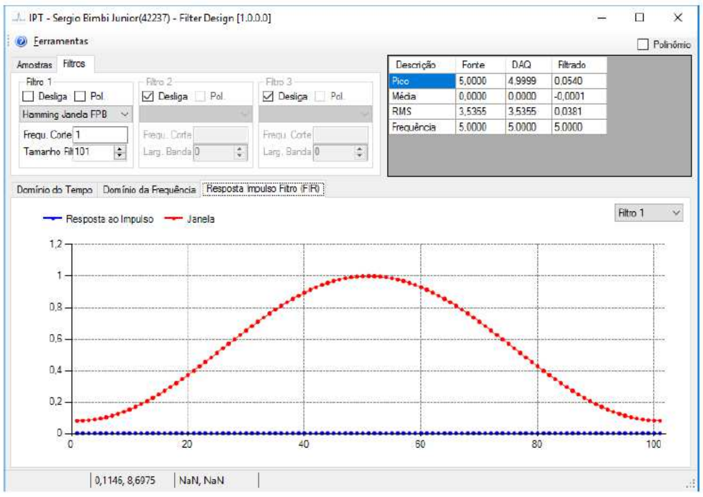
Filtro 1: ☐ Desliga ☑ Pol. — Hamming Janela FPB — Frequ. Corte 1 — Tamanho Filt 101
Filtro 2: ☑ Desliga ☐ Pol. — Frequ. Corte — Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol. — Frequ. Corte — Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0920
Média	0,0000	0,0000	-0,0002
RMS	3,5355	3,5355	0,0648
Frequência	5,0000	5,0000	5,0000

256

0,2152, 8,6975

Fig. 11: Hamming window response in the frequency domain (1 Hz cut-off frequency, changed by the polynomial)



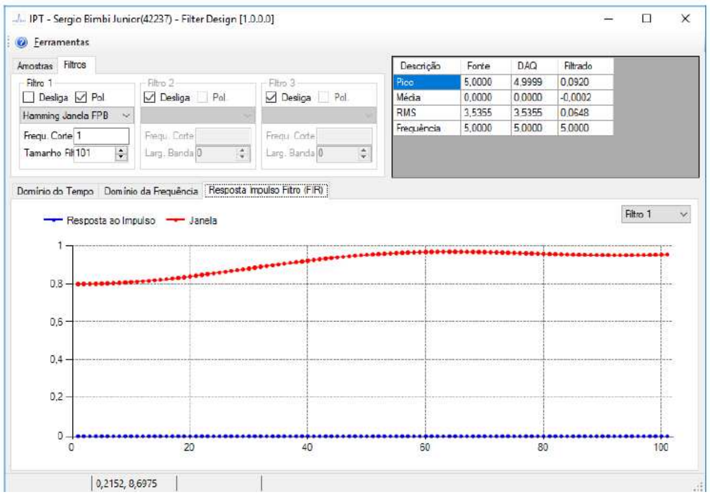
☐ Polinômio

Filtro 1: ☐ Desliga ☐ Pol. — Hamming Janela FPB — Frequ. Corte 1 — Tamanho Filt 101
Filtro 2: ☑ Desliga ☐ Pol. — Frequ. Corte — Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol. — Frequ. Corte — Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0540
Média	0,0000	0,0000	-0,0001
RMS	3,5355	3,5355	0,0381
Frequência	5,0000	5,0000	5,0000

0,1146, 8,6975 | NaN, NaN

Fig. 12: Hamming window impulse response (1 Hz cut- off frequency)

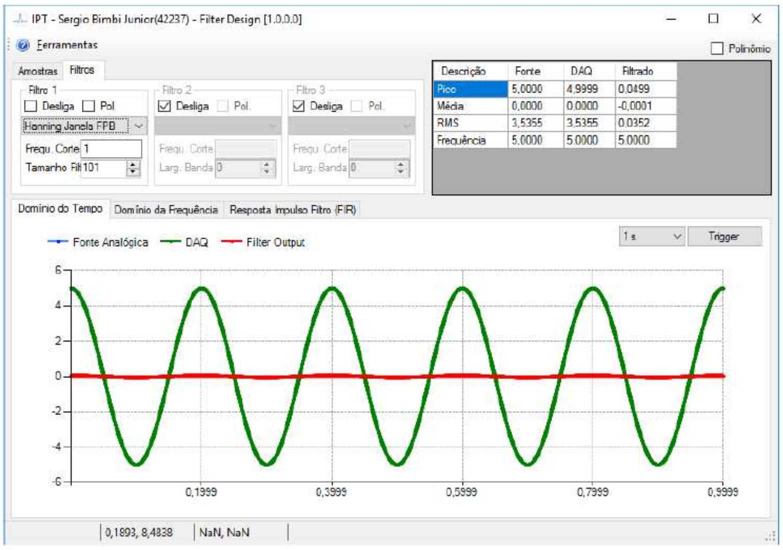


Filtro 1: ☐ Desliga ☑ Pol. — Hamming Janela FPB — Frequ. Corte 1 — Tamanho Filt 101
Filtro 2: ☑ Desliga ☐ Pol. — Frequ. Corte — Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol. — Frequ. Corte — Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0920
Média	0,0000	0,0000	-0,0002
RMS	3,5355	3,5355	0,0648
Frequência	5,0000	5,0000	5,0000

0,2152, 8,6975

Fig. 13: Hamming window impulse response (1 Hz cut-off frequency, changed by the polynomial)


IPT - Sergio Bimbi Junior(42237) - Filter Design [1.0.0.0]

Ferramentas   ☐ Polinômio

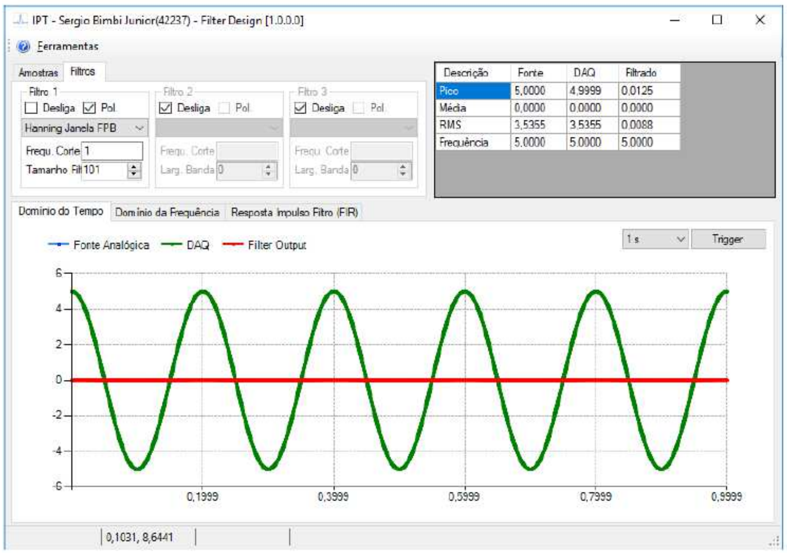
Amostras | Filtros

Filtro 1: ☐ Desliga ☐ Pol; Hanning Janela FPB; Frequ. Corte 1; Tamanho Filt 101
Filtro 2: ☑ Desliga ☐ Pol.; Frequ. Corte; Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol.; Frequ. Corte; Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0499
Média	0,0000	0,0000	-0,0001
RMS	3,5355	3,5355	0,0352
Frequência	5,0000	5,0000	5,0000

0,1893, 8,4838 | NaN, NaN

Fig. 14: Hann window response in the time domain (1 Hz cut-off frequency)


IPT - Sergio Bimbi Junior(42237) - Filter Design [1.0.0.0]

Ferramentas

Amostras | Filtros

Filtro 1: ☐ Desliga ☑ Pol; Hanning Janela FPB; Frequ. Corte 1; Tamanho Filt 101
Filtro 2: ☑ Desliga ☐ Pol.; Frequ. Corte; Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol.; Frequ. Corte; Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0125
Média	0,0000	0,0000	0,0000
RMS	3,5355	3,5355	0,0088
Frequência	5,0000	5,0000	5,0000

0,1031, 8,6441

Fig. 15: Hann window response in the time domain (1Hz cut-off frequency, changed by the polynomial)

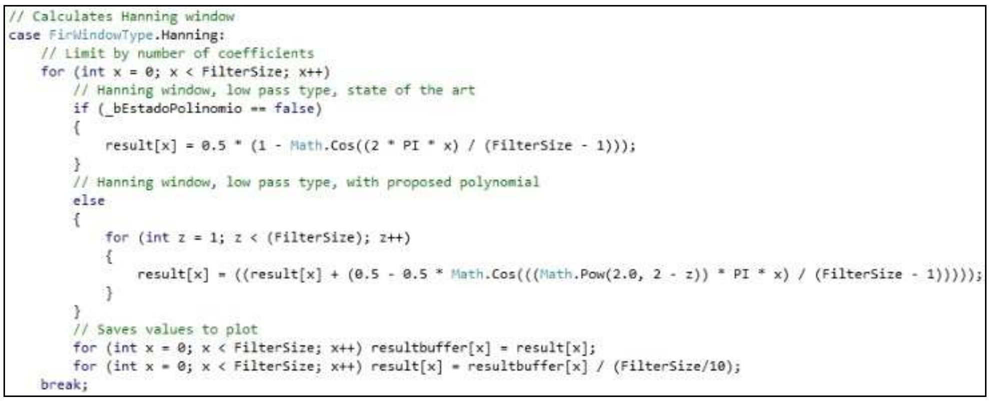
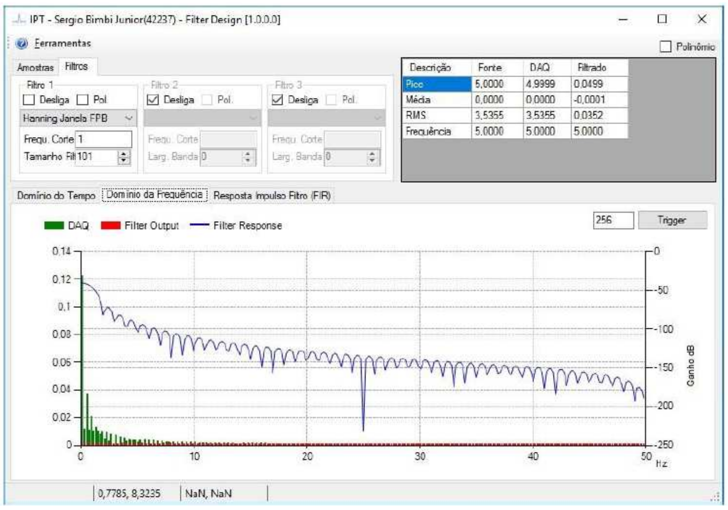

```
// Calculates Hanning window
case FirWindowType.Hanning:
    // Limit by number of coefficients
    for (int x = 0; x < FilterSize; x++)
        // Hanning window, low pass type, state of the art
        if (_bEstadoPolinomio == false)
        {
            result[x] = 0.5 * (1 - Math.Cos((2 * PI * x) / (FilterSize - 1)));
        }
        // Hanning window, low pass type, with proposed polynomial
        else
        {
            for (int z = 1; z < (FilterSize); z++)
            {
                result[x] = ((result[x] + (0.5 - 0.5 * Math.Cos(((Math.Pow(2.0, 2 - z)) * PI * x) / (FilterSize - 1)))));
            }
        }
        // Saves values to plot
        for (int x = 0; x < FilterSize; x++) resultbuffer[x] = result[x];
        for (int x = 0; x < FilterSize; x++) result[x] = resultbuffer[x] / (FilterSize/10);
    break;
```

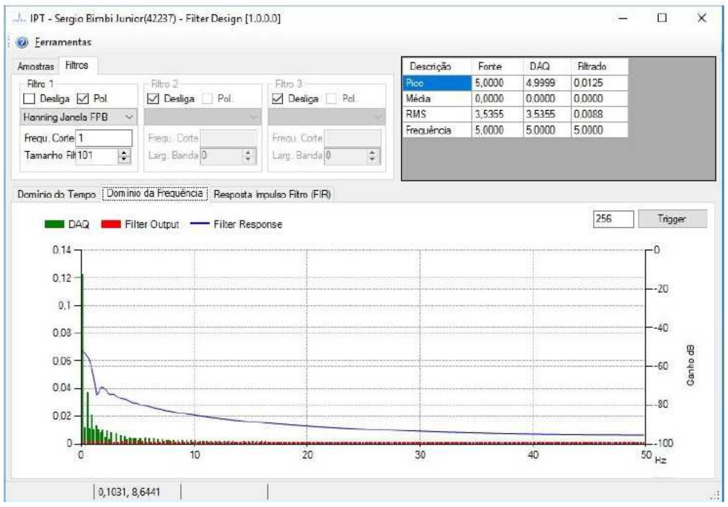
Fig. 16: Hann window code calculates the coefficients in art state and polynomial form


IPT - Sergio Bimbi Junior(42237) - Filter Design [1.0.0.0]

Filtro 1: ☐ Desliga ☐ Pol. — Hanning Janela FPB — Frequ. Corte 1 — Tamanho Filt 101

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4.9999	0.0499
Média	0,0000	0.0000	-0,0001
RMS	3,5355	3.5355	0.0352
Frequência	5,0000	5.0000	5.0000

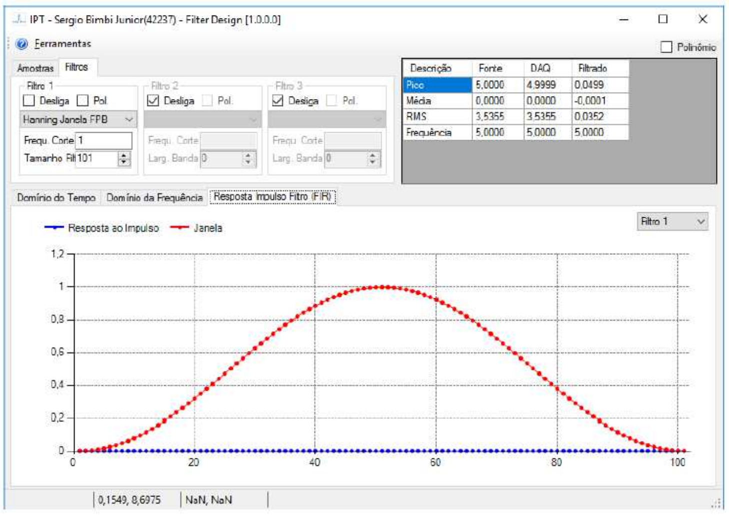
Fig. 17: Hann window response in the frequency domain (1 Hz cut-off frequency)


IPT - Sergio Bimbi Junior(42237) - Filter Design [1.0.0.0]

Filtro 1: ☐ Desliga ☑ Pol. — Hanning Janela FPB — Frequ. Corte 1 — Tamanho Filt 101

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4.9999	0.0125
Média	0,0000	0.0000	0.0000
RMS	3,5355	3.5355	0.0088
Frequência	5,0000	5.0000	5.0000

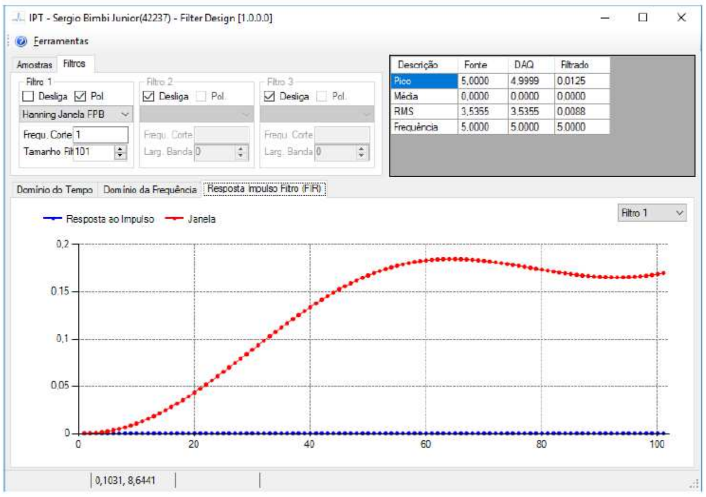
Fig. 18: Hann window response in the frequency domain (1 Hz cut-off frequency, changed by the polynomial)


IPT - Sergio Bimbi Junior(42237) - Filter Design [1.0.0.0]

Filtro 1: ☐ Desliga ☐ Pol. — Hanning Janela FPB — Frequ. Corte 1 — Tamanho Filt 101

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4.9999	0.0499
Média	0,0000	0.0000	-0,0001
RMS	3,5355	3.5355	0.0352
Frequência	5,0000	5.0000	5.0000

Fig. 19: Hann window impulse response (1 Hz cut-off frequency)

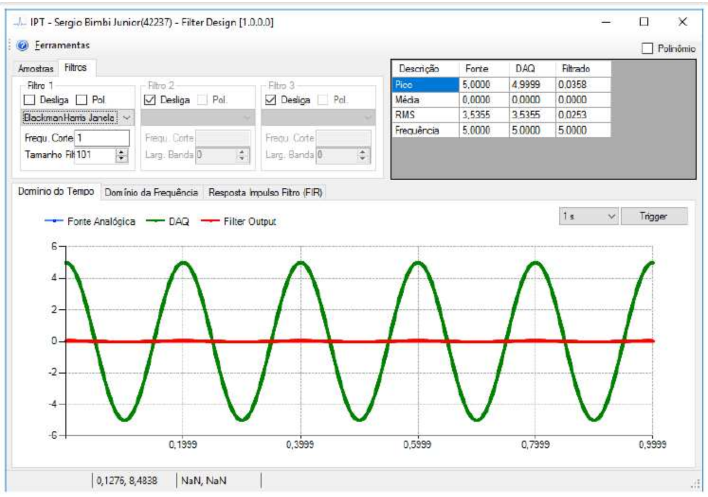


Filtro 1: ☐ Desliga ☑ Pol
Hanning Janela FPB
Frequ. Corte 1
Tamanho Filt 101

Filtro 2: ☑ Desliga ☐ Pol. Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol. Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0125
Média	0,0000	0,0000	0,0000
RMS	3,5355	3,5355	0,0088
Frequência	5.0000	5.0000	5.0000

Fig. 20: Hann window impulse response (1 Hz cut-off frequency, changed by the polynomial)



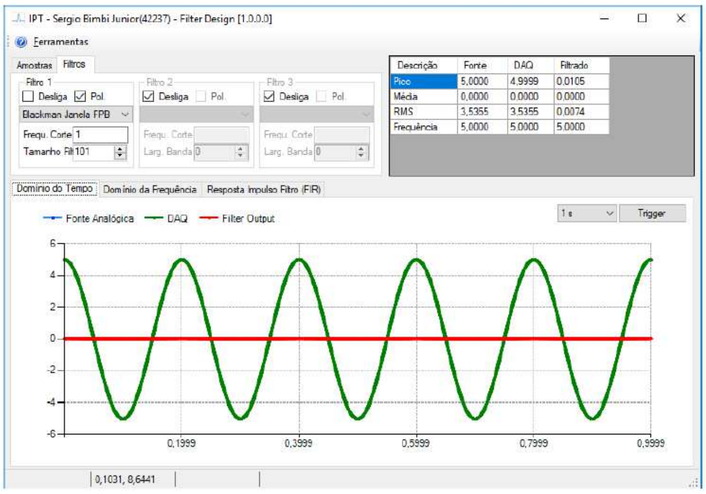
☐ Polinômio

Filtro 1: ☐ Desliga ☐ Pol
BlackmanHarris Janela
Frequ. Corte 1
Tamanho Filt 101

Filtro 2: ☑ Desliga ☐ Pol. Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol. Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0358
Média	0,0000	0,0000	0,0000
RMS	3,5355	3,5355	0,0253
Frequência	5.0000	5.0000	5.0000

Fig. 21: Blackman window response in the time domain (1 Hz cut-off frequency)

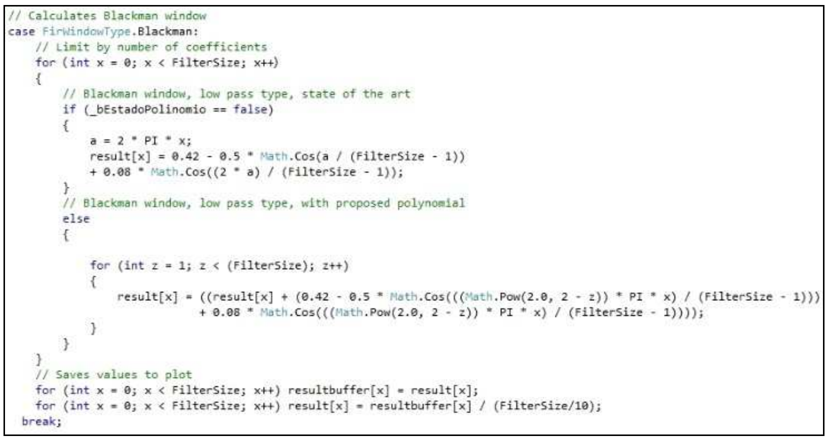


Filtro 1: ☐ Desliga ☑ Pol.
Blackman Janela FPB
Frequ. Corte 1
Tamanho Filt 101

Filtro 2: ☑ Desliga ☐ Pol. Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol. Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0105
Média	0,0000	0,0000	0,0000
RMS	3,5355	3,5355	0,0074
Frequência	5,0000	5,0000	5,0000

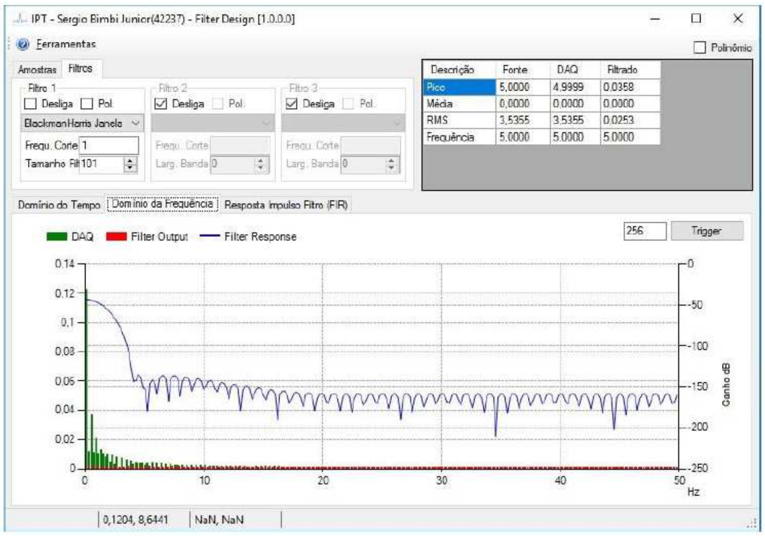
Fig. 22: Blackman window response in the time domain (1 Hz cut-off frequency, changed by the polynomial)



```
// Calculates Blackman window
case FirWindowType.Blackman:
    // Limit by number of coefficients
    for (int x = 0; x < FilterSize; x++)
    {
        // Blackman window, low pass type, state of the art
        if (_bEstadoPolinomio == false)
        {
            a = 2 * PI * x;
            result[x] = 0.42 - 0.5 * Math.Cos(a / (FilterSize - 1))
            + 0.08 * Math.Cos((2 * a) / (FilterSize - 1));
        }
        // Blackman window, low pass type, with proposed polynomial
        else
        {

            for (int z = 1; z < (FilterSize); z++)
            {
                result[x] = ((result[x] + (0.42 - 0.5 * Math.Cos(((Math.Pow(2.0, 2 - z)) * PI * x) / (FilterSize - 1)))
                            + 0.08 * Math.Cos(((Math.Pow(2.0, 2 - z)) * PI * x) / (FilterSize - 1))));
            }
        }
    }
    // Saves values to plot
    for (int x = 0; x < FilterSize; x++) resultbuffer[x] = result[x];
    for (int x = 0; x < FilterSize; x++) result[x] = resultbuffer[x] / (FilterSize/10);
  break;
```

Fig. 23: Blackman window code calculates the coefficients in art state and polynomial form



IPT - Sergio Bimbi Junior(42237) - Filter Design [1.0.0.0]

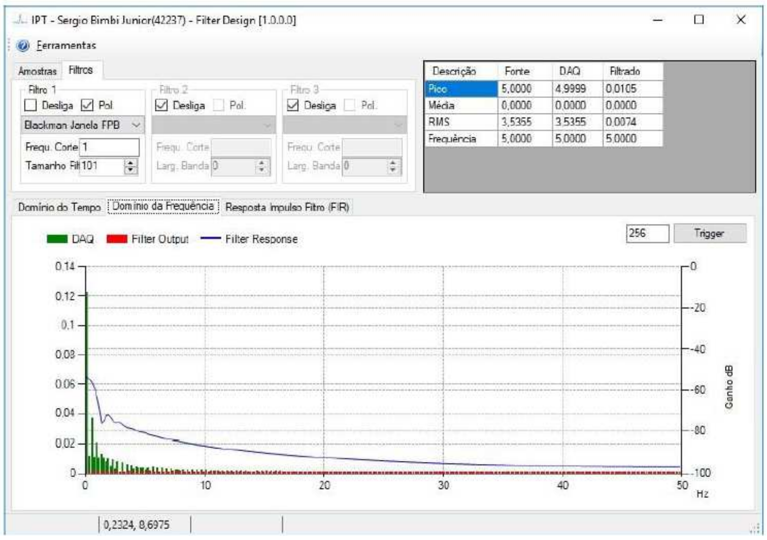
Filtro 1: ☐ Desliga ☐ Pol. — BlackmanHarris Janela — Frequ. Corte 1 — Tamanho Filt 101
Filtro 2: ☑ Desliga ☐ Pol. — Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol. — Larg. Banda 0
☐ Polinômio

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0358
Média	0,0000	0,0000	0,0000
RMS	3,5355	3,5355	0,0253
Frequência	5,0000	5,0000	5,0000

256 Trigger

0,1204, 8,6441 | NaN, NaN

Fig. 24: Blackman window response in the frequency domain (1 Hz cut-off frequency)



IPT - Sergio Bimbi Junior(42237) - Filter Design [1.0.0.0]

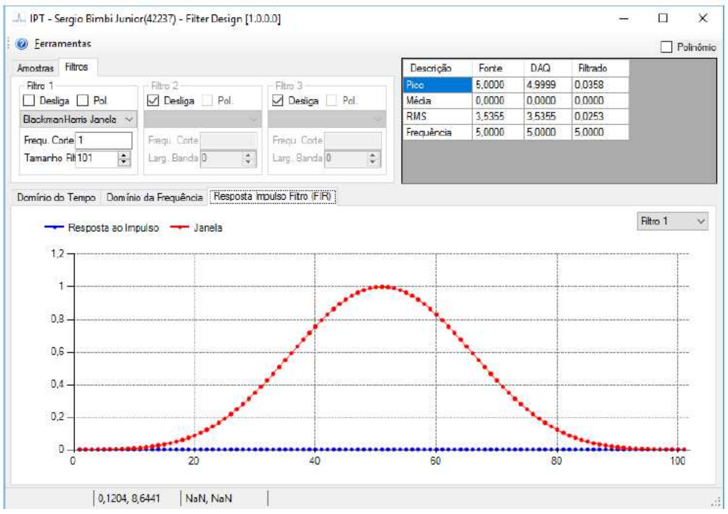
Filtro 1: ☐ Desliga ☑ Pol. — Blackman Janela FPB — Frequ. Corte 1 — Tamanho Filt 101
Filtro 2: ☑ Desliga ☐ Pol. — Larg. Banda 0
Filtro 3: ☑ Desliga ☐ Pol. — Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0105
Média	0,0000	0,0000	0,0000
RMS	3,5355	3,5355	0,0074
Frequência	5,0000	5,0000	5,0000

256 Trigger

0,2324, 8,6975

Fig. 25: Blackman window response in the frequency domain (1 Hz cut-off frequency, changed by the polynomial)



Filtro 1: ☐ Desliga ☐ Pol — BlackmanHarris Janela — Frequ. Corte 1 — Tamanho Filt 101

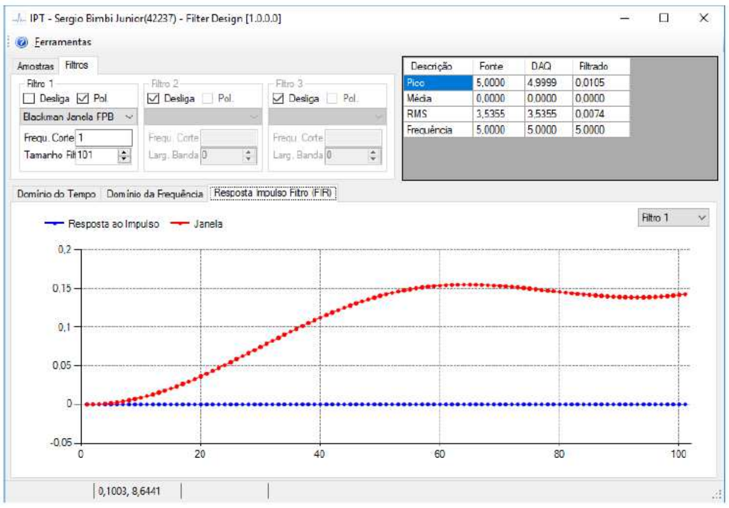
Filtro 2: ☑ Desliga ☐ Pol. — Larg. Banda 0

Filtro 3: ☑ Desliga ☐ Pol. — Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0358
Média	0,0000	0,0000	0,0000
RMS	3,5355	3,5355	0,0253
Frequência	5,0000	5,0000	5,0000

0,1204, 8,6441 | NaN, NaN

Fig. 26: Blackman window impulse response (1 Hz cut- off frequency)



Filtro 1: ☐ Desliga ☑ Pol — Blackman Janela FPB — Frequ. Corte 1 — Tamanho Filt 101

Filtro 2: ☑ Desliga ☐ Pol. — Larg. Banda 0

Filtro 3: ☑ Desliga ☐ Pol. — Larg. Banda 0

Descrição	Fonte	DAQ	Filtrado
Pico	5,0000	4,9999	0,0105
Média	0,0000	0,0000	0,0000
RMS	3,5355	3,5355	0,0074
Frequência	5,0000	5,0000	5,0000

0,1003, 8,6441

Fig. 27: Blackman window impulse response (1 Hz cut- off frequency, changed by the polynomial)

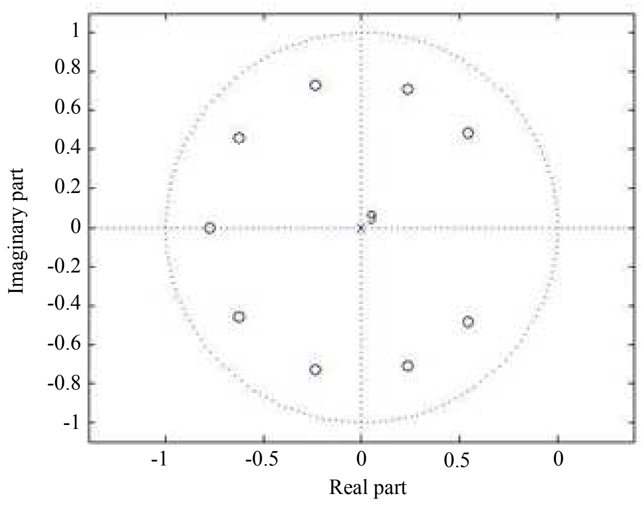


Fig. 28: Poles and zeros of the Hamming-windowed filter with 20 coefficients

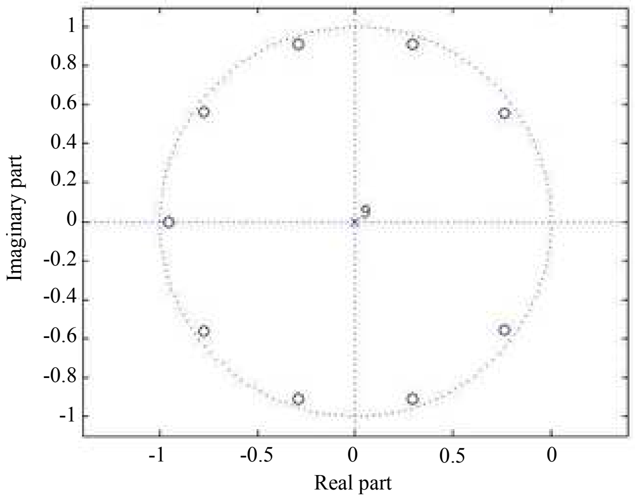


Fig. 29: Poles and zeros of the Hamming-windowed filter with 20 coefficients (changed by the polynomial)

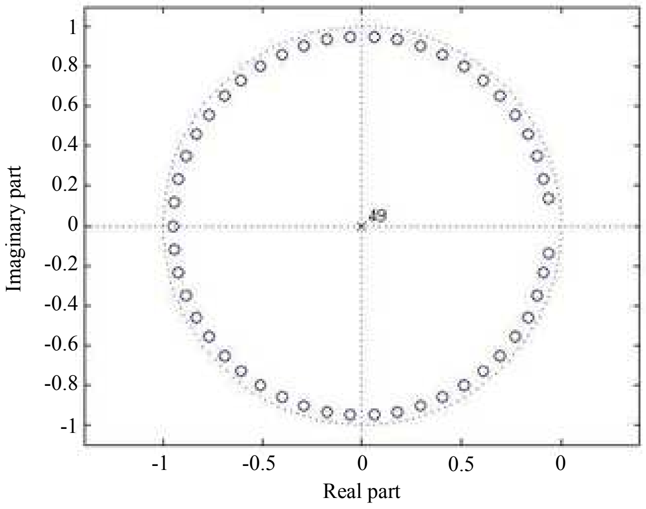


Fig. 30: Poles and zeros of the Hamming-windowed filter with 100 coefficients

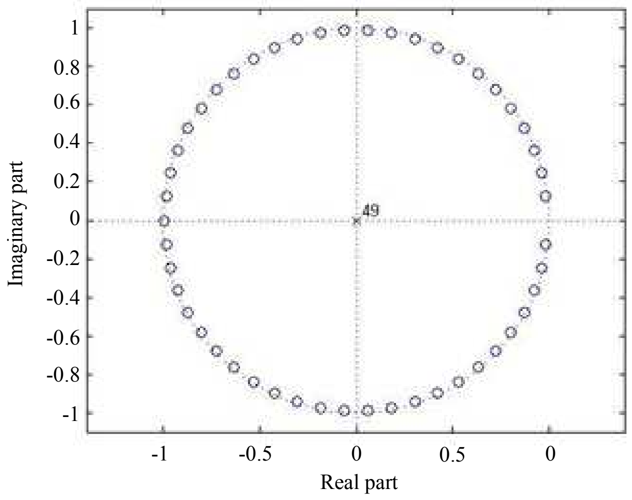
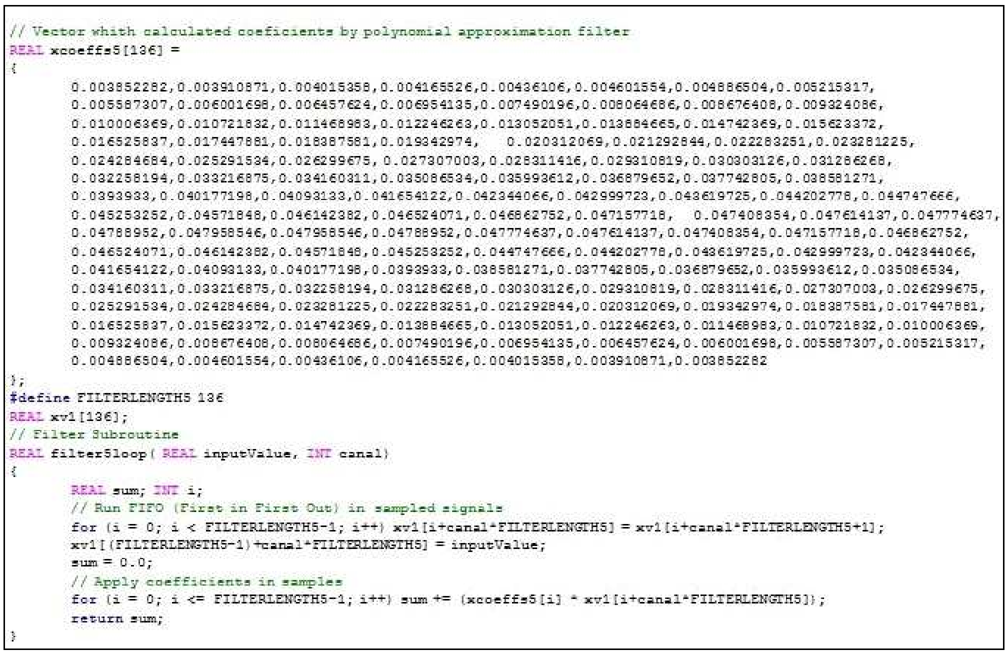


Fig. 31: Poles and zeros of the Hamming-windowed filter with 100 coefficients (changed by the polynomial)



```
// Vector whith calculated coeficients by polynomial approximation filter
REAL xcoeffs5[136] =
{
        0.003852282,0.003910871,0.004015358,0.004165526,0.00436106,0.004601554,0.004886504,0.005215317,
        0.005587307,0.006001698,0.006457624,0.006954135,0.007490196,0.008064686,0.008676408,0.009324086,
        0.010006369,0.010721832,0.011468983,0.012246263,0.013052051,0.013884665,0.014742369,0.015623372,
        0.016525837,0.017447881,0.018387581,0.019342974,    0.020312069,0.021292844,0.022283251,0.023281225,
        0.024284684,0.025291534,0.026299675, 0.027307003,0.028311416,0.029310819,0.030303126,0.031286268,
        0.032258194,0.033216875,0.034160311,0.035086534,0.035993612,0.036879652,0.037742805,0.038581271,
        0.0393933,0.040177198,0.04093133,0.041654122,0.042344066,0.042999723,0.043619725,0.044202778,0.044747666,
        0.045253252,0.04571848,0.046142382,0.046524071,0.046862752,0.047157718,   0.047408354,0.047614137,0.047774637,
        0.04788952,0.047958546,0.047958546,0.04788952,0.047774637,0.047614137,0.047408354,0.047157718,0.046862752,
        0.046524071,0.046142382,0.04571848,0.045253252,0.044747666,0.044202778,0.043619725,0.042999723,0.042344066,
        0.041654122,0.04093133,0.040177198,0.0393933,0.038581271,0.037742805,0.036879652,0.035993612,0.035086534,
        0.034160311,0.033216875,0.032258194,0.031286268,0.030303126,0.029310819,0.028311416,0.027307003,0.026299675,
        0.025291534,0.024284684,0.023281225,0.022283251,0.021292844,0.020312069,0.019342974,0.018387581,0.017447881,
        0.016525837,0.015623372,0.014742369,0.013884665,0.013052051,0.012246263,0.011468983,0.010721832,0.010006369,
        0.009324086,0.008676408,0.008064686,0.007490196,0.006954135,0.006457624,0.006001698,0.005587307,0.005215317,
        0.004886504,0.004601554,0.00436106,0.004165526,0.004015358,0.003910871,0.003852282
};
#define FILTERLENGTH5 136
REAL xv1[136];
// Filter Subroutine
REAL filter5loop( REAL inputValue, INT canal)
{
        REAL sum; INT i;
        // Run FIFO (First in First Out) in sampled signals
        for (i = 0; i < FILTERLENGTH5-1; i++) xv1[i+canal*FILTERLENGTH5] = xv1[i+canal*FILTERLENGTH5+1];
        xv1[(FILTERLENGTH5-1)+canal*FILTERLENGTH5] = inputValue;
        sum = 0.0;
        // Apply coefficients in samples
        for (i = 0; i <= FILTERLENGTH5-1; i++) sum += (xcoeffs5[i] * xv1[i+canal*FILTERLENGTH5]);
        return sum;
}
```

Fig. 32: Code to apply calculated coefficients in acquired samples for all cases in FIR filters

Conclusion

Based on the results obtained, the behavior of signals in the time domain and the frequency domain there is a possibility of reducing the group delay (Oppenheim and Schafer, 2010), once the zone of convergence is increased, allowing the reduction of taps (Oppenheim and Schafer, 2010). This feature is of paramount importance, reducing computational effort as well as obtaining tangible values more quickly. In this way, it allows the increase of operation speed in dynamic systems. And, for the three proposed tests, i.e., Hamming window, Hann window and Blackman window, it is possible to observe the change in response in both the time domain and the frequency domain with a 50% gain in the frequency domain. For the Hamming and Blackman windows, it is possible to observe attenuation of the response by rejecting the unwanted frequency more efficiently. In the frequency domain, the best slope for the cut-off frequency in the first conduction lobe can be observed. Additionally, in these cases, windows behave with a larger spectrum and thus demonstrate the largest number of stable samples. In the case of the Hann window, the response to the frequency domain is more efficient, with the first driving lobe sloping the cut-off frequency more accurately. However, due to the gain, we obtain a greater amplitude than the state-of-the-art response for the time domain. The spectrum response was also shown to be more efficient by demonstrating an increased number of stable samples. With respect to the behavior of the poles and the zeros, an interesting characteristic observed is the inclination of the poles to always tend towards the unit radius (independent of the number of M coefficients). This fact indicates an improved response with respect to the target (i.e., reduced distortion). It is still necessary to analyze practical tests with the proposed case studies in this study by checking samples and calculating averages and standard deviations to verify whether the polynomial shows improved efficiency. In practical cases, implemented in checkweigher with mass 36.2g with filter implemented in art state obtains standard deviation of 0.2696, with polynomial for the standard deviation of 0.1318. This factor represents an improvement of 48.88%. With speeds until 200 packages per minute in a belt with 250 mm of length. It is also necessary to test other windows in future studies, as well as other features (e.g., changing the filter type). With these tests, the efficiency of other applications could be determined, such as digital TV, aerospace studies, image segmentation, and other applications (i.e., any application that uses digital filters). An interesting topic to be developed in future studies is the implementation of a "*Code Generator*" with the "IPT_Filter_Design" tool, which could export coefficients together with the First-In-First-Out (FIFO) structure for application in industrial processes by using a Programmable Logic Controller (PLC) or microprocessor.

Acknowledgement

The authors would like to thank the support of their respective institutions.

Author's Contributions

Sérgio Bimbi Junior: Research about Digital Filters; Advanced Mathematics; Statistical Analysis; Studies and experiments about digital filters; Fourier Transform; Distribution Analysis, Statistics, Implementation and Development Software.

Vitor Chaves de Oliveira: Research about Implementation and Development Software, Statistical Analyses, Graphs Development, Optimization Software and Algorithms.

Agenor de Toledo Fleury: Research and development of Mathematical Modelling and interpretation of Results, Overall software adjustments.

Ronaldo Ruas: Research and development of Mathematical Modelling and interpretation of Results, Overall software adjustments.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

Adams, J.W. and A.N. Willson Jr., 1983. Anew approach to FIR digital filters with fewer multipliers and reduced sensitivity. IEEE Trans. Circuits Syst., CAS-30: 277-83. DOI: 10.1109/TCS.1983.1085356

Adams, J.W. and A.N. Willson Jr., 1984. Some efficient digital prefilter structures. IEEE Trans. Circuits Syst., CAS-31: 260-5. DOI: 10.1109/TCS.1984.1085492

Ahmed, N., T. Natarajan and K.R. Rao, 1974. Discrete cosine transform. IEEE Trans. Computers, C-23: 90-3. DOI: 10.1109/T-C.1974.223784

Akansu, A.N. and M.J. Medley, 1999. Wavelets, Subband and Block Transforms in Communications and Multimedia. Boston, MA: Kluwer Academic Publishers.

AMIN, 2011. Hardware approach of a multipurpose finite impulse response filter for real-time filtering applications. Am. J. Applied Sci., 12: 1272-1281. DOI: 10.3844/ajassp.2011.1272.1281

Antoniou, A. and M.G. Rezk, 1977. Digital filters synthesis using concept of generalize dimmitance converter. IEEE Trans. Circuits Syst., CAS-27: 1184-94. DOI: 10.1109/TCS.1980.1084766

Antoniou, A., 1982. Accelerated procedure for the design of equiripple nonrecursive digital filters. IEE Proceedings – Part G, 129: 1-10. DOI: 10.1049/ip-g-1.1982.0001

Antoniou, A., 1993. Digital Filters: Analysis, Design and Applications, 2nd Edn., New York, NY: McGraw-Hill.

Antoniou, A., 2006. Digital Signal Processing: Signals, Systems, and Filters. In: Practical Optimization: Algorithms and Engineering Applications New York, Antoniou, A. and W.S. Lu (Eds.), McGraw-Hill, New York.

Avenhaus, E., 1972. On the design of digital filters with coefficients of limited word length. IEEE Trans. Audio Electroacoustics, 20: 206-12. Bauer, P.H. and J. Wang, 1993. Limit cycle bounds for floating point implementations of second-order recursive digital filters. IEEE Transactions Circuits Systems II: Analog and Digital Signal Processing, 40: 493-501. DOI: 10.1109/82.242338

Belevitch, V., 1968. Classical Network Theory. San Francisco, CA: Holden-Day.

Benvenuto, N., L.E. Franks and F.S. Hill, Jr., 1984. On the design of FIR filters with power-of-two coefficients. IEEE Trans. Communications, 32: 1299-307. DOI: 10.1109/TCOM.1984.1096001

Bimbi Jr. S., F. Agenor de Toledo, R. Ronaldo and V. Chaves de Oliveira, 2016a. Distortion reduction in FIR filters by approximation through window factor on function coefficients. Set Int. J. Broadcast Eng., 2: 1-9. DOI: 10.18580/setijbe.2016.3

Bimbi Jr. S., F. Agenor de Toledo, R. Ronaldo and V. Chaves de Oliveira, 2016b. Application of Distortion Reduction in FIR Filters in Dynamic Systems through Computational Methods. Set Int. J. Broadcast Eng., 2: 2446-9432. DOI: 10.18580/setijbe.2016.1

Bomar, B.W. and R.D. Joseph, 1987. Calculation of L∞ norms in second-order state-space digital filter sections. IEEE Trans. Circuits Syst., 34: 983-4. DOI: 10.1109/TCS.1987.1086231

Bomar, B.W., 1989. On the design of second-order state-space digital filter sections. IEEE Trans. Circuits Syst., 36: 542-52. DOI: 10.1007/978-1-4615-6199-6

Bomar, B.W., L.M. Smith and R.D. Joseph, 1997. Roundoff noise analysis of state-space digital filters implemented on floating-point digital signal processors. IEEE Trans. Circuits Systems II: Analog and Digital Signal Processing, 44: 952-5. DOI: 10.1109/82.644048

Boyd, S. and L. Vandenberghe, 2004. Convex Optimization. Cambridge, UK: Cambridge University Press. Bracewell, R. N. (1984). The fast Hartley transform. Proceedings IEEE, 72, 1010-18.

Burrus, C.S. and T.W. Parks, 1970. Time domain design of recursive digital filters. IEEE Transactions Audio Electroacoustics, 20: 137-41. DOI: 10.1109/TAU.1970.1162093

Butterweck, H.J., 1975. Suppression of parasitic oscillations in second-order digital filters by means of a controlled- rounding arithmetic. Archiv Elektrotechnik und Übertragungstechnik, 29: 371-4.

Butterweck, H.J., van Meer, A.C.P. and G. Verkroost, 1984. New second-order digital filter sections without limit cycles. IEEE Trans. Circuits Syst., 31: 141-6. DOI: 10.1109/TCS.1984.1085477

Cabezas, J.C.E. and P.S.R. Diniz, 1990. FIR filters using interpolated prefilters and equalizers. IEEE Trans. Circuits Syst., 37: 17-23. DOI: 10.1109/31.45687

Oppenheim, A.V. and R.W. Schafer, 2010. Discrete-Time Signal Processing, 3rd Edn., Georgia: Pearson, pp: 1120.

PANICH, 2010. Indirect Kalman filter in mobile robot application. J. Mathematics Statistics, 6: 381-384. DOI: 10.3844/jmssp.2010.381.384

Senthilkum, A. and A.M. Natarajan, 2008. FPGA Implementation of power aware FIR filter using reduced transition pipelined variable precision gating. J. Computer Sci., 4: 87-94. DOI: 10.3844/jcssp.2008.87.94

TAGHOUTI, 2010. How to find wave-scattering parameters from the causal bond graph model of a high frequency filter. Am. J. Applied Sci., 7: 702-710. DOI: 10.3844/ajassp.2010.702.710