International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 01 | Jan 2021 www.irjet.net p-ISSN: 2395-0072

Development of Digital Pulse Emulator

Muskaan Gupta[1], Simran Koul[2], Ayush Suri[3], Vishal Kothari[4]

[1-4]*Student, Dept. of Computer Engineering, K J Somaiya College Of Engineering, Mumbai, India.*

Abstract - *Large scale data acquisition systems are used in big data experiments currently being carried out in different International institutes. Testing of these systems is a challenging task wherein each analog input channel chain has to be tested for the integrity of the digitized input data along with ensuring the robustness/correctness of the signal processing algorithms. A digital emulator is an important philosophy which can be utilized in the testing of such systems. In a digital emulator, pulses are generated depending upon the characteristics of the sensor detectors and the process information. This project would involve the development of routines required for the generation of digitized pulses given the information pertaining to the process that needs to be probed and analyzed. Thus it is proposed to generate pulses with the help of algorithms based on probability and amplitude spectrum. These pulses could, in turn, be used for testing data acquisition systems and associated signal processing techniques. The entire project would involve the development of associated routines/algorithms using languages like Python and C.*

Key Words: **High-speed data acquisition system, Pulse generator, Exponential Rise and Decay, Gaussian distribution, Event Capturing based on amplitude-time spectrum, Real time Data Emulation, Digital Signal Processing**

1. Introduction

High-speed data acquisition systems are used for analyzing and processing large amounts of data. The signals which are analyzed or recorded using high-speed data acquisition systems may be pulsed or continuous. Testing of high-speed data acquisition systems involves high risk and hence digital pulse emulators can be used for testing these systems. The purpose of the proposed system would involve the development of routines required for the generation of digitized pulses given the information pertaining to the process that needs to be probed and analyzed. The system will be used to generate pulses from the given probability and amplitude spectrum. The high-speed multi-channel data acquisition will have a large number of channels which means more inputs can be given to the system. The system will take inputs which are 10 millivolts to 100 millivolts and will handle Giga samples or even mega samples. This project involves the development of software to generate pulses which in turn can be used to test data acquisition systems. In order to test 20 thousand channels or 1 million channels, we need many such types of equipment as it cannot be tested using a single equipment. For nuclear data acquisition, we require radioactive resources to test the system. These radioactive sources are random in nature i.e., random peak to peak value, random arrival, random rise time and random fall time. If we are able to successfully generate such kinds of pulses, it will be easy to test large scale data acquisition systems. We have arbitrary waveform generators in the market but there are many constraints which involve little or no randomization, nonexistent pile-up phenomenon and certain memory constraints. Pulse pile-up is basically when two or more closely generating pulses combine together to form a single distorted pulse. The project focuses on writing an algorithm to generate random pulses. The random pulses will have a random time of arrival, random peak to peak amplitude and random rise time and fall time which are hardly possible in arbitrary waveform generators. With the development of this project, we will be minimizing the need for radioactive sources for testing large scale data acquisition systems.

2. Overview

Essentially, our project involves the development of routines required for the generation of digitized pulses. These pulses are generated by fetching values from the amplitude spectrum as well as probability spectrum.

In order to test the validity of the simulations and theoretical work relating to the shaping, acquisition, and processing of spectrometric signals, a test system is needed. Our test system can be considered as an electronic software system which has features offering the functionality of - 1) Random event generation 2) Visualizing the pulses 3) User interface to alter the event distribution

Each individual event has a rising peak and falling edge which follows an exponential rise and decay functions.

For real-time emulation of high counting rates, it is necessary to speed up the whole test System.

The experimenter needs all of the advantages of having a general-purpose instrument, so the generator requirements were as follows:

1.) The ability for the generation of pulses with programmable amplitude and programmable time intervals between the events in real-time:- Depending on the particular application, the two parameters can be periodic or random based on the distribution parameters. The next phase for the project will include pile-up phenomenon in order to emulate two or more pulses with different amplitudes in any given point of time, also the

fault tolerance is to be measured for Emulation with best resolution of pulses.

2.) A capability for ensuring repeatability of the test waveforms: - This implies that all the necessary random parameters must be selected in the course of the project by using pseudorandom generators.

3.) Wide counting rate dynamics:- In order to demonstrate the effectiveness of techniques to correct for count rate dependent distortion, it is necessary to have a low counting rate reference spectrum and a high counting rate spectrum that are identical in all respects.

The generator must be so designed that it can be used in a number of different ways. For instance:

1) Its output can be used as the test signal when testing a charge sensitive preamplifier and shaping amplifier system (analog shaper);

2) Its output can be used as an input to real-time digital pulse shapers;

3) It can be used to emulate digitized signals coming from both simple charges sensitive preamplifiers and more refined shapers in an all-digital spectroscopy system, or signals coming directly from a scintillator.

Usage of libraries enhances the speed of development process example data visualization libraries like oscilloscope, matplotlib, etc. With the help of features from the visualization libraries easily integratable software for simulation can be produced. Also, working on electronic test instruments like DSO develops a better understanding of the project flow.

3. DSP algorithms

3.1 Guassian Distribution

In probability theory, the central limit theorem establishes that, in some situations, when independent random variables are added, their properly normalized sum tends toward a normal distribution even if the original variables themselves are not normally distributed [1].

Similarly in statistics, the importance of Gaussian distribution comes from the central limit theorem. This, in essence, says, if you sum up a lot of independent distributions, the eventual distribution is a Gaussian[2].

Also in probability theory, a normal (or Gaussian or Gauss or Laplace–Gauss) distribution is a type of continuous probability distribution for a real-valued random variable. The general form of its probability density function is

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

The parameter mu is the mean or expectation of the distribution (and also it's median and mode), and sigma is its standard deviation. The variance of the distribution is σ^2 where σ = sigma.

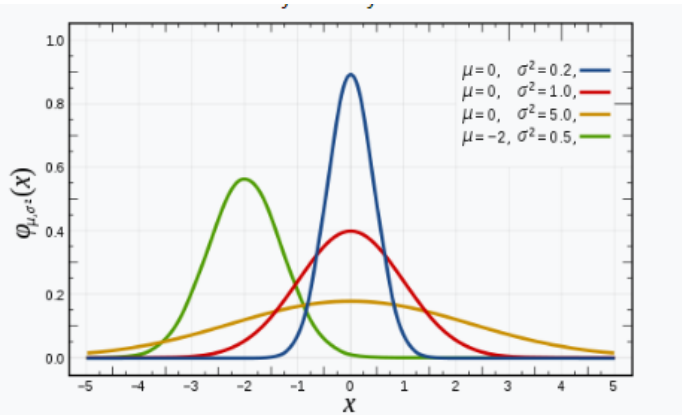

Figure 1. Variance of distribution

3.2 Exponential rise and fall

A random variable with a Gaussian distribution is said to be normally distributed and is called a normal deviate.[4]

In our system, we need to model this behaviour as function. We generate different digital signal algorithms to do so. In mathematical modelling, we choose a familiar general function with properties that suggest that it will model the real-world phenomenon which we wish to analyze. In the case of rapid growth, we have chosen the exponential growth function:

y=A0e^kt

where A0 is equal to the value at time zero, e is Euler's constant, and k is a positive constant that determines the rate (percentage) of growth. We may use the exponential growth function in applications involving doubling time i.e. the time it takes for a quantity to double.

Such phenomena are seen in real life in wildlife populations, financial investments, biological samples, and natural resources where they may exhibit growth based on doubling time.

On the other hand, if a quantity is falling rapidly toward zero, without ever reaching zero, then we should probably choose the exponential decay model. Equation for which is

y=A0e^−kt

where A0 is the starting value, and e is Euler's constant. Now k is a negative constant that determines the rate of decay[3].

We illustrate the phenomenon with the example of an X-ray tube IRF where in a rapid growth and a never ending decay of the event can be observed as in figure 2

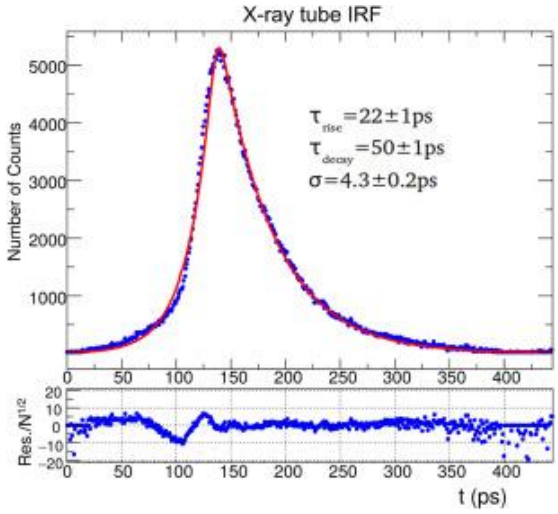

Figure 2. X-ray tube IRF

4. Simulation

4.1 Guassian graph plots

Now the system is designed to generate different waves in a step manner. First we generate Gaussian curve and plot it as shown in figure 3 here we illustrate how we apply this above concept of gaussian distribution which is used in a similar fashion in the event generator module of the system.

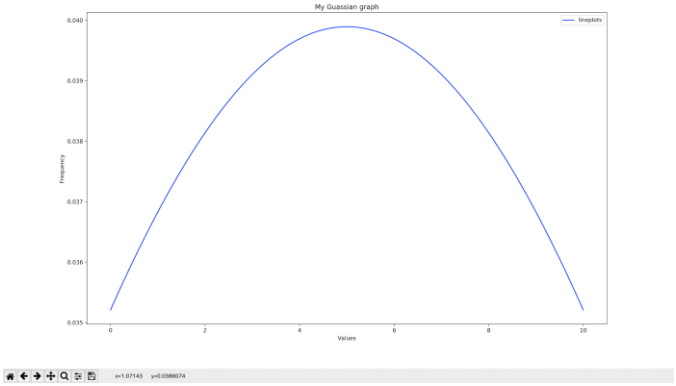

Figure 3 : Gaussian Curve of the System plot

We found out events based on different input parameters like mean and standard deviation which generate a Gaussian curve. An event generator generates events based on the input from the Guassian file. At stage two, a reverse process is applied to recalculate the Gaussian graph. From the event file of the events. We notice that it still retransforms to a gaussian distribution as in the initial plot as seen in figure 4.

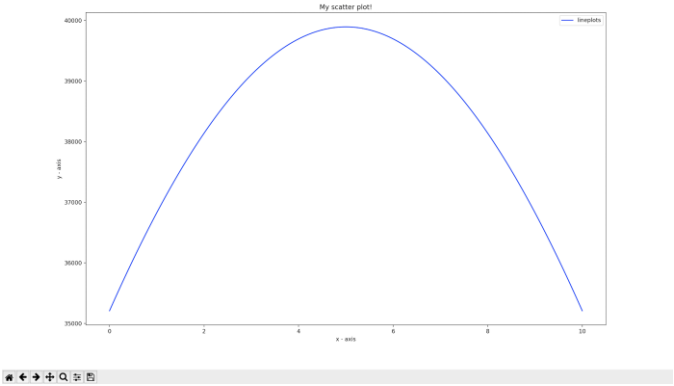

Figure 4: Re-Plot of Events

4.2 Pulse Event graph plots

In stage three, we generate the plot for the events using plotting tools. The figure described how events are increasing in a linear fashion.

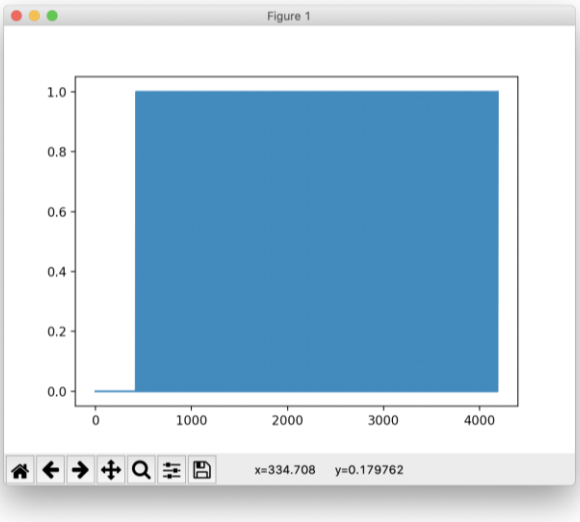

Figure 5: Linear event plot

Then, in stage four, the system uses the random generator to redistribute the events with different time lapse. From this we draw a sawtooth plot as shown in figure 6. We observe that they have steep rise and fall.

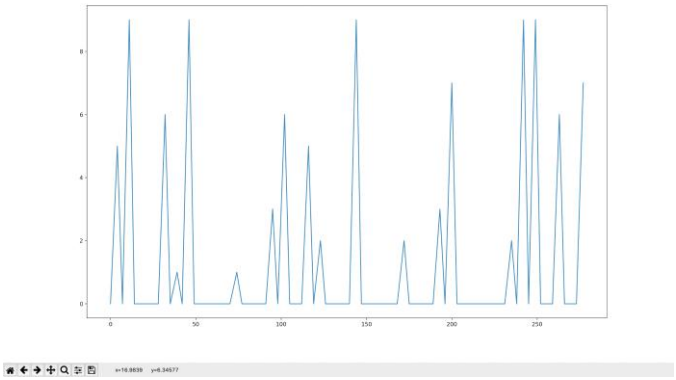

Figure 6: Sawtooth curve of pulses

Finally, as we see in figure 7, we apply the exponential rise and exponential fall to generate a function similar to that in a reactor. Problems like pile up are yet to be addressed.

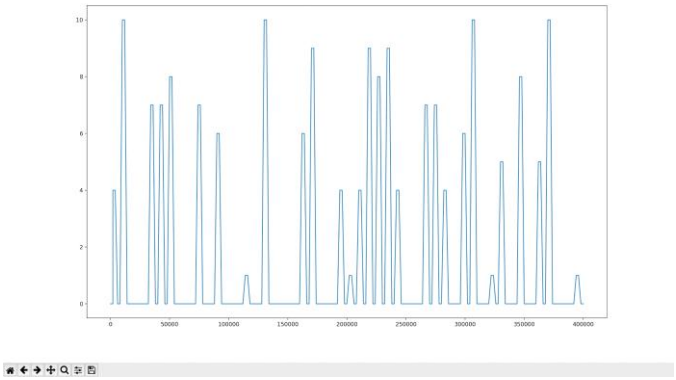

Figure 7 : exponential rise and fall of events

In order to understand how events are generated, one can run the application that our group has created on any UNIX or Linux architecture system with different plots for various purposes. Additionally, a system with nanosecond functionality to requisite data through files obtained from analog to digital converter is collected and visualized with different modes discussed in the paper below.

5. Results and discussion

In this section, we put our inferences after the simulation has been carried out. This will give a holistic view of the performances of each of the modes with regards to the plot.

5.1 Oscilloscope mode and Pyplot mode for DSP :

There are two choices of modes to display the system generated event files. Each one displays output files with dynamic scaling of the amplitude against the increasing time. A sample file of different events was passed to test the performance of these tools which are discussed in the upcoming sections. The figure 8 and 9 below shows random files being displayed in the different modes of the system.

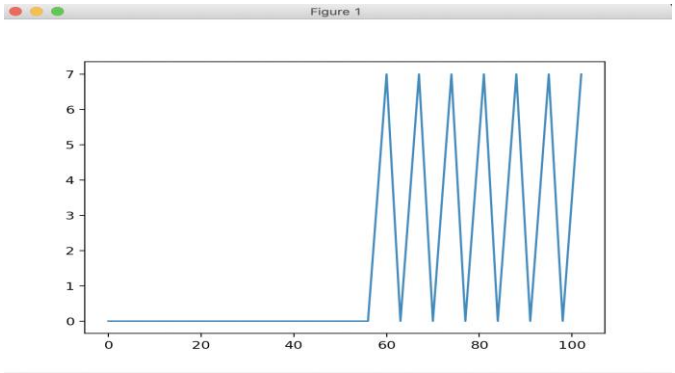

Figure 8. PyPlot Mode Graph

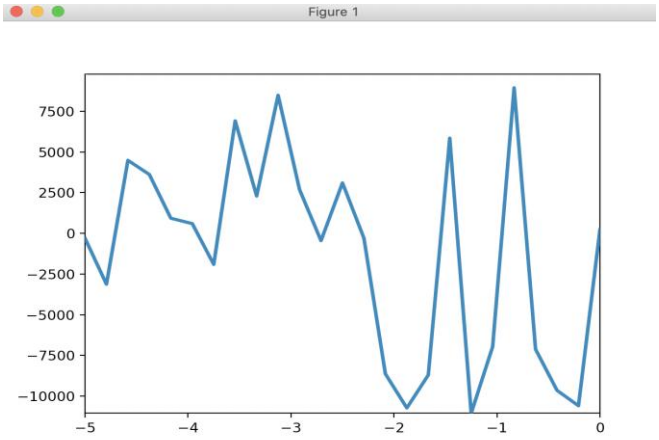

Figure 9. Oscilloscope Mode Based on Random file

5.2 Read/Write In/Out to evaluate performance

Figure 10. PyPlot Mode Image

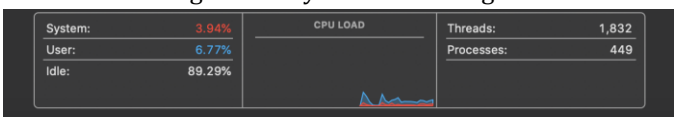

Figure 11. Oscilloscope Mode Image

The idle time in pyplot mode is 94.6% and that in OSC has a value of around 89.2%. Which further brings to the discussion that CPU is extensively utilized more efficiently in the later mode. Also, the number of threads in

the latter is greater by a value of 44 which tells how efficiently the plot is internally synchronized.

5.3 Summarising discussion based on the system process reports

Oscilloscopes can measure the amplitude of the signal, the frequency of a signal and we can also verify if the signal is of the shape that we are expecting. Also, an oscilloscope helps us to recognize the peak to peak amplitude of the signal along with the frequency of the signal.

Additionally, an oscilloscope is perfect for troubleshooting DC power supplies with excessive ripple resulting from component failure.

5.4 Memory size used by the two modes

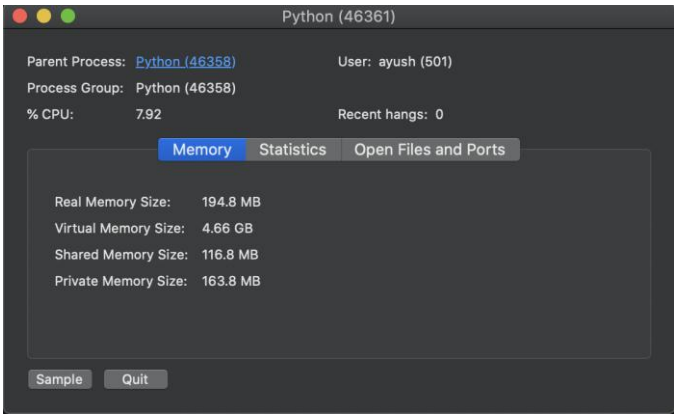

Figure 12. PyPlot Process Memory Summary

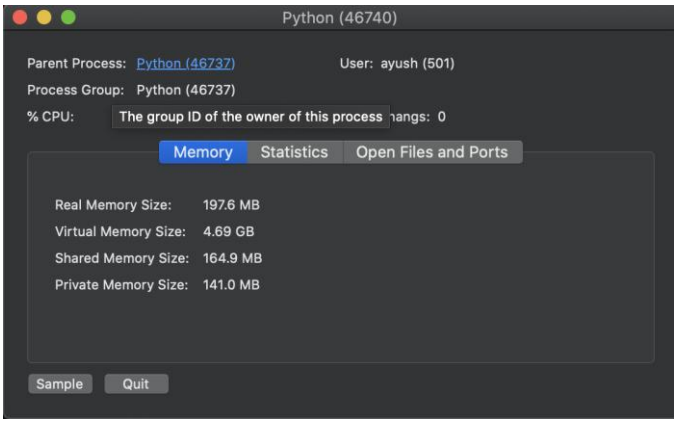

Figure 13. OSC Process Memory Summary

5.5 Statistics of both the modes:

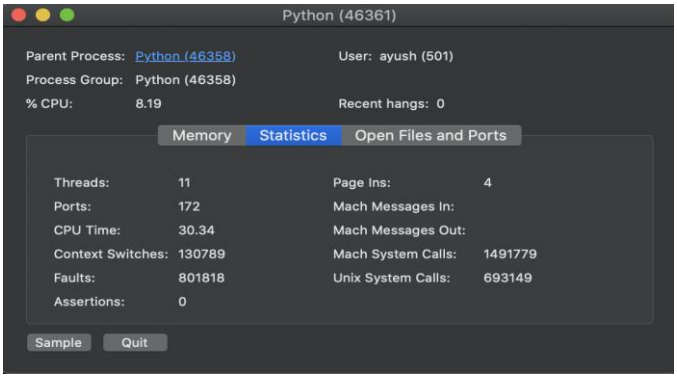

Figure 15. Pyplot Mode Statistics Image

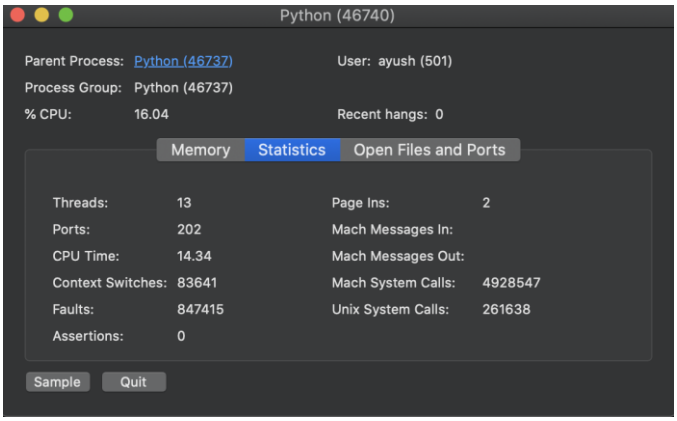

Figure 16. Oscilloscope Mode Statistics Image

So overall based on the processing of the programs one can say that the CPU utilisation is to be kept in mind while running programs also there is a huge amount of parallel processing which makes better at the real-time acquisition of the data.

6. Conclusion

This project involved the development of routines which looked after the amplitude and time spectrum. The optimised subroutines were used for the generation of digitized pulses given the information pertaining to the process that was probed and analysed. These pulses which were generated were in turn used for testing of data acquisition systems and associated signal processing techniques. As these systems acquire data in nanosecond time, we built a wrapper for python which could halt for nanosecond periods. These libraries were created using C and then imported in the Python library. The algorithm was initialized by a reference pulse shape, with the statistical distribution of amplitude and time.

7. Future work

In the foresight, we will build a Desktop Application for our project specifically for the Windows Operating System. A problem called “Pile-up problem” which frequently occurs in randomly generated events will be dealt with by writing appropriate algorithms. We plan to change the way to acquire data where we will be using ADCs in the Zynq hardware using PetaLinux tools.

8. Acknowledgements:

We as part of our final year project at KJSCE developed and tested the software under the guidance of Professor Gopal Sonune. After Analyses of the data we wrote the manuscript. He has read and approved the final manuscript. Also, we would like to thank our external guide Professor Bhakti Paulakar for their support.

REFERENCES

[1] Abraham de Moivre who, in a remarkable article published in 1733 and Sir Francis Galton described the Central Limit Theorem in this way the wikipedia article on Central Limit Theorem states the step-by-step explanation of the classical it

[2] Hazewinkel, Michiel, ed. (2001) [1994], "Normal distribution", Encyclopedia of Mathematics, Springer Science+Business Media B.V. / Kluwer Academic Publishers, ISBN 978-1-55608-010-4

[3] Abramson, Jay et al.. Provided by: OpenStax Under module 12 College Algebra describing Exponential and Logarithmic Equations and Models located at http://cnx.org/

[4] Abba, F. Caponio, F. Guerrieri, A. Geraci, G. Ripamonti, "A Novel algorithm for pulse amplitude modulation for digital emulation of radioactive sourced & quot;, Proc. of the IEEE Nucl. Sci. Symp., Oct. 30 – Nov. 6, 2010, Knoxville, Tennessee, USA.

[5] A. Abba, F. Caponio, A. Geraci, G. Ripamonti, " Design and test equipment of digital processors for output analysis from radiation detectors & quot;, Proc. of 2011 IEEE Nuclear Science Symposium, October 23-29, 2011, Valencia, Spain.

[6] R. Abbiati, A. Geraci, G. Ripamonti, "Self-configuring digital processors for on-line pulse analysis & quot;, IEEE Trans. Nucl. Sci., Vol.51, Issue:3, June 2004, Pages:826-830.

[7] Papoulis, S.U.Pillai, "Probability, random variables and stochastic processes & quot;, McGraw-Hill, 2001.