Extended SINAD

Individual Project
M.Eng Information Technology
Submitted By : Yash Vyas 1266490
Email: vyas@stud.fra-uas.de
Guide : Prof. Dr. Pech

ACKNOWLEDGEMENT

I would like to thank Prof. Dr. Pech, for the continuous support and guidance provided for this project. Furthermore, I would also like to put my sincere gratitude for providing all the necessary information and useful links which proved to be very helpful for the successful completion of this project

TABLE OF CONTENTS

Abstract—**In the following project different functions have been implemented for calculating ADC's and DAC's dynamic parameter SINAD From the signal analysis perspective frequency domain is used for the calculation of SINAD. Function is tested with different kinds of data and the output is compared with the SINAD function implemented in MATLAB's Signal Processing Toolbox as well as theoretically calculated SINAD**

Keywords—SINAD, SNDR, Sampling, FFT, DFT, Signal, Coherent, Non-coherent, Data Window, Decibels, Windowing

I. Introduction

In the field of the modern electronic system, ADC and DAC are the fundamental block. They form a captious link between front end analog transducers and back end digital computers that can competently implement a wide variety of signal processing functions, which leads to a variety of ADC and DAC in terms of performance, price, and quality. [1]
Due to the large diversity of applications, there are different figures of merit available for the measurement of performance of ADC and DAC which are loosely divided into three categories static parameters, time-domain dynamic parameters and frequency domain dynamic parameters

The static parameter includes: Accuracy, resolution, Dynamic Range, Offset error, Gain error, Differential Nonlinearity, Integral Nonlinearity, Missing codes. The time-domain dynamic parameter includes Aperture Delay, Aperture Jitter, Transient Response, overvoltage Recovery. Frequency domain dynamic parameter includes Signal to noise distortion ratio [SINAD], Effective number of bits [ENOB], Spurious-Free Dynamic Range [SFDR] , Total Harmonic Distortion [THD], Intermodulation Distortion [IMD], Effective Resolution Bandwidth [ERBW], Full power Bandwidth [FPBW], Full linear Bandwidth. [1]

II. Signal to noise and distorion ratio description

A. Definition

Signal-to-noise-and-distortion ratio (S/N+D, SINAD, or SNDR) is the ratio of the input signal amplitude to the rms [Root mean Square] sum of all other spectral components in frequency domain.

In other words it is equal to the ratio of total received power of the signal to the noise-plus-distortion power

B. Mathematical Representation

i. In terms of Power

$$\boldsymbol{SINAD = 10\log_{10}\left[\frac{P_{signal}}{P_{noise}+P_{distortion}}\right]} \quad (1)$$

[2]

$$P_{signal} = Power\ of\ Fundamental\ Signal$$
$$P_{noise} = Noise\ Power$$
$$P_{distortion} = Distortion\ Power$$

ii. In terms of SNR [Signal Noise Ratio] and THD [Total Harmonic Distortion] is:

$$\boldsymbol{SINAD = 20\log_{10}\sqrt{10^{\left(\frac{-SNR}{10}\right)} + 10^{\left(\frac{THD}{10}\right)}}} \quad (2)$$

[2]

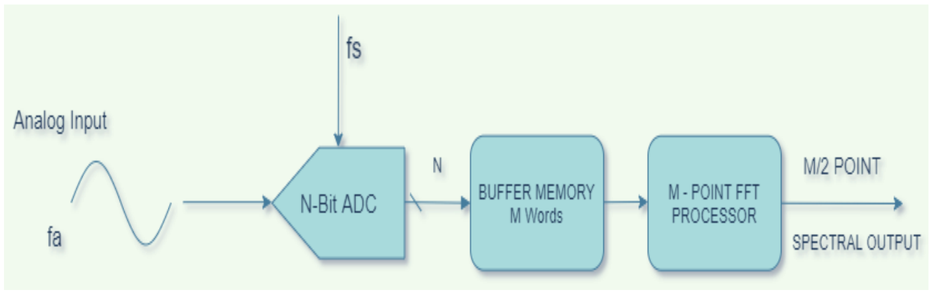


Figure 1 Generalized Test setup for FFT analysis of ADC output [2]

$$INPUT = ANALOG\ SIGNAL$$
$$OUTPUT = \frac{M}{2}\ Points\ [In\ Frequency\ Domain]$$
$$M = Number\ of\ Samples\ stored\ in\ Buffer\ Memory$$
$$Noise\ Range = dc - \frac{f_s}{2}\ [Nyquist\ Bandwidth]$$
$$Bin\ width = \frac{f_s}{M}$$

$$Noise\ Floor = 10 * \log\left(\frac{M}{2}\right)[Theoretical]$$

iii. SINAD representation in terms of Noise

$$\boldsymbol{SINAD} = -\mathbf{20}\log\sqrt{\frac{2}{3}\left[\frac{\sqrt{\frac{BW}{100}}(1+DNL)}{2^8}\right]^2 + \left(\frac{2\pi Tj}{10^6}\right)^2 + \left(2\sqrt{2}*\frac{V_n}{2^8}\right)^2 + \left(\frac{THD\%}{100}\right)^2}\ dB$$

(3) **[3]**

iv. SINAD IN TIME DOMAIN

$$\boldsymbol{SINAD[dB]} = \mathbf{10}\log\left(\frac{MA^2}{2\sum_{n=0}^{M-1}(y_n-e_n)^2}\right) \quad (4)$$

[4]

$M = Data\ record\ length$
$A = Amplitude$
$y_n = ADC\ output\ data$
$e_n = Sampled\ Input\ Signal$

v. SINAD IN Frequency DOMAIN

$$SINAD = 10*log\left[A_k^2\left(\sum_{l=1}^{k-1}A_l^2 + \sum_{l=k+1}^{K/2}A_l^2\right)^{-1}\right]$$

(5) **[1]**

$A_k = Fundamental\ Amplitude$
$k = Frequency\ Bin$

C. *Mathematical Relations between SINAD,THD,SNR,ENOB*

Following relationship derivation are referenced from [2]

$$\boldsymbol{SNR} = \mathbf{20}*\log\left(\frac{S}{N}\right) \quad (6)$$

$$\boldsymbol{THD} = \mathbf{20}*\log\left(\frac{S}{D}\right) \quad (7)$$

$$\boldsymbol{SINAD} = \mathbf{20}*\log\left(\frac{S}{N+D}\right) \quad (8)$$

Eq 6, 7, 8 can be solved for numerical ratios as follows:

$$\frac{N}{S} = \mathbf{10}^{-\frac{SNR}{20}} \quad (9)$$

$$\frac{D}{S} = \mathbf{10}^{-\frac{THD}{20}} \quad (10)$$

$$\frac{N+D}{S} = \mathbf{10}^{-\frac{SINAD}{20}} \quad (11)$$

From the Eq 9, 10, 11:

$$\frac{N+D}{S} = \left[\left(\frac{N}{S}\right)^2 + \left(\frac{D}{S}\right)^2\right]^{\frac{1}{2}} = \left[\left(10^{-\frac{SNR}{20}}\right)^2 + \left(10^{-\frac{THD}{20}}\right)^2\right]^{\frac{1}{2}}$$

(12)

$$\frac{N+D}{S} = \left[\mathbf{10}^{\frac{-SNR}{10}} + \mathbf{10}^{-\frac{THD}{10}}\right]^{\frac{1}{2}} \quad (13)$$

$$\frac{S}{N+D} = \left[\mathbf{10}^{-\frac{SNR}{10}} + \mathbf{10}^{-\frac{THD}{10}}\right]^{-\frac{1}{2}} \quad (14)$$

And hence,

$$SINAD = 20*\log\left(\frac{S}{N+D}\right) = 10*log\left[10^{-\frac{SNR}{10}} + 10^{-\frac{THD}{10}}\right]$$

(15)

$$\boldsymbol{ENOB} = \frac{\boldsymbol{SINAD}-\mathbf{1.76}}{\mathbf{6.02}} \quad (16)$$

Following relation holds true only if the input frequency and amplitude is same for all three measurements

III. SINGAL ANALYSIS BACKGROUND

Following section gives brief introduction about the mathematical concepts used in the field of Signal analysis

A. *Fourier Transformation*

a) Discrete Fourier Transform:

Fourier Transform is a mathematical transformation of a signal into its constituent's frequencies

$$\boldsymbol{X(w)} = \int_{-\infty}^{+\infty}\boldsymbol{x(t)e^{-jwt}dt} \quad (17)\ \mathbf{[5]}$$

$t = time\ variable$
$w = frequency$

Members of Fourier Transform family:

- Fourier Transform
- Fourier Series
- Discrete Fourier Transform
- Discrete Time Fourier Transform

Following members also includes there inverse transformation function with them

Mathematical Notation for Inverse Fourier Transform

$$\boldsymbol{x(t)} = \int_{-\infty}^{+\infty}\boldsymbol{X(w)e^{j2\pi wt}dw} \quad (18)\ \mathbf{[5]}$$

$x(t) = weighted\ sum\ of\ complex\ exponentials$

Mathematical Notation:

$$\boldsymbol{X(kw_o) = \sum_{n}^{N-1} x[n] * (\cos\left(\frac{2\pi kn}{N}\right) + i.\sin(\frac{2\pi kn}{N}))}$$
(19)

b) Fast Fourier Transform:

Following section is referenced from [5]
Fast Fourier transform is a mathematical method for transforming a function of time into a function of frequency. It is described as transforming from the time domain to the frequency domain.
The Fast Fourier transform (FFT) is a development of the Discrete Fourier transform (DFT) which removes duplicated terms in the mathematical algorithm to reduce the number of mathematical operations performed. In this way, it is possible to use large numbers of samples without compromising the speed of the transformation. The FFT reduces computation by a factor of N/ (log2 (N)).

c) Properites of Fourier Transform

Following section is referenced from [5]

$\boldsymbol{SampleRate = Max\_Sig\_Freq}$ *2*1.25 **(20)**
$AbsoluteMinFactor = 1.25$

Absolute Min Factor is used for getting the right values also in the upper region of the FFT.
This is the equation another way around famous Nyquist criteria, which says that maximal signal frequency adequately presented in the digitized wave is the half of the sampling rate.

$$\boldsymbol{LineRes = \frac{SampleRate}{2*NumberOfLines}} \quad (21)$$

$$\boldsymbol{TimeToCalculate = (\frac{No.OfLines*2}{SampleRate})} \quad (22)$$

From Eq 23 and 24 we get Eq 25

$$\boldsymbol{LineRes = \frac{1}{TimeToCalculate}} \quad (23)$$

B. Windowing

In real-life scenarios, it is difficult to get exactly integer number of cycles which leads to Non-Coherent data. A Coherently sampled waveform intrinsically has an integer number of cycles that means waveform starts and ends at the same level. For solving the problem of Non-Coherent data windowing is applied to the data which forces the data to begin and end at the same or nearly the same level.

From Mathematical orientation, windowing is achieved by multiplying sampled waveform data set by an appropriate window function which prevents discontinuity at the window edges.

Spectral Leakage occurs due to the fraction number of frequency cycles of fundamental signals. Windowing helps to reduce spectral leakage. It is typically chosen such that FFT produces best spectral purity near the fundamental frequency. The amount of spectral leakage depends on the window shape and how the signal fits into the window.

Some common types of windowing function are:

a) Hamming Window

The customary cosine sum windows for case K = 1have the form:

$$w[n] = a_0 - (1 - a_0) * \cos\left(\frac{2\pi n}{N}\right), \quad 0 \le n \le N$$
(24) [10]

Setting a0 = 0.54 to produce Hamming windows

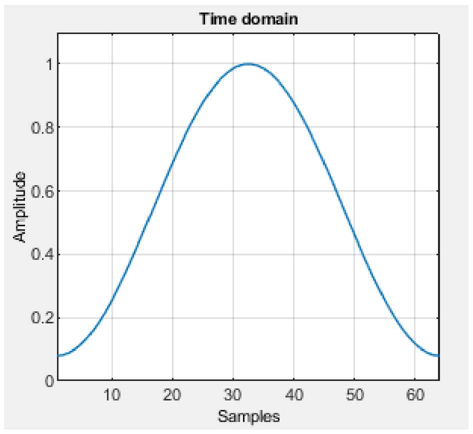


Figure 2 Hamming Window in Time Domain

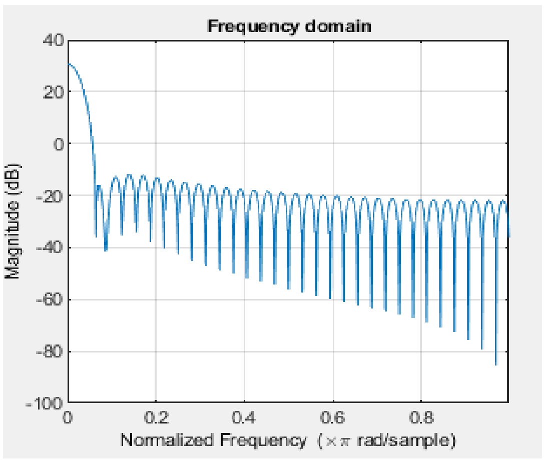


Figure 3 Hamming Window in Frequency Domain

b) Hann Window

$$\boldsymbol{w(n) = 0.5\left(1 - \cos\left(\frac{2\pi n}{N}\right)\right) \quad 0 \le n \le N} \quad \textbf{(25) [10]}$$

Window length $L = N + 1$

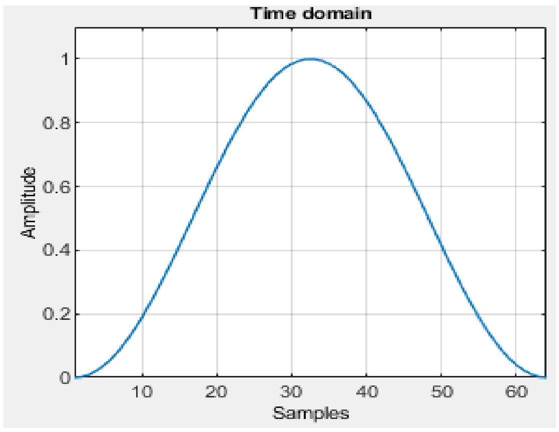


Figure 4 Hann Window in Time Domain

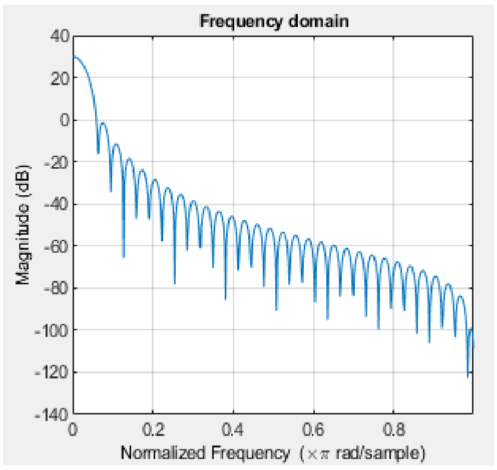


Figure 5 Hann Window in Freq Domain

c) Kaiser Window

$$w[n] = \frac{I_0\left(\pi\alpha\sqrt{1 - \left(\frac{2\pi}{N} - 1\right)^2}\right)}{I_0(\pi\alpha)}, \qquad 0 \le n \le N$$

(26) [10]

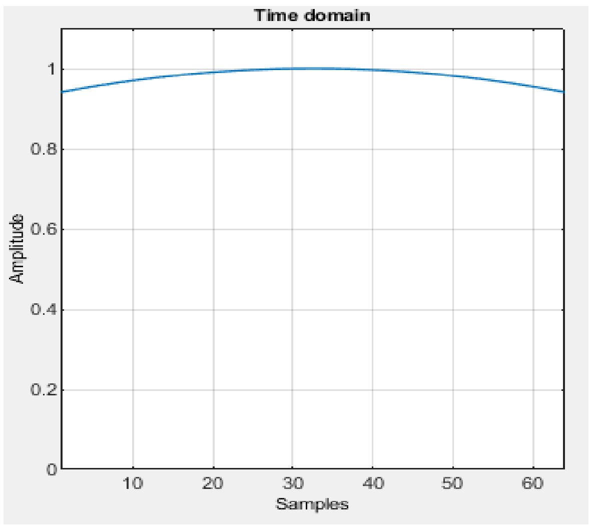


Figure 6 Kaiser Window in Time Domain

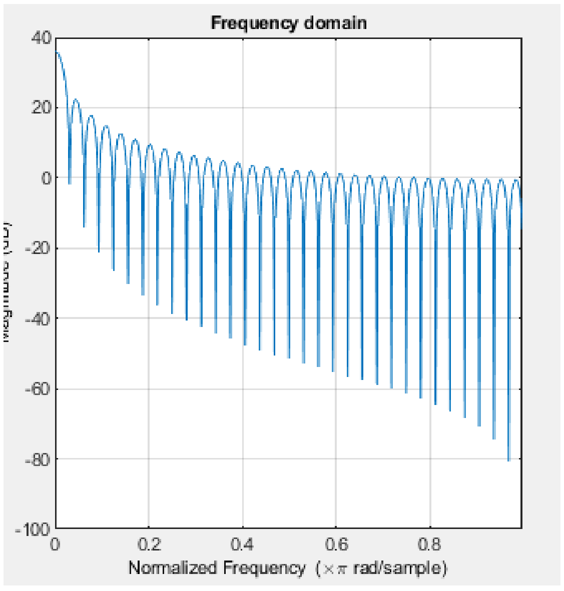


Figure 7 Kaiser Window in Freq Domain

d) Gauusian Window

$$w[n] = \exp\left(-\frac{1}{2}\left(\frac{n-\left(\frac{N}{2}\right)}{\frac{\sigma N}{2}}\right)^2\right), \qquad 0 \leq n \leq N$$

(27) [10]

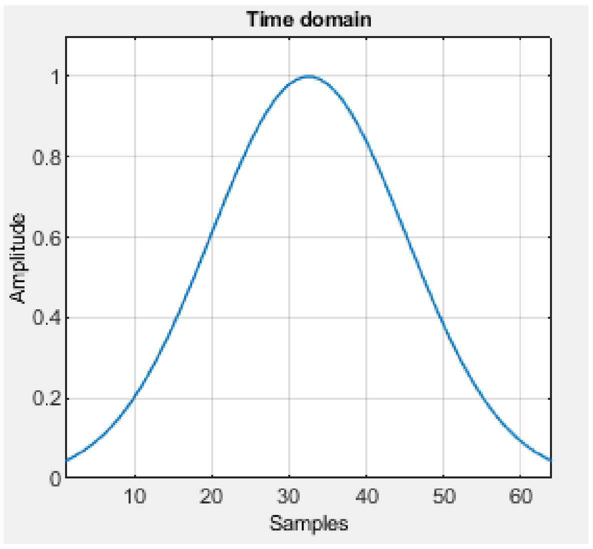


Figure 8 Gaussian Window in Time Domain

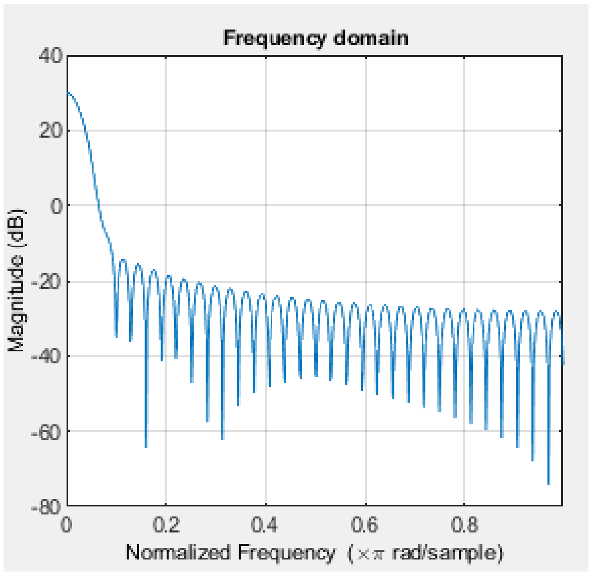


Figure 9 Gaussian Window in Freq Domain

C. Sampling

Mathematical process of converting a continuous time signal to discrete time signal for computers to process the data digitally is known as Sampling

In the sampling group there are two type available:

a) Coherent Samping

When the sampled waveform data include an integer number of the cycle then sampling is known as Coherent sampling, in other words, for Coherent sampling, the relation in (28) should hold true. It dramatically improves the FFT spectral purity and builds an ideal mathematical ground for evaluating system performance.

$$\frac{f_{in}}{f_s} = \frac{M}{N} \qquad \textbf{(28)}$$

$$f_s = Sampling\ Frequency$$
$$f_{in} = Input\ Frequency$$
$$M = Integer\ Number\ of\ Cycles\ in\ data\ record$$
$$N = factor\ of\ 2\ ,number\ \ of\ samples\ in\ the\ record$$

Relation in (28) will work for any arbitrary M and N, but practical values provide better results. Theoretically, N is chosen as a power of 2 as FFT requires the number of samples to be the power of 2 because of its intrinsic periodicity.
For DFT, N can be any arbitrary number, M should be odd or prime due to which all common factors are eliminated. The uniqueness of M is adjuratory to SNR and THD calculations. Common factors between M and N values lead to different harmonics of bin having the same frequency in FFT after aliasing which in turn corrupts the THD and SNR measurement

b) Non-Coherent Samping

Non-coherent data is produced when M in (28) is not an integer. FFT or DFT of Non-Coherent data will produce a spectrum with leakage and spurs around the fundamental frequency which can be seen in the Figure. For frequency domain analysis of such data, the mathematical function of windowing is applied beforehand which approximates the signal quality.

D. Decibel scale

In many scientific disciplines for comparing some value with the reference value the logarithmic unit used in known as decibels. Chronically the reference value is likely to be the smallest or largest. The quantities of measurement when changes drastically, it is inefficient to use a linear scale in such conditions dB scale is preferred.

For example large power ratio are expressed as Equation 30

$$A(B) = \log(\frac{P_1}{P_2}) \quad (29)$$

For small power reference ratios are expressed as Equation 31

$$A(dB) = 10 * \log(\frac{P_1}{P_2}) \quad (30)$$

Relative power ratios can be expressed as Equation 32

$$10 * \log\left(\frac{A_1^2}{A_2^2}\right) = 20 * \log(\frac{A_1}{A_2}) \quad (31)$$

E. *Nyquist Sampling Theorm*

Bandlimited signal is a signal x (t) which does not have spectral component beyond frequency F Hz [7].

$$X(w) = 0, \ |w| > 2\pi F \quad (32)$$

Nyquist sampling theorem states that a real bandlimited signal can be reconstructed without error from samples taken uniformly at rate R > 2F samples per second.

Following minimum frequency is known as Nyquist Frequency.

$$F_S = 2F \ Hz \quad (33)$$

Corresponding sampling interval is known as Nyquist interval [8]

$$T = \frac{1}{2F} \quad (34)$$

Signal sampled at $F > F_S$ is known as under sampled signal.

Signal sampled at $F > Fs$ is known as over sampled Signal

IV. Project version 1 Details

Following section provides in depth details regarding the project design, implementation, testing and manual

A. *Version 1 Aim:*

Implementation of function which will calculate following version of SINAD:

1. Classical SINAD:
   In following SINAD calculation version we calculate fundamental frequency from the data and calculate SINAD
2. Extended SINAD:
   In the following SINAD calculation user gives the input frequency on the basis of which the SINAD calculation are done.
3. Extended SINAD bundled:
   In the following SINAD calculation user gives multiple frequency on the basis of which the bundled SINAD calculation are done.

B. *Technology Used:*

MatLab R2019b

C. *Project Architecture*

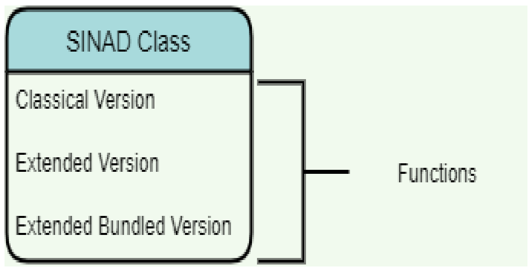


Figure 10 Project Architecture

D. *Algorithm*

1) *Classical SINAD*

Algorithm : Classical SINAD

Given Inputs $X = \{x_1, x_2, \dots, x_N\}, x_i \in \mathbb{R}^d$ *Signal data in time domain*;
Output : Signal to Noise Distortion Ratio

```
1.  Switch nargin
2.  ……case 3 : check window input argument
3.      **If input argument is out of predefined set**
4.          **Generate Window Error**
5.      **Else Calculate windowed signal** $wx = X * \dots\dots\dots\dots window$
6   ……**END**
7.      case 2 : If SamplingRate <= 0 **Generate ………..Sampling Error**
8.      Case 1 : Assigning sampling rate as 1
9.      Remove Dc Offset $wx = wx - mean(wx)$
10.     Datapoints = length($wx$)
11.     FFT_transformation of $abs(wx)$
12.     Removing DC bin i.e. Bin 1
13.     FreqLines = SamplingRate*((Datapoints/2)-1)
```

```
14.         Calculation Fundamental Freq
15.         **Calculating** Numerator $= f_{fund}^2$
16.     …   **Calculating** Denominator $= \sum FFt_{sinal}^2 - f_{fund}^2$
17.     ……**Calculating Sinad Ratio.**
18.     **Return** $Sinad$
```

2) Extended SINAD:

Algorithm : Extended SINAD

Given Inputs
$X = \{x_1, x_2, \dots, x_N\}, x_i \in \mathbb{R}^d$ *Signal data in time domain*; *Sampling Rate, Windowing , Input Freq*
Output : Signal to Noise Distortion Ratio

```
1.  Switch nargin
2.  … … case 3 : check window input argument
3.        **If input argument is out of predefined set**
4.            **Generate Window Error**
5.        **Else Calculate windowed signal**
                  $wx = X * window$
6   ……**END**
7.        case 2 : If SamplingRate <= 0 **Generate ………..Sampling Error**
8.        Case 1 : Assigning sampling rate as 1
9.        Remove Dc Offset $wx = wx - mean(wx)$
10.       Datapoints = length**(**$wx$**)**
11.       FFT_transformation of $abs(wx)$
12.       Removing DC bin i.e. Bin 1
13.       FreqLines = SamplingRate*((Datapoints/2)-1)
14.       Checking Input Freq
15. ……**If** Input Freq is not in present frequency range
16. ……….**Genrate** Error
17. ……**End**
18. ……**Remove Spectral Leakage**
15.       **Calculating** Numerator $= f_{fund}^2$
16. …     **Calculating** Denominator $= \sum FFt_{sinal}^2 - f_{fund}^2$
17. ……**Calculating Sinad Ratio.**
18. **Return** $Sinad$
```

3) Extended Bundled SINAD:

Algorithm : Extended Bundled SINAD

Given Inputs
$X = \{x_1, x_2, \dots, x_N\}, x_i \in \mathbb{R}^d$ *Signal data in time domain*; *Sampling Rate, Windowing , Input Freq array*
Output : Signal to Noise Distortion Ratio

```
1.  Check input frequency array
2.  … … **If** input freq array is having error
4.          **Generate Error**
5.      **Else**
6   ……**For each Input Frequency Calculate SINAD using Extended algorithm**
7.          Sum Each SINAD
8.    **Return Bundled** $Sinad$
```

E. FlowChart

1) Classical:

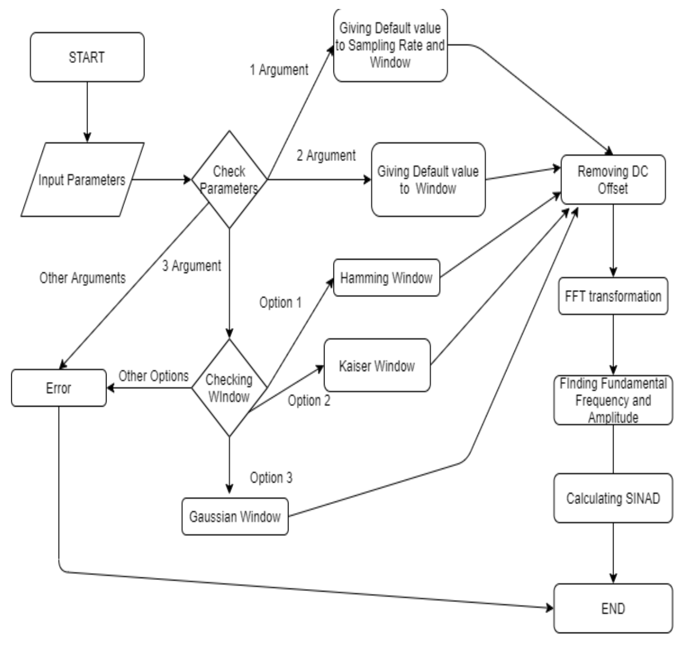


Figure 9 FLOWCHART FOR CLASSICAL SINAD

2) Extended:

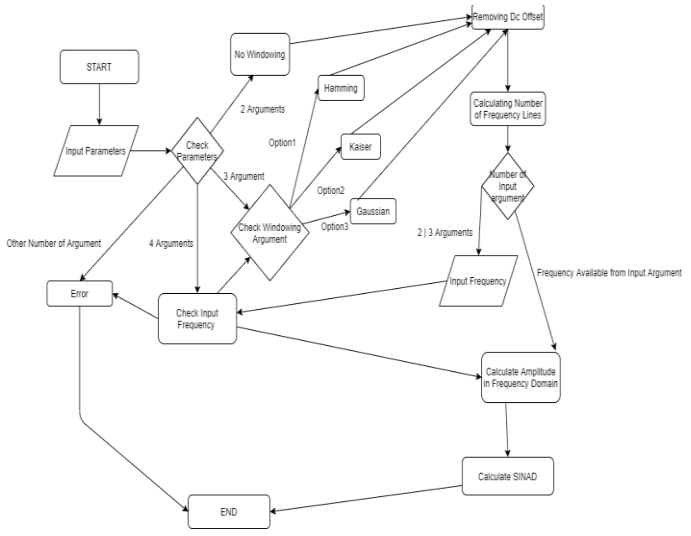


Figure 10 FLOWCHART FOR EXTENDED SINAD

3) *Extended Bundled*

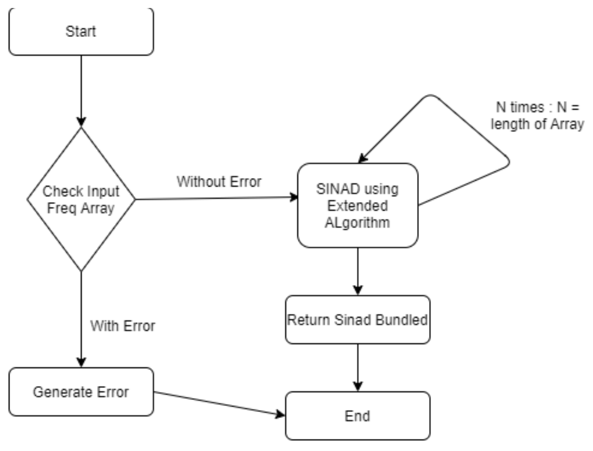
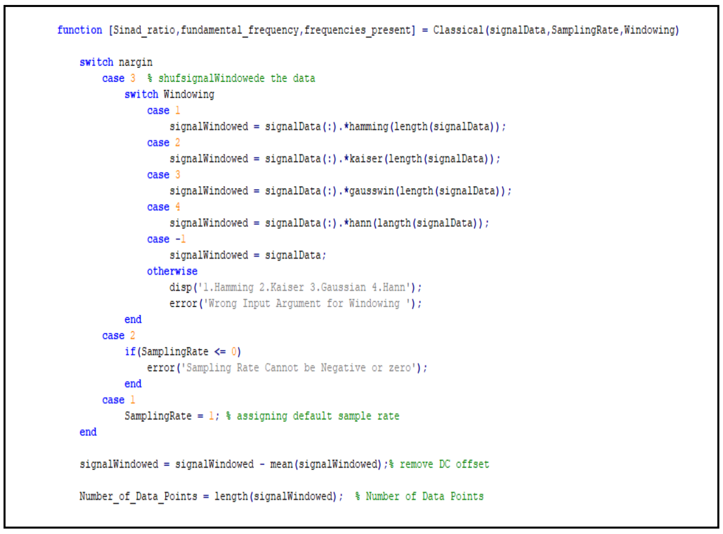


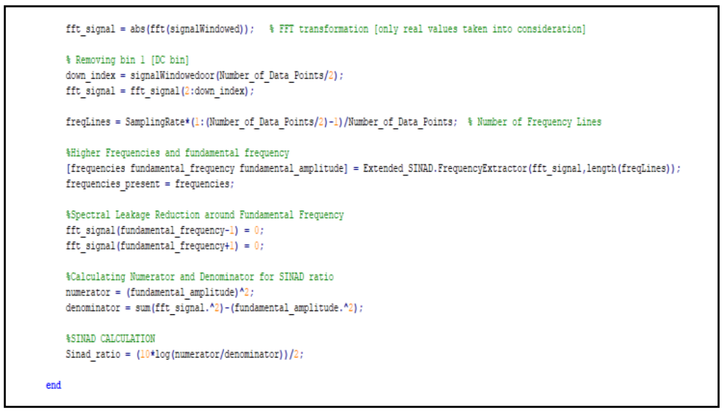
Figure 11 Extended Bundled SINAD flowchart

F. *Code Snapshot*

1) *Classical version:*

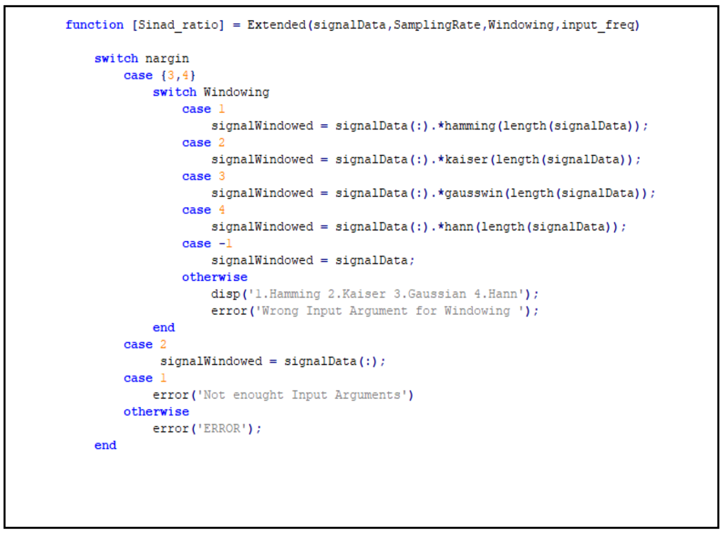


```
function [Sinad_ratio,fundamental_frequency,frequencies_present] = Classical(signalData,SamplingRate,Windowing)

    switch nargin
        case 3  % shufsignalWindowede the data
            switch Windowing
                case 1
                    signalWindowed = signalData(:).*hamming(length(signalData));
                case 2
                    signalWindowed = signalData(:).*kaiser(length(signalData));
                case 3
                    signalWindowed = signalData(:).*gausswin(length(signalData));
                case 4
                    signalWindowed = signalData(:).*hann(langth(signalData));
                case -1
                    signalWindowed = signalData;
                otherwise
                    disp('1.Hamming 2.Kaiser 3.Gaussian 4.Hann');
                    error('Wrong Input Argument for Windowing ');
            end
        case 2
            if(SamplingRate <= 0)
                error('Sampling Rate Cannot be Negative or zero');
            end
        case 1
            SamplingRate = 1; % assigning default sample rate
    end

    signalWindowed = signalWindowed - mean(signalWindowed);% remove DC offset

    Number_of_Data_Points = length(signalWindowed);  % Number of Data Points
```

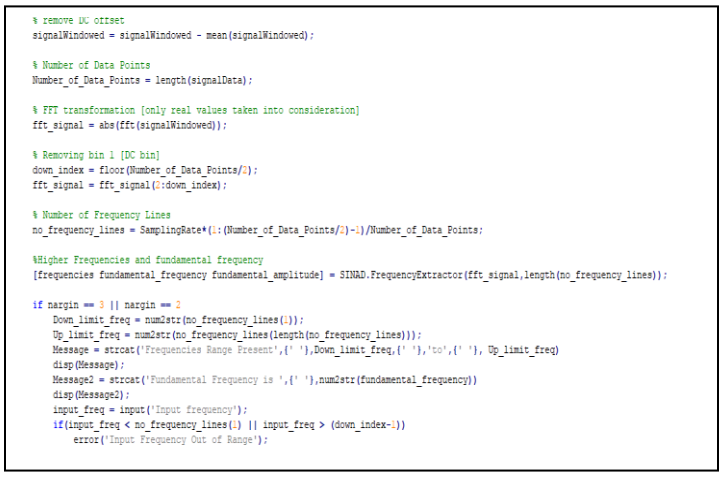


```
        fft_signal = abs(fft(signalWindowed));   % FFT transformation [only real values taken into consideration]

        % Removing bin 1 [DC bin]
        down_index = signalWindowedoor(Number_of_Data_Points/2);
        fft_signal = fft_signal(2:down_index);

        freqLines = SamplingRate*(1:(Number_of_Data_Points/2)-1)/Number_of_Data_Points;  % Number of Frequency Lines

        %Higher Frequencies and fundamental frequency
        [frequencies fundamental_frequency fundamental_amplitude] = Extended_SINAD.FrequencyExtractor(fft_signal,length(freqLines));
        frequencies_present = frequencies;

        %Spectral Leakage Reduction around Fundamental Frequency
        fft_signal(fundamental_frequency-1) = 0;
        fft_signal(fundamental_frequency+1) = 0;

        %Calculating Numerator and Denominator for SINAD ratio
        numerator = (fundamental_amplitude)^2;
        denominator = sum(fft_signal.^2)-(fundamental_amplitude.^2);

        %SINAD CALCULATION
        Sinad_ratio = (10*log(numerator/denominator))/2;

    end
```

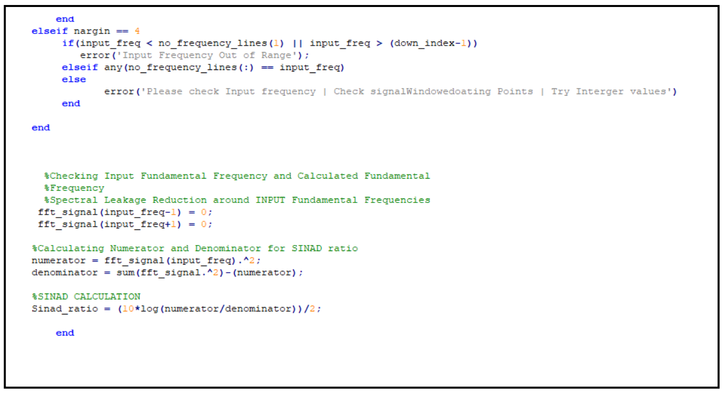
2) *Extended version:*



```
    function [Sinad_ratio] = Extended(signalData,SamplingRate,Windowing,input_freq)

        switch nargin
            case {3,4}
                switch Windowing
                    case 1
                        signalWindowed = signalData(:).*hamming(length(signalData));
                    case 2
                        signalWindowed = signalData(:).*kaiser(length(signalData));
                    case 3
                        signalWindowed = signalData(:).*gausswin(length(signalData));
                    case 4
                        signalWindowed = signalData(:).*hann(length(signalData));
                    case -1
                        signalWindowed = signalData;
                    otherwise
                        disp('1.Hamming 2.Kaiser 3.Gaussian 4.Hann');
                        error('Wrong Input Argument for Windowing ');
                end
            case 2
                 signalWindowed = signalData(:);
            case 1
                error('Not enought Input Arguments')
            otherwise
                error('ERROR');
        end
```

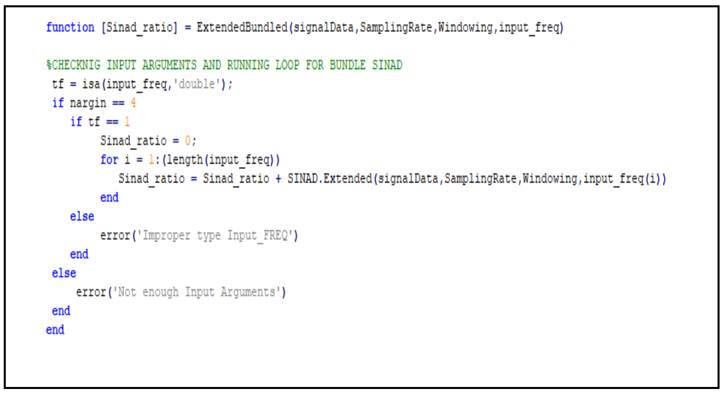


```
    % remove DC offset
    signalWindowed = signalWindowed - mean(signalWindowed);

    % Number of Data Points
    Number_of_Data_Points = length(signalData);

    % FFT transformation [only real values taken into consideration]
    fft_signal = abs(fft(signalWindowed));

    % Removing bin 1 [DC bin]
    down_index = floor(Number_of_Data_Points/2);
    fft_signal = fft_signal(2:down_index);

    % Number of Frequency Lines
    no_frequency_lines = SamplingRate*(1:(Number_of_Data_Points/2)-1)/Number_of_Data_Points;

    %Higher Frequencies and fundamental frequency
    [frequencies fundamental_frequency fundamental_amplitude] = SINAD.FrequencyExtractor(fft_signal,length(no_frequency_lines));

    if nargin == 3 || nargin == 2
        Down_limit_freq = num2str(no_frequency_lines(1));
        Up_limit_freq = num2str(no_frequency_lines(length(no_frequency_lines)));
        Message = strcat('Frequencies Range Present',{' '},Down_limit_freq,{' '},'to',{' '}, Up_limit_freq)
        disp(Message);
        Message2 = strcat('Fundamental Frequency is ',{' '},num2str(fundamental_frequency))
        disp(Message2);
        input_freq = input('Input frequency');
        if(input_freq < no_frequency_lines(1) || input_freq > (down_index-1))
            error('Input Frequency Out of Range');
```



```
        end
    elseif nargin == 4
         if(input_freq < no_frequency_lines(1) || input_freq > (down_index-1))
            error('Input Frequency Out of Range');
         elseif any(no_frequency_lines(:) == input_freq)
         else
                error('Please check Input frequency | Check signalWindowedoating Points | Try Interger values')
         end

    end



      %Checking Input Fundamental Frequency and Calculated Fundamental
      %Frequency
      %Spectral Leakage Reduction around INPUT Fundamental Frequencies
     fft_signal(input_freq-1) = 0;
     fft_signal(input_freq+1) = 0;

    %Calculating Numerator and Denominator for SINAD ratio
    numerator = fft_signal(input_freq).^2;
    denominator = sum(fft_signal.^2)-(numerator);

    %SINAD CALCULATION
    Sinad_ratio = (10*log(numerator/denominator))/2;

        end
```

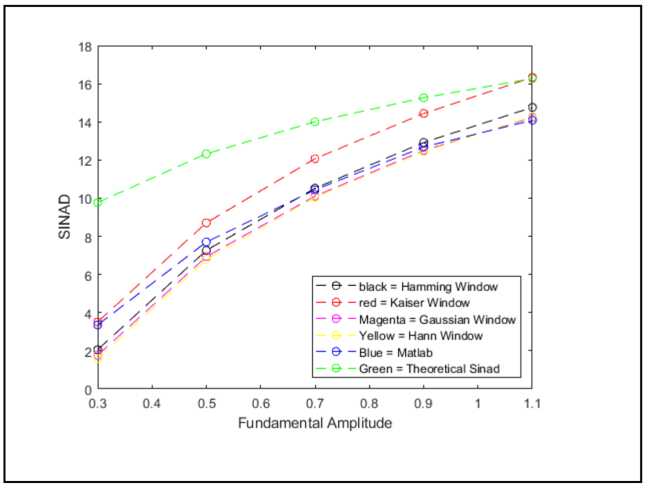
3) *Extended Bundled Version*



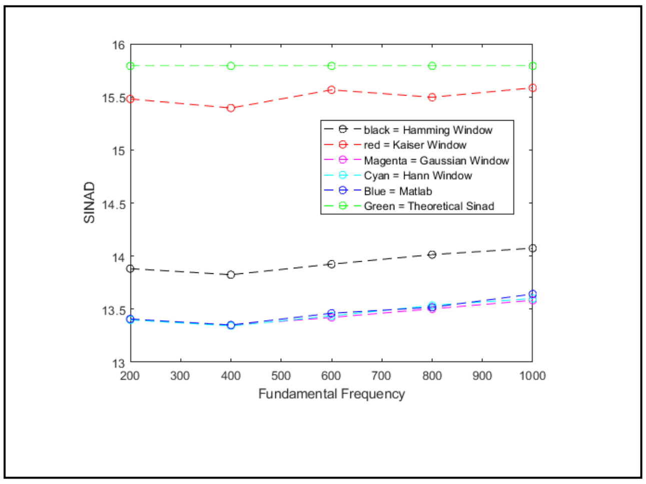
```
function [Sinad_ratio] = ExtendedBundled(signalData,SamplingRate,Windowing,input_freq)

%CHECKNIG INPUT ARGUMENTS AND RUNNING LOOP FOR BUNDLE SINAD
 tf = isa(input_freq,'double');
 if nargin == 4
    if tf == 1
         Sinad_ratio = 0;
         for i = 1:(length(input_freq))
            Sinad_ratio = Sinad_ratio + SINAD.Extended(signalData,SamplingRate,Windowing,input_freq(i))
         end
    else
         error('Improper type Input_FREQ')
    end
 else
     error('Not enough Input Arguments')
 end
end
```

G. *Function Testing*

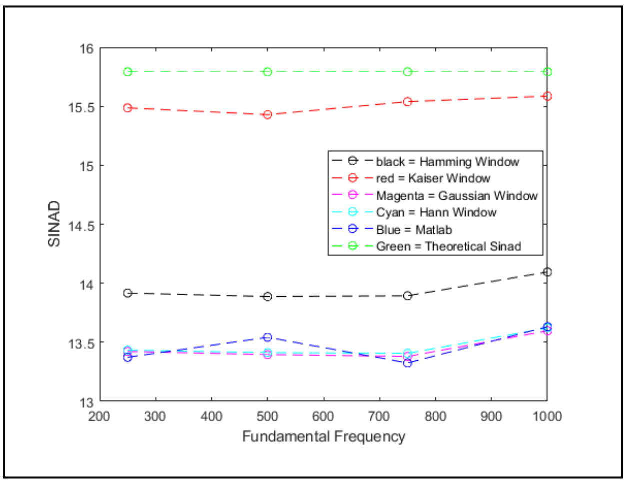
1) *Same Frequency different Amplitude*



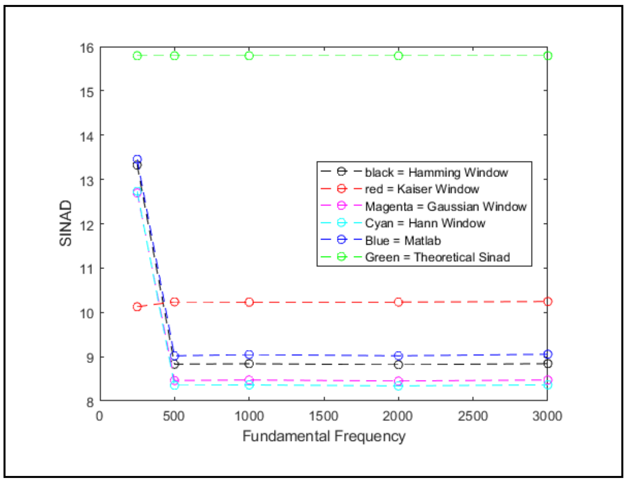
2) *Same Amplitude different Frequency*



3) *Coherent Data*

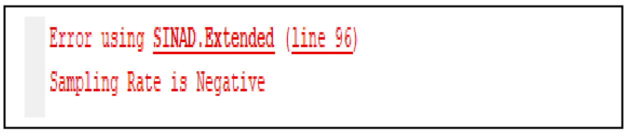


4) *Non-Coherent Data*



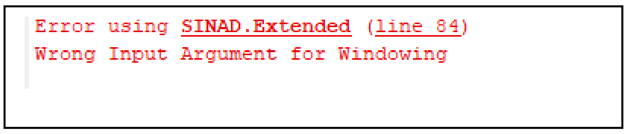
5) *Errors*

a) *Input Sampling Rate:*



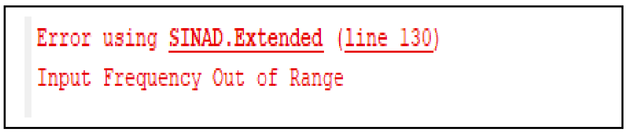
```
Error using SINAD.Extended (line 96)
Sampling Rate is Negative
```

b) *Giving Improper Windowing Options*

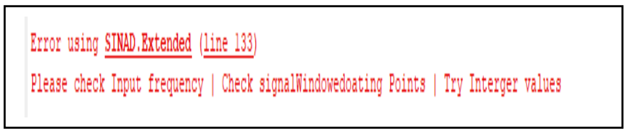


```
Error using SINAD.Extended (line 84)
Wrong Input Argument for Windowing
```

c) Giving Improper Input frequency



```
Error using SINAD.Extended (line 130)
Input Frequency Out of Range
```



```
Error using SINAD.Extended (line 133)
Please check Input frequency | Check signalWindowedoating Points | Try Interger values
```

H. Manual To Use Project

The following section contains snapshots and documentation for using the function in MatLab or externally

1) By using function from script

Put the script at same location as function files and use the class function in the script

2) From command window

After adding or defining signal data in the workspace as variable following functions can be used from command line window

V. Project Version 2 Details

Following section provides in depth details regarding the project design, implementation, and manual

A. Version 2 Aim:

Implementation of Standalone application which can be used for calculation of Extended SINAD and Extended Bundled SINAD.

B. Technology Used:

MatLab R2019b

C. Project Architecture:

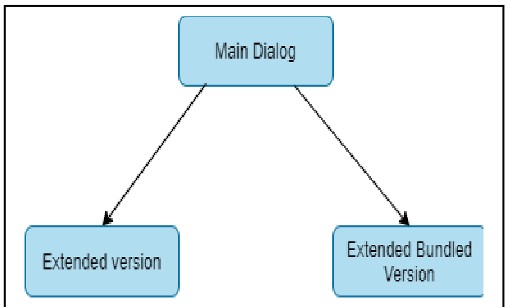


Figure 12 Version 2 Architecture

D. Application GUI SnapShot:

1) Main Dialog:

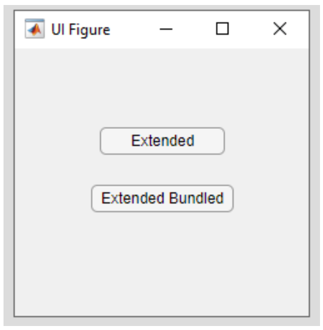
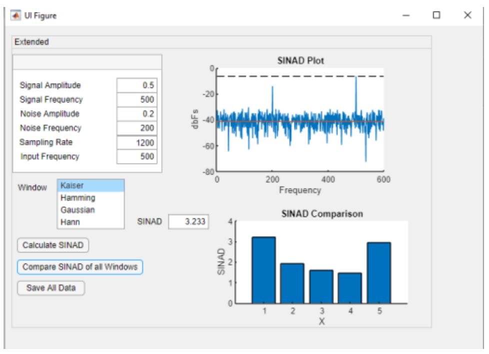


Figure 13 App Main Dialog

2) Extended Dialog:



UI Figure

Extended

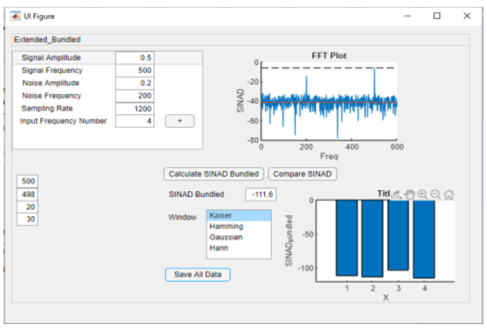
Signal Amplitude	0.5
Signal Frequency	500
Noise Amplitude	0.2
Noise Frequency	200
Sampling Rate	1200
Input Frequency	500

Window: Kaiser (selected), Hamming, Gaussian, Hann

SINAD 3.233

Figure 14 Extended Window Dialog

3) *Extended Bundled Dialog*



UI Figure

Extended_Bundled

Signal Amplitude	0.5
Signal Frequency	500
Noise Amplitude	0.2
Noise Frequency	200
Sampling Rate	1200
Input Frequency Number	4

500
498
20
30

SINAD Bundled -111.6

Window: [✓ Kaiser] Hamming Gaussian Hann

Figure 15 Extended Bundled Dialog

E. *Plot Details:*

a) *Decibel relative to Full scale*

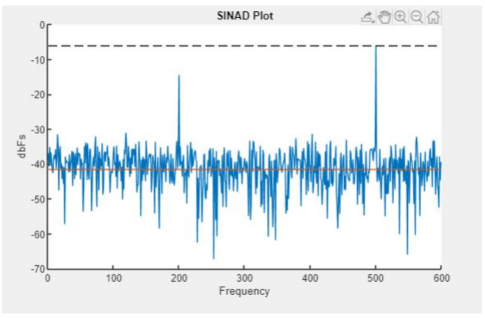


Figure 16 dbFs plot

$$\text{In Fig } 16$$
$$x - axis \ = \ Frequency \mid y - axis \ = \ dbFs$$
$$Blacked\ Dashed - - = Max\ Freq\_Amplitude$$
$$Red\ Solid\ Line = Noise\ Floor$$

b) *Bar Graph:*

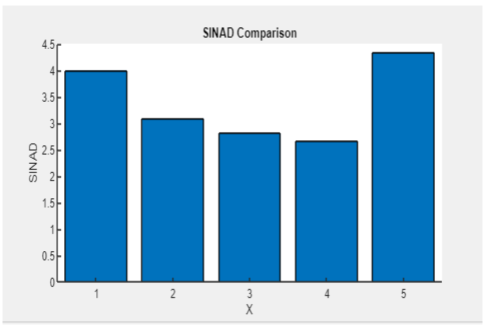


Figure 17 Bar Graph plot

$$\text{In Fig } 17$$
$X\ axis$
$1. Kaiser\ \ 2. Hamming\ 3. Gaussian\ 4. Hann$
$5. Inbuilt\ Matlab$
$Y\ axis = calculated\ SINAD$

F. *Features:*

1) *Extended Dialog:*

a) *User defined Signal :*

In Extended Dialog User can define the input amplitude and frequency of the signal with noise frequency and amplitude In addition random noise is also added to signal
Sampling rate must also be defined for the signal which should be Nyquist frequency or more for getting proper result.

b) *Window selection:*

In Window List box four options are present from which User can select the window for the Signal-to-noise distortion ratio. Options available are 1.Kaiser 2.Hamming 3.Gaussian 4.Hann

c) *SINAD Plot:*

SINAD plot have frequency available on x-axis and SINAD dbFs on Y-axis. Noise Floor and Maximum amplitude are also shown in the plot for better understanding.

d) *SINAD comparison plot:*

From advance analysis perspective SINAD for all windows and MatLab is shown with bar graph which helps to recognize efficiency of all windows and MatLab inbuilt function.

e) *SAVE all data:*

User can also save SignalData, InputFrequency, SINAD_data in text file for further usage.
User can also save plots in png or other format by clicking save icon on plot.

File Names:
Extended_SINAD_Signal_Data
Extended_SINAD_SINAD_Data
Extended_SINAD_InputFrequency_Data

2) *Extended Bundled Dialog:*

In addition to all features available in Extended Dialog , Following Dialog can have multiple Input frequency which ranges from 2[min]-32[max].

File Name when save all data is selected:

Extended_Bundled_SignalData
Extended_Bundled_SinadData
Extended_Bundled_Input_freq

G. Manual to Use Application:

1) *Install the application using the installer.*
2) *Open the application.*
3) *Select the version you want to use.*
4) *Define the input parameters for the signal and noise.*

*random noise will also be added to create signal as real life data

5) *Press calculate SINAD button for the results.*

H. Error:

In case of error windows notification sound will come if sampling rate or input frequency are improper.

References

[1] Kester Walt, “Understand SINAD, ENOB, SNR, THD, THD + N, and SFDR so,” 2009.

[2] [Online]. Available: https://www.electronics-notes.com/articles/radio/radio-receiver-sensitivity/what-is-sinad-signal-to-noise-and-distortion.php. [Accessed 10 06 2020].

[3] "Understanding Noise in the Signal Chain," [Online]. Available: http://www.intersil.com.

[4] K. H. Lundberg, "Analog-to-Digital Converter Testing".

[5] J. G. a. D. G. M. Proakis, Digital Signal Processing: Principles, Algorithms, and Applications, Upper Saddle River, NJ:, 1996.

[6] "FFT Spectral Analysis," DEWEsoft, [Online]. Available: https://training.dewesoft.com/online/course/fft-spectral-analysis.

[7] Y. B. D. H. a. P. M. David Slepiˇcka, "Frequency Estimation to Linearize Time-Domain," *IEEE.*

[8] A. J. Aude, "Audio Quality Measurement Primer," 1988.

[9] C. R. J. a. W. A. S. Johnson, "Telecommunications Breakdown: Concepts of," 2004.

[10] "Windows," Mathworks, [Online]. Available: https://www.mathworks.com/help/signal/ug/windows.html.