Published November 29, 2013 by Mathuranathan

Simulation and Analysis of White Noise in Matlab

(**35** votes, average: **4.57** out of 5)

If you are looking for a way to generate AWGN noise in Matlab, go here.

Checkout this ebook : Digital Modulations using Matlab – by Mathuranathan Viswanathan

White Noise Process

A random process (or signal for your visualization) with a constant power spectral density (PSD) function is a white noise process.

Power Spectral Density

Power Spectral Density function shows how much power is contained in each of the spectral component. For example, for a sine wave of fixed frequency, the PSD plot will contain only one spectral component present at the given frequency. PSD is an even function and so the frequency components will be mirrored across the Y-axis when plotted. Thus for a sine wave of fixed frequency, the double sided plot of PSD will have two components – one at +ve frequency and another at –ve frequency of the sine wave. (How to plot PSD/FFT in Matlab)

Gaussian and Uniform White

Noise:

A white noise signal (process) is constituted by a set of independent and identically distributed (i.i.d) random variables. In discrete sense, the white noise signal constitutes a series of samples that are independent and generated from the same probability distribution. For example, you can generate a white noise signal using a random number generator in which all the samples follow a given Gaussian distribution. This is called White Gaussian Noise (WGN) or Gaussian White Noise. Similarly, a white noise signal generated from a Uniform distribution is called Uniform White Noise. White Gaussian Noise and Uniform White Noise are frequently used in system modelling. In modelling/simulation, a white noise can be generated using an appropriate random generator. White Gaussian Noise can be generated using "randn" function in Matlab which generates random numbers that follow a Gaussian distribution. Similarly, "rand" function can be used to generate Uniform White Noise in Matlab that follows a uniform distribution. When the random number generators are used, it generates a series of random numbers from the given distribution. Let's take the example of generating a White Gaussian Noise of length 10 using "randn" function in Matlab – with zero mean and standard deviation=1.

```
>> mu=0;sigma=1;
>> noise= sigma *randn(1,10)+mu
```

noise =   -1.5121   0.7321   -0.1621   0.4651   1.4284   1.0955   -0.5586
1.4362   -0.8026   0.0949

More simulation techniques available in this ebook – Digital Modulations using Matlab – by Mathuranathan Viswanathan

What is i.i.d ?

This simply generates 10 random numbers from the standard normal distribution. As we know that a white process is seen as a random process composing several random variables following the same Probability Distribution Function (PDF). The 10 random numbers above are generated from the same PDF (standard normal distribution). This condition is called "identically distributed" condition. The individual samples given above are “independent” of each other. Furthermore, each sample can be viewed as a realization of one random variable. In effect, we have generated a random process that is composed of realizations of 10 random variables. Thus, the process above is constituted from “independent identically distributed” (i.i.d) random variables.

Strictly and weakly defined White noise:

Since the white noise process is constructed from i.i.d random variable/samples, all the samples follow the same underlying probability distribution function (PDF). Thus, the Joint Probability Distribution function of the process will not change with any shift in time. This is called a stationary process. Thus the white noise is a stationary process. As with a stationary process which can be classified as Strict Sense Stationary (SSS) and Wide Sense Stationary (WSS) processes, we can have white noise that is SSS and white noise that is WSS. Correspondingly they can be called “strictly white noise signal” and “weakly white noise signal”.

What’s with Covariance Function/Matrix ?

A white noise signal, denoted by x(t), is defined in weak sense is a more practical condition. Here, the samples are statistically uncorrelated and identically distributed with some variance equal to σ^2. This condition is specified by using a covariance function as

$$cov(x_i, x_j) = \begin{cases} \sigma^2 & , i = j \\ 0 & , i \neq j \end{cases}$$

Why do we need a covariance function? Because, we are dealing with a random process which composes “n” random variables(10 variables in the modelling example above). Such a process is viewed as multivariate random vector or multivariate random variable. For multivariate random variables, Covariance function specified how each of the “n” variables in the given random process behaves with respect to each other. Covariance function generalizes the notion of variance to multiple dimensions. The above equation when represented in the matrix form gives the covariance matrix of the white noise random process.

$$C_{XX} = \begin{pmatrix} \sigma^2 & \cdots & 0 \\ \vdots & \sigma^2 & \vdots \\ 0 & \cdots & \sigma^2 \end{pmatrix} = \sigma^2 \mathbf{I}$$

Since the random variables in the white noise process are statistically uncorrelated, the covariance function contains values only along the diagonal. The matrix above indicates that only the auto-correlation function exists for each random variable. The cross-correlation values are zero (samples/variables are statistically uncorrelated with respect to each other). The diagonal elements are equal to the variance and all other elements in the matrix are zero.The ensemble auto-correlation function of the weakly defined white noise is given by

$$R_{XX}(\tau) = E[x(t)x^*(t-\tau)] = \sigma^2 \delta(\tau)$$

This indicates that the auto-correlation function of weakly defined white noise process is zero everywhere except at lag (\tau=0).
Related topic: Constructing the auto-correlation matrix in Matlab

Frequency Domain Characteristics:

Wiener-Khintchine Theorem states that for Wide Sense Stationary Process (WSS), the power spectral density function (S_{xx}(f)) of the random process can be obtained by Fourier Transform of auto-correlation function of the random process. In continuous time domain, this is represented as

$$S_{XX}(f) = F[R_{XX}(\tau)] = \int_{-\infty}^{\infty} R_{XX}(\tau) e^{-j2\pi f \tau} \, d\tau$$

For the weakly defined white noise process, we find that the mean is a constant and its covariance does not vary with respect to time. This is a sufficient condition for a WSS process. Thus we can apply Weiner Khintchine Theorem.

Therefore, the power spectral density of the weakly defined white noise process is constant (flat) across the entire frequency spectrum. The value of the constant is equal to the variance or power of the white noise.

Testing the characteristics of White Gaussian Noise in Matlab:

Generate a Gaussian white noise signal of length L=100,000 using the randn function in Matlab and plot it. Here the underlying pdf is a Gaussian pdf with mean μ=0 and standard deviation σ=2. Thus the variance of the Gaussian pdf is σ^2=4. More simulation techniques available in this ebook – Digital Modulations using Matlab – by Mathuranathan Viswanathan

```
clear all; clc; close all;
L=100000; %Sample length for the random signal
mu=0;
sigma=2;
X=sigma*randn(L,1)+mu;

figure();
subplot(2,1,1)
plot(X);
title(['White noise : \mu_x=',num2str(mu),' \sigma^2=',num2str(sigma^2)
xlabel('Samples')
ylabel('Sample Values')
grid on;
```

Plot the histogram of the generated white noise and verify the histogram by plotting against the theoretical pdf of the Gaussian random variable.

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

More simulation techniques available in this ebook – Digital Modulations using Matlab – by Mathuranathan Viswanathan

```
subplot(2,1,2)
n=100; %number of Histrogram bins
[f,x]=hist(X,n);
bar(x,f/trapz(x,f)); hold on;
%Theoretical PDF of Gaussian Random Variable
g=(1/(sqrt(2*pi)*sigma))*exp(-((x-mu).^2)/(2*sigma^2));
plot(x,g);hold off; grid on;
title('Theoretical PDF and Simulated Histogram of White Gaussian Noise'
legend('Histogram','Theoretical PDF');
xlabel('Bins');
ylabel('PDF f_x(x)');
```

Compute the auto-correlation function of the white noise. The computed auto-correlation function has to be scaled properly. If the ‘xcorr’ function (inbuilt in Matlab) is used for computing the auto-correlation function, use the ‘biased’ argument in the function to scale it properly. More simulation techniques available in this ebook – Simulation of Digital Communication systems using Matlab

```
figure();
Rxx=1/L*conv(flipud(X),X);
lags=(-L+1):1:(L-1);

%Alternative method
%[Rxx,lags] =xcorr(X,'biased');
%The argument 'biased' is used for proper scaling by 1/L
%Normalize auto-correlation with sample length for proper scaling

plot(lags,Rxx);
title('Auto-correlation Function of white noise');
xlabel('Lags')
ylabel('Correlation')
grid on;
```

Simulating the PSD of the white noise:

Simulating the Power Spectral Density (PSD) of the white noise is a little tricky business. There are two issues here 1) The generated samples are of finite length. This is synonymous to applying truncating an infinite series of random samples. This implies that the lags are defined over a fixed range. (FFT and spectral leakage – an additional resource on this topic can be found here) 2) The random number generators used in simulations are pseudo-random generators. Due these two reasons, you will not get a flat spectrum of psd when you apply Fourier Transform over the generated auto-correlation values.The wavering effect of the psd can be minimized by generating sufficiently long random signal and averaging the psd over several realizations of the random signal.

Simulating Gaussian White Noise as a Multivariate Gaussian Random Vector:

To verify the power spectral density of the white noise, we will use the approach of envisaging the white noise as a composite of 'N' Gaussian random variables. We want to average the psd over L such realizations. Since there are 'N' Gaussian random variables (N individual samples) per realization, the covariance matrix C_{xx} will be of dimension NxN. The vector of mean for this multivariate case will be of dimension 1xN. Cholesky decomposition of covariance matrix gives the equivalent standard deviation for the multivariate case. Cholesky decomposition can be viewed as square root operation. Matlab's randn function is used here to generate the multi-dimensional Gaussian random process with the given mean matrix and covariance matrix.

```
%Verifying the constant PSD of White Gaussian Noise Process
%with arbitrary mean and standard deviation sigma

mu=0; %Mean of each realization of White Gaussian Noise Process
sigma=2; %Sigma of each realization of White Gaussian Noise Process

L = 1000; %Number of Random Signal realizations to average
N = 1024; %Sample length for each realization set as power of 2 for FFT

%Generating the Random Process - White Gaussian Noise process
MU=mu*ones(1,N); %Vector of mean for all realizations
Cxx=(sigma^2)*diag(ones(N,1)); %Covariance Matrix for the Random Proces
R = chol(Cxx); %Cholesky of Covariance Matrix
%Generating a Multivariate Gaussian Distribution with given mean vector
%Covariance Matrix Cxx
z = repmat(MU,L,1) + randn(L,N)*R;
```

Compute PSD of the above generated multi-dimensional process and average it to get a smooth plot.

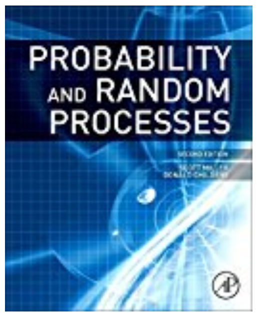
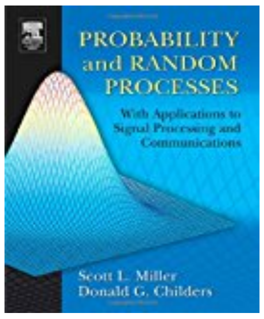
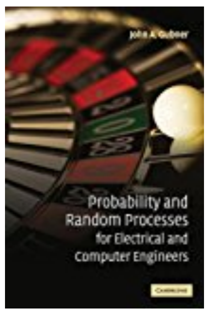
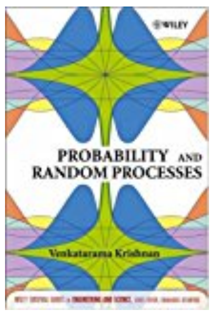
```
%By default, FFT is done across each column - Normal command fft(z)
%Finding the FFT of the Multivariate Distribution across each row
%Command - fft(z,[],2)
Z = 1/sqrt(N)*fft(z,[],2); %Scaling by sqrt(N);
Pzavg = mean(Z.*conj(Z));%Computing the mean power from fft

normFreq=[-N/2:N/2-1]/N;
Pzavg=fftshift(Pzavg); %Shift zero-frequency component to center of spe
plot(normFreq,10*log10(Pzavg),'r');
axis([-0.5 0.5 0 10]); grid on;
ylabel('Power Spectral Density (dB/Hz)');
xlabel('Normalized Frequency');
title('Power spectral density of white noise');
```

The PSD plot of a white noise gives almost fixed power in all the frequencies. In other words, for a white noise signal, the PSD is constant (flat) across all the frequencies (-∞ to + ∞). The y-axis in the above plot is expressed in dB/Hz unit. We can see from the plot that the constant power = 10*log10(σ^2)=10*log10(4)=6 dB.

References:

[1] Robert Grover Brown, Introduction to Random Signal Analysis and Kalman Filtering. John Wiley and Sons, 1983 [2] Athanasios Papoulis, Probability, Random Variables, and Stochastic Processes, 3rd ed. WCB/McGraw-Hill, 1991

Recommended Books on Probability and Random Process:

	 PROBABILITY and RANDOM PROCESSES With Applications to Signal Processing and Communications Scott L. Miller Donald G. Childers	 Probability and Random Processes for Electrical and Computer Engineers	Steven Kay Intuitive Probability and Random Processes using MATLAB	

Share this:

Share 4 Tweet Partager Share submit tumblr.
More

Previous Post
Breaking the Bandwidth Barrier – Li-Fi turns LED bulbs into High Speed Internet Hotspots

Next Post
Simulation of DPSK performance curves in Matlab

Mathuranathan

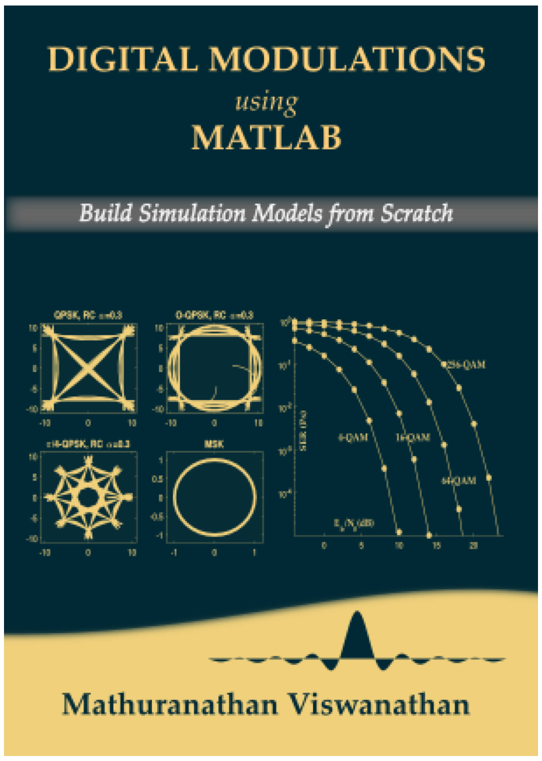
<a href="https://plus.google.com/116291006484377164553?rel=author" rel="author">Mathuranathan Viswanathan</a> - Founder and Author @ gaussianwaves.com which has garnered worldwide readership. He is a masters in communication engineering and has 9 years of technical expertise in channel modeling and has worked in various technologies ranging from read channel design for hard drives, GSM/EDGE/GPRS, OFDM, MIMO, 3GPP PHY layer and DSL. He also specializes in tutoring on various subjects like signal processing, random process, digital communication etc.., <a href="https://sg.linkedin.com/pub/mathuranathan-viswanathan/5/723/b2a"> LinkedIn Profile</a>

CHANNEL MODELLING LATEST ARTICLES MATLAB CODES PROBABILITY RANDOM PROCESS TIPS & TRICKS

AUTO-CORRELATION COVARIANCE MATRIX FOURIER ANALYSIS MATLAB CODE MULTIVARIATE RANDOM VARIABLES POWER SPECTRAL DENSITY WHITE NOISE WIENER-KHINTCHINE THEOREM

Log in with:

NEW RELEASE


DIGITAL MODULATIONS
using
MATLAB
Build Simulation Models from Scratch
Mathuranathan Viswanathan

SEARCH QUESTIONS

Search questions...

Search

SEARCH ARTICLES

Search...

Go

LIKE THIS WEBSITE ? SUBSCRIBE

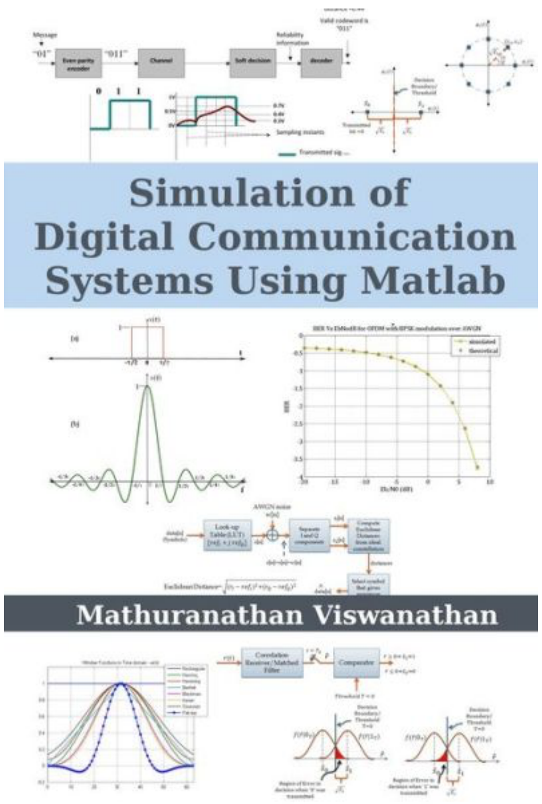
Enter your email address to subscribe to this blog and receive notifications of new posts by email.

Join 157 other subscribers

Email Address

Subscribe

GRAB THIS EBOOK TODAY



Simulation of Digital Communication Systems Using Matlab
Mathuranathan Viswanathan

RECENTLY SUBMITTED

A mathematical theory of communication – Shannon

An Introduction to Information Theory and Applications

More tutorials

CATEGORIES

8-PSK

Book reviews

BPSK

Channel Coding

Channel Modelling

Constellations

Correlative Coding

Digital Modulations

Digital Modulations using Matlab

DPSK

Estimation Theory

Free Books

GMSK

Hamming Codes

Inter Symbol Interference

Interleaver

Introduction

Latest Articles

Line Coding

M-PSK

M-QAM

Matlab Codes

MIMO systems

OFDM

Probability

Pulse Shaping

Python

QPSK

Quiz

Random Process

Reed Solomon codes

Shannon Theorem

Signal Processing

Software Defined Radio

Source Coding

Spread Spectrum

Tips & Tricks

Tutorials

User submitted Codes

VLSI

FOLLOW

Digital Communicatio…
7,309 likes

Like Page

Shop Now

Be the first of your friends to like this

GaussianWaves

Suivre

TAGS

Auto-Correlation Auto regressive AWGN BPSK Channel Capacity Channel Coding Channel Modelling cholesky convolution Cramer Rao Lower Bound CRLB Cross-Correlation Digital Modulations Estimation Fading FFT Fisher Information Fisher Matrix Fourier Analysis Fourier transform gray code M-QAM Matlab Code Matlab Codes matrix algebra Maximum Likelihood Estimation Minimum Variance Unbiased Estimator MLE Multi-carrier Modulation OFDM Orthogonal Frequency Division Multiplexing oversampling positive definite Power spectral Density PSD Pulse Shaping Random Process Random Variables Rayleigh Sampling Theorem Score Shannon Capacity Signal Processing Spread Spectrum Tips & Tricks

Rigol DS1054Z Digital...

$349.00 ✓Prime

Shop now

Phantom 3 Professional...

$1,099.00

Shop now

Arduino Uno R3 Microcontroller...

$19.85 ✓Prime

Shop now

Multimeter, Exwell Digital...

$15.81 ✓Prime

Shop now