ISSN: 0975-766X
CODEN: IJPTFI
Available Online through
www.ijptonline.com
Research Article

PERFORMANCE EVOLUTION OF BASIC SIGNALS USING MATLAB AND LAB VIEW

A.Mohammed Irfan[1], M.Reji[2]
Student, Department of Electronics and Communication Engineering, Saveetha School of Engineering [1]
Assistant professor, Department of Electronics and Communication Engineering, Saveetha School of Engineering[2]
Gmail: mohfan1997@gmail.com

Received on: 25.09.2016 *Accepted on: 15.10.2016*

Abstract:

Nowadays engineers, scientists and researchers use propelled programming suites for their work in the field of sign investigation and reproduction of element frameworks. Among the most generally utilized instruments are Labview and MATLAB. Subsequently an inquiry emerges – which one to pick? In this paper both devices are quickly exhibited, followed by an examination focused on practical examples. Featuring four comparisons – generation of basic signals, convolution, Butterworth filter and noise removal.

Introduction:

Labview and MATLAB were produced by two diverse American organizations. The first is created by National Instruments and the second by Mathworks. programing environment focused around graphical programming language. MATLAB (Matrix Laboratory) is the name utilized for the programming language and programing environment particular for numerical estimations. Both stages help designers and researchers around the globe in different phases of configuration, demonstrating, reproduction, model testing, or organization of new technologies. Researchers predominantly utilize MATLAB for their simulations, particularly because of a lot of extra libraries furthermore the Simulink add-on. Libraries contain particular higher-level capacities of a specific field. Such capacities accelerate the improvement of cutting edge applications. When a utilitarian and natural graphical client interface or association with hardware (signal acquisition and era) is needed rather than MATLAB, Labview is basically utilized. Some more experienced clients likewise pick a mix of both instruments, which impart by means of either the API or DLL libraries. Information can be sent out from one and transported in into an alternate project for further processing. There is additionally a probability to implement MATLAB code in LabviewThe majority of MATLAB capacities can be coordinated into Labview by utilizing Mathscriptfunction module (accessible with »mathscript RT Module« add-on).

Lab View: The beginnings of Labview back to the mid-eighties when Macintosh Company created first machine with graphical client interface. Graphical client interface empowered visualization of stream outlines (flow chart) on the machine screen, which motivated Jeff Kodosky for graphical programming. Since he was utilizing generally data acquisition for his work, he started making a graphical programming language focused around dataflow instead of successive handling, which is most regular for John von Neumann's computer architecture. In 1986 the first form of Labview programing environment was issued, which officially offered programming by uniting the functional blocks, as we know it today. The primary destination of the programming environment was to improve information securing from GPIB transport. Consequently, the first capacity squares empowered information securing, which is still a principle capacity of Labview today. Today, Labview programming environment is the pioneer in the field of machine based estimation and information securing. It characteristics remarkable similarity with National Instruments fittings furthermore different gadgets. In addition it offers easy-to-use development of graphical client interface. There are likewise a lot of add-ons which can be utilized to convey code to standalone gadgets, dissect and procedure signs, control, reproduction, framework examination, report creation and database connection.

started to create MATLAB in late seventies with a specific end goal to rearrange utilization of libraries for numerical calculation (Linpack, EISPACK), which were modified utilizing FORTRAN programming language. The primary favorable circumstances over the use of libraries inside FORTRAN were basic input and yield of information and arithmetic operations on vectors and frameworks all in all. The capability of MATLAB was later found by Jack Little, who established the Mathworkscompany in 1984 alongside Cleve Moler. The organization initially made an interpretation of MATLAB from FORTRAN to C, upgraded the client interface, included M-files and libraries.

Nowadays MATLAB is broad over all scientific disciplines primarily because of a basic linguistic structure, solidness and an extensive variety of applications. There is likewise a great deal of extra libraries and add-ons from different ranges, for example, control and distinguishing proof of frameworks, neural systems, fuzzy logic, measurements, typical math… The most imperative add-on for architects is unquestionably Simulink, which permits modeling, recreation and examination of dynamic systems.

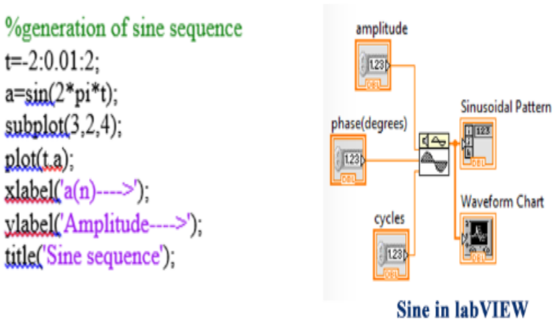
Comparison of Lab View and MATLAB: Below, LabVIEW and MATLAB/Simulink are compared in different areas: generation of basic signals, convolution, Butterworth filter and noise removal.

Each comparison area will emphasize examination of how basic and survey able the project code is what's more the time expected to execute the system will likewise be measured. The straightforwardness of the code will be evaluated by number of components or lines utilized. The speedier the code can be redesigned or debugged the better survey able the project is. Time estimations will be completed in ten reiterations, and after that most extreme, least and normal qualities will be calculated.

Generation of Basic Signals:

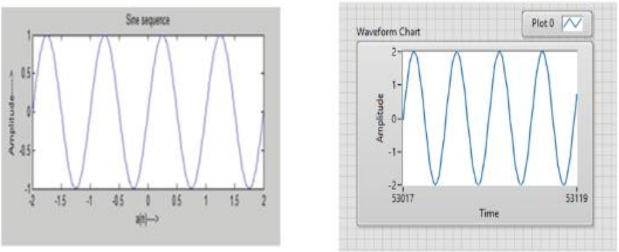
Generation of Sine Sequence:

A simple algorithm was selected to test the computational speed. The program code for generation of sine sequence and the resultant waveforms in MATLAB and LABVIEW are given in fig (1a) and fig (1b) respectively, as follows:



```
%generation of sine sequence
t=-2:0.01:2;
a=sin(2*pi*t);
subplot(3,2,4);
plot(t,a);
xlabel('a(n)---->');
ylabel('Amplitude---->');
title('Sine sequence');
```

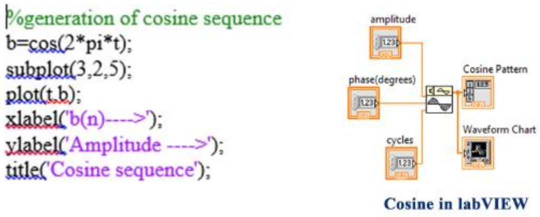
Fig(1a).code for sine sequence in MATLAB and LABVIEW.



Fig(1b). Resultant waveform.

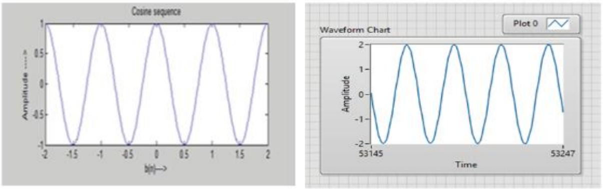
Generation of Cosine Sequence:

The code for generation of cosine sequence and the resultant waveforms are given in fig(2a) and fig (2b) respectively, as given below:



```
%generation of cosine sequence
b=cos(2*pi*t);
subplot(3,2,5);
plot(t,b);
xlabel('b(n)---->');
ylabel('Amplitude ---->');
title('Cosine sequence');
```

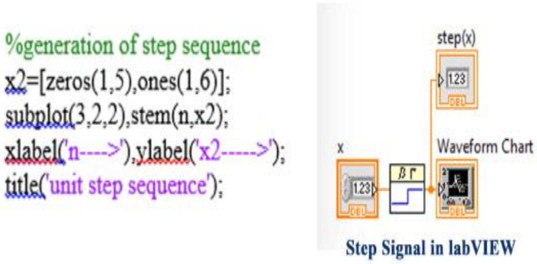
fig(2a).code for cos sequence in MATLAB and LABVIEW.



Fig(2b). Resultant waveform.

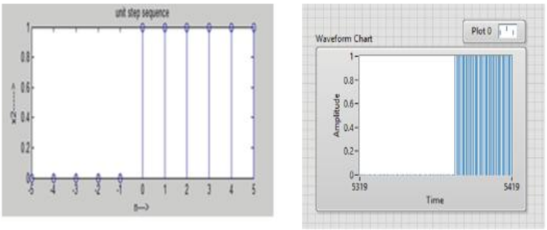
Generation of Step Sequence:

The MATLAB and LABVIEW codes for the generation of step sequence is given in fig(3a) and their respective resultant signal is given in fig(3b).



```
%generation of step sequence
x2=[zeros(1,5),ones(1,6)];
subplot(3,2,2),stem(n,x2);
xlabel('n---->'),ylabel('x2----->');
title('unit step sequence');
```

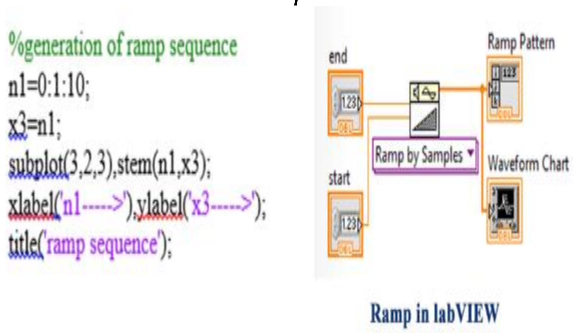
Fig(3a).code for step sequence in MATLAB and LABVIEW.



Fig(3b). Resultant waveform.

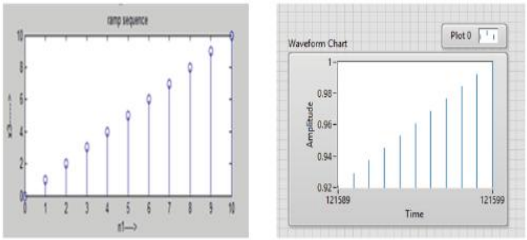
Generation of Ramp Sequence:

The MATLAB and LABVIEW codes for the generation of step sequence is given in fig(4a) and their respective resultant signal is given in fig(4b).



```
%generation of ramp sequence
n1=0:1:10;
x3=n1;
subplot(3,2,3),stem(n1,x3);
xlabel('n1----->'),ylabel('x3----->');
title('ramp sequence');
```

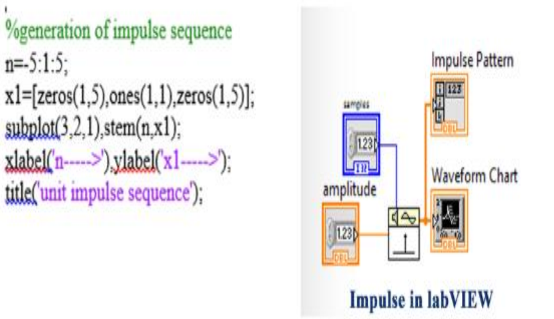
Fig(4a).code for ramp sequence in MATLAB and LABVIEW.



Fig(4b). Resultant waveform

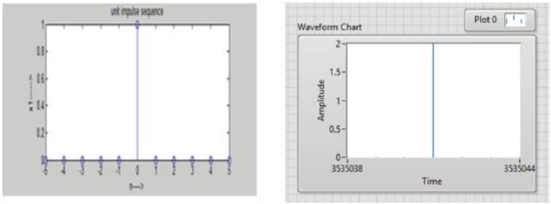
Generation of Impulse Sequence:

The code for generation of impulse sequence and the resultant waveforms are given in fig(5a) and fig (5b) respectively.



```
%generation of impulse sequence
n=-5:1:5;
x1=[zeros(1,5),ones(1,1),zeros(1,5)];
subplot(3,2,1),stem(n,x1);
xlabel('n----->'),ylabel('x1----->');
title('unit impulse sequence');
```

Fig(5a).code for impulse sequence in MATLAB and LABVIEW.



Fig(5b). Resultant waveform.

The results of the comparison of calculation with basic signals are shown in Table 1. The results show that LabVIEW is much faster than

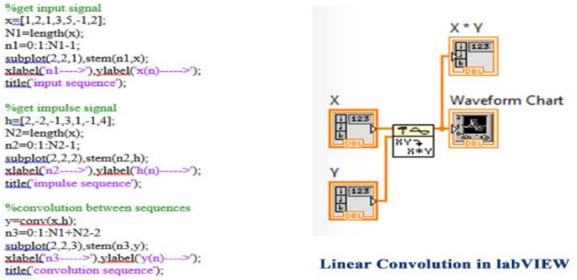
MATLAB. Moreover code in LabVIEW looks very simple and clear. For generation of signals LabVIEW is the way to go.

Table 1. Comparison of calculation with basic signals.

Programming environment	Execution time(sec)		
	Average	Minimum	Maximum
MATLAB	1.158928	1.137303	1.174788
LABVIEW	0.58651	0.48325	0.72964

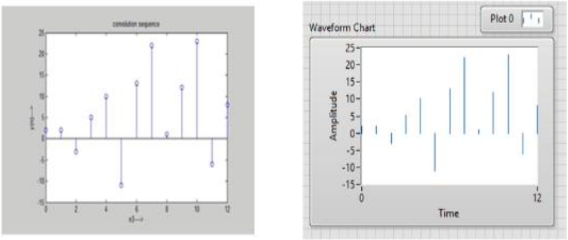
Computation of Linear Convolution:

A simple algorithm was selected to test the computational speed as in the case of generation of basic signal. The program code for computation of Linear Convolution and the resultant waveforms in MATLAB and LABVIEW are given in fig(6a) and fig (6b) respectively.



```
%get input signal
x=[1,2,1,3,5,-1,2];
N1=length(x);
n1=0:1:N1-1;
subplot(2,2,1),stem(n1,x);
xlabel('n1---->'),ylabel('x(n)----->');
title('input sequence');

%get impulse signal
h=[2,-2,-1,3,1,-1,4];
N2=length(x);
n2=0:1:N2-1;
subplot(2,2,2),stem(n2,h);
xlabel('n2---->'),ylabel('h(n)----->');
title('impulse sequence');

%convolution between sequences
y=conv(x,h);
n3=0:1:N1+N2-2
subplot(2,2,3),stem(n3,y);
xlabel('n3----->'),ylabel('y(n)---->');
title('convolution sequence');
```

Fig(6a).code for linear convolution in MATLAB and LABVIEW.



Fig(6b). Resultant waveform.

The results of the comparison of calculation with linear convolution are shown in Table 2. The results show that LabVIEW is as twice as faster than MATLAB. Thus for computation of linear convolution LabVIEW is more convenient.

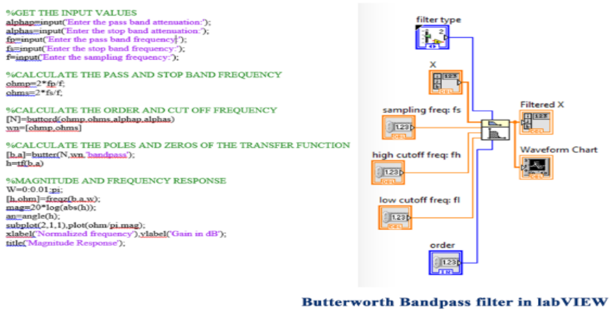
Table 2. Comparison of computation of linear convolution.

Programming	Execution time(sec)

environment	Average	Minimum	Maximum
MATLAB	0.27427	0.269607	0.28303
LABVIEW	0.1951	0.0936	0.1564

Design of Butterworth Bandpass Filter:

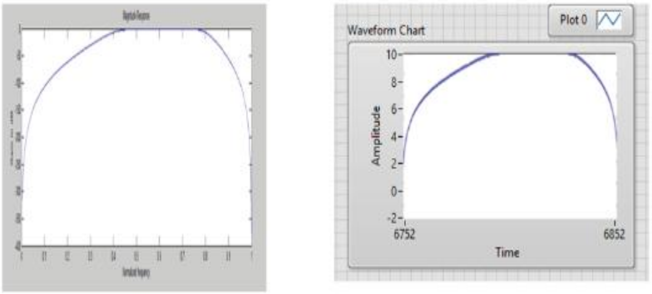
The MATLAB and LABVIEW codes for the design of Butterworth Bandpass filter is given in fig(7a) and their respective resultant signal is given in fig(7b).



```
%GET THE INPUT VALUES
alphap=input('Enter the pass band attenuation:');
alphas=input('Enter the stop band attenuation:');
fp=input('Enter the pass band frequency:');
fs=input('Enter the stop band frequency:');
f=input('Enter the sampling frequency:');

%CALCULATE THE PASS AND STOP BAND FREQUENCY
ohmp=2*fp/f;
ohms=2*fs/f;

%CALCULATE THE ORDER AND CUT OFF FREQUENCY
[N]=buttord(ohmp,ohms,alphap,alphas)
wn=[ohmp,ohms]

%CALCULATE THE POLES AND ZEROS OF THE TRANSFER FUNCTION
[b,a]=butter(N,wn,'bandpass');
h=tf(b,a)

%MAGNITUDE AND FREQUENCY RESPONSE
W=0:0.01:pi;
[h,ohm]=freqz(b,a,w);
mag=20*log(abs(h));
an=angle(h);
subplot(2,1,1),plot(ohm/pi,mag);
xlabel('Normalized frequency'),ylabel('Gain in dB');
title('Magnitude Response');
```

Butterworth Bandpass filter in labVIEW

Fig(7a).code for design of butterworth bandpass filter in MATLAB and LABVIEW.



Fig(6b). Resultant waveform.

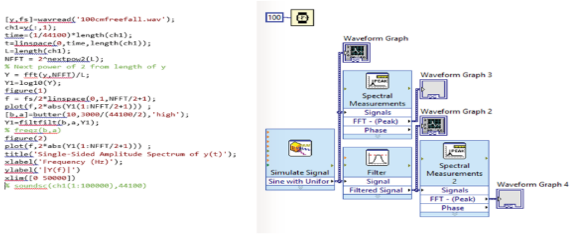
The results of the comparison of calculation with Butterworth Bandpass filter are shown in Table 3. The results show that LabVIEW is faster than MATLAB. Thus for computation of Butterworth Bandpass filter LabVIEW is more convenient and over more, the filter types can be easily changed in LabVIEW.

Table 3. Comparison of calculation of Butterworth Bandpass filter.

Programming Environment	Execution time(sec)		
	Average	Minimum	Maximum
MATLAB	7.819405	5.997056	9.494876
LABVIEW	0.62742	0.560792	0.83302

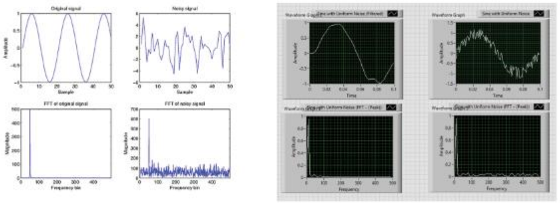
Remove noise from Audio signal:

The program code for removing noise from the audio signal and the resultant waveforms in MATLAB and LABVIEW are given in fig(1a) and fig (1b) respectively, as follows:



```
[y,fs]=wavread('100cmfreefall.wav');
ch1=y(:,1);
time=(1/44100)*length(ch1);
t=linspace(0,time,length(ch1));
L=length(ch1);
NFFT = 2^nextpow2(L);
% Next power of 2 from length of y
Y = fft(y,NFFT)/L;
Y1=log10(Y);
figure(1)
f = fs/2*linspace(0,1,NFFT/2+1);
plot(f,2*abs(Y1(1:NFFT/2+1))) ;
[b,a]=butter(10,3000/(44100/2),'high');
Y1=filtfilt(b,a,Y1);
% freqz(b,a)
figure(2)
plot(f,2*abs(Y1(1:NFFT/2+1))) ;
title('Single-Sided Amplitude Spectrum of y(t)');
xlabel('Frequency (Hz)');
ylabel('|Y(f)|')
xlim([0 50000])
% soundsc(ch1(1:100000),44100)
```

Fig(7a).code for noise removal in audio signal in MATLAB and LABVIEW.



Fig(7b). Resultant waveform

The results of the comparison of calculation with noise removal in audio signal are shown in Table 4.

Table 4. Comparison of noise removal from audio signal in MATLAB and LABVIEW.

Programming environment	Execution time(sec)		
	Average	Minimum	Maximum
MATLAB	3.654285	3.356982	3.801656
LABVIEW	0.87515	0.84561	1.09532

Conclusion:

Thus, we describe that while the executing an audio signal in that the noise removals from a circuit in MATLAB will it takes much time comparing LabVIEW. The execution time is reduced half of the time while the execution takes place in MATLAB. For efficient computation and algorithm development MATLAB is much faster than LabVIEW. Similarly for Signal acquisition and Signal processing LabVIEW is faster.

References:

1. “Comparison of labview and matlab for scientific research”, hawe hidravlika d.o.o., petrovče 225, petrovče, Slovenia.

2. Gross, B., Kozek, M., Jörgl, H.: Identification and Inversion of Magnetic Hysteresis Using LabVIEW and MATLAB, International Journal of Online Engineering iJOE, 1 (2005).
3. Xiong, Y., Qin, B., Wu, M., Yang, J., Fan, M.: LabVIEW and MATLAB-Based Virtual Control System for Virtual Prototyping of Cyclotron, Proceedings of PAC07, New Mexico, USA, 2007, str. 281-283.
4. Coito, F., Almeida, P., Palma, L.: SMCRVI-A LabVIEW/MATLAB Based Tool for Remote Monitoring and Control, 10th IEEE Conference on Emerging Technologies and Factory Automation, Catania, 2005, str. 1039-1044.
5. Tekin, R.: MATLAB and LabVIEW in Modeling, Analysis and Real Time Control of a Motion Control System, 8th IEEE International Conference on Control and Automation, Xiamen, China, 2010, str. 2077-2081.
6. Kodosky, J.: The History of LabVIEW, http://zone.ni.com/wv/app/doc/p/id/wv-2868/upvisited/y, 2011-11-17.
7. Grossman, L.: LabVIEW 1.0, it was cool to see you,http://www.dmcinfo.com/Portals/0/Blog%20Pictures/LabVIEWv1small.jpg, 2011-11-17.
8. Moler, C.: The Origins of MATLAB, http://www.mathworks.com/company/newsletters/news_notes/clevescorner/dec04.html, 2011-11-17.
9. Tašner, T., Lovrec, D.: Napredna raba LabVIEW za sisteme nadzora in regulacij v hidravliki, Fluidna tehnika 2011, Maribor, 2011, str. 299-310.