clear all; close all; clc;

num\_samples = 10000; % Number of samples

num\_simulations = 3; % Number of simulations (random processes)

length\_data = 1000; % Number of time instances

% Generate the time axis (x-axis)

time\_axis = 1:num\_samples;

% Initialize matrix to store simulation data

simulation\_data = zeros(num\_simulations, num\_samples);

% Generate simulation data

for sim = 1:num\_simulations

% Generate random samples from a uniform distribution between -0.5 and 0.5

random\_samples = rand(1, num\_samples) - 0.5;

simulation\_data(sim, :) = random\_samples;

end

% Plot the random processes and calculate their mean and variance

figure;

for sim = 1:num\_simulations

subplot(num\_simulations, 2, 2\*sim - 1);

plot(time\_axis, simulation\_data(sim, :));

xlabel('Time');

ylabel('Value');

title(['Random Process ', num2str(sim) ' with \mu=0']);

grid on;

% Calculate and display mean and variance

process\_mean = mean(simulation\_data(sim, :));

process\_variance = var(simulation\_data(sim, :));

disp(['Random Process ', num2str(sim), ' - Mean: ', num2str(process\_mean), ', Variance: ', num2str(process\_variance)]);

% Plot the histogram and probability density curve

subplot(num\_simulations, 2, 2\*sim);

histogram(simulation\_data(sim, :), 'Normalization', 'pdf');

xlabel('Value');

ylabel('Probability Density');

title(['PDF of Random Process ', num2str(sim)]);

grid on;

end

% Calculate Ensemble Averages for specific time instances

time\_instants = [1, 100, 1000]; % Specific time instances for analysis

ensemble\_averages = mean(simulation\_data(:, time\_instants), 1);

% Display Ensemble Averages

disp('Ensemble Averages:');

for i = 1:length(time\_instants)

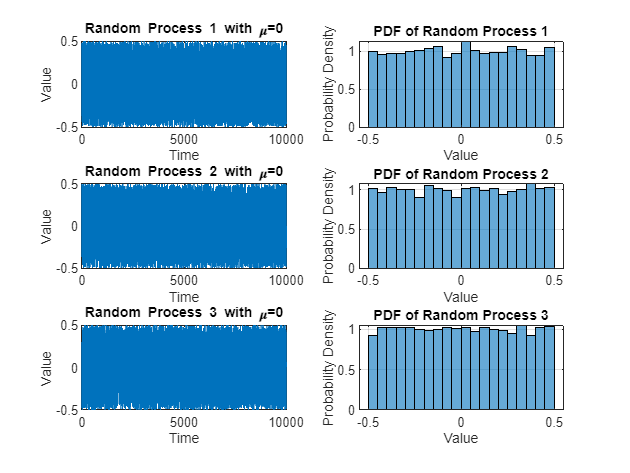
fprintf('At time instance %d: %f\n', time\_instants(i), ensemble\_averages(i));

end

Random Process 1 - Mean: 0.0013252, Variance: 0.082786

Random Process 2 - Mean: 0.0021248, Variance: 0.084013

Random Process 3 - Mean: 3.0132e-05, Variance: 0.083143



Ensemble Averages:

At time instance 1: 0.038166

At time instance 100: 0.044562

At time instance 1000: 0.103993

clear all; close all; clc;

% Parameters

num\_samples = 1000; % Number of samples

mean\_values = [0, 0, 0, 1, 1, 1]; % Different mean values

variance\_values = [1, 2, 3, 1, 2, 3]; % Different variance values

% Generate the time axis (x-axis)

time\_axis = 1:num\_samples;

% Initialize matrix to store simulation data

num\_simulations = length(mean\_values);

simulation\_data = zeros(num\_simulations, num\_samples);

% Loop through different mean and variance values

for i = 1:num\_simulations

mean\_val = mean\_values(i);

variance = variance\_values(i); % Using the variance value as the range for uniform distribution

% Generate random samples from a uniform distribution within the specified range

random\_samples = variance \* (rand(1, num\_samples) - 0.5) + mean\_val;

simulation\_data(i, :) = random\_samples;

% Create a new figure for each random process

figure;

% Plot the random process

subplot(2, 1, 1);

plot(time\_axis, random\_samples);

xlabel('Time');

ylabel('Value');

title(['Random Process with \mu=', num2str(mean\_val), ' and variance=', num2str(variance)]);

grid on;

% Plot the histogram and probability density curve

subplot(2, 1, 2);

histogram(random\_samples, 'Normalization', 'pdf');

xlabel('Value');

ylabel('Probability Density');

title('Probability Density Curve');

grid on;

end

% Calculate Time Averages and Variances for each simulation

time\_averages = mean(simulation\_data, 2);

variances = var(simulation\_data, 0, 2);

% Display Time Averages and Variances for each simulation

disp('Time Averages and Variances for each simulation:');

for sim = 1:num\_simulations

fprintf('Simulation %d - Time Average: %f, Variance: %f\n', sim, time\_averages(sim), variances(sim));

end

% Calculate Ensemble Averages for specific time instances

time\_instants = [1, 100, 1000]; % Specific time instances for analysis

ensemble\_averages = mean(simulation\_data(:, time\_instants), 1);

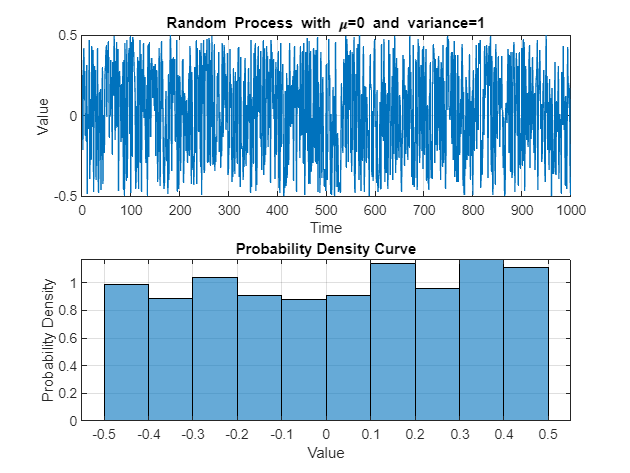
% Display Ensemble Averages

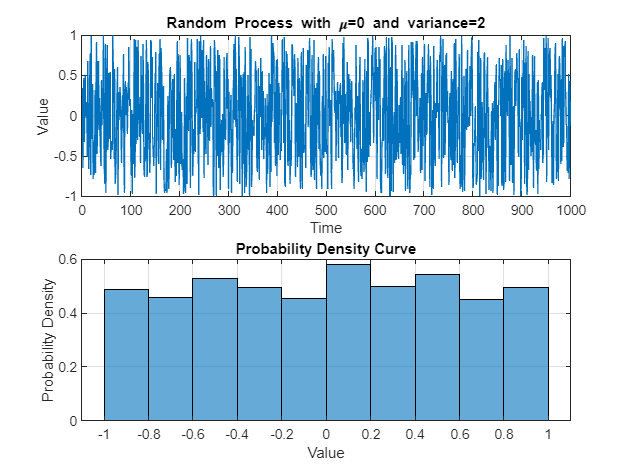
disp('Ensemble Averages:');

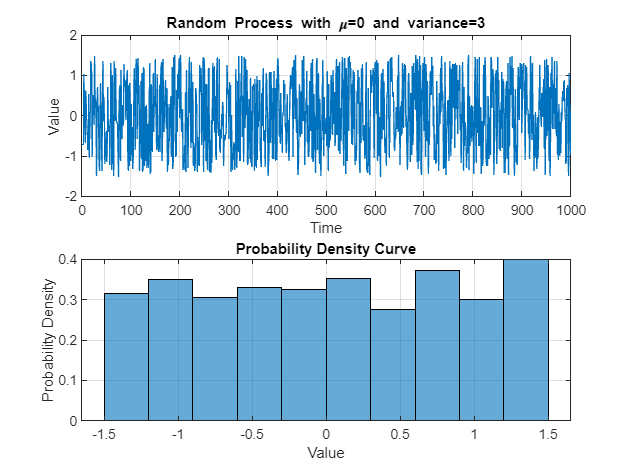
for i = 1:length(time\_instants)

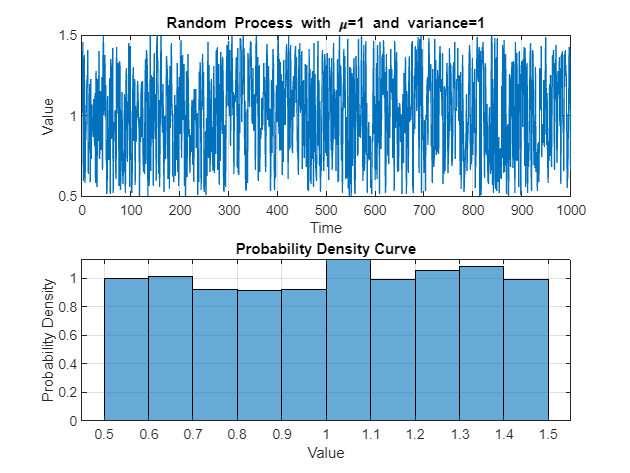
fprintf('At time instance %d: %f\n', time\_instants(i), ensemble\_averages(i));

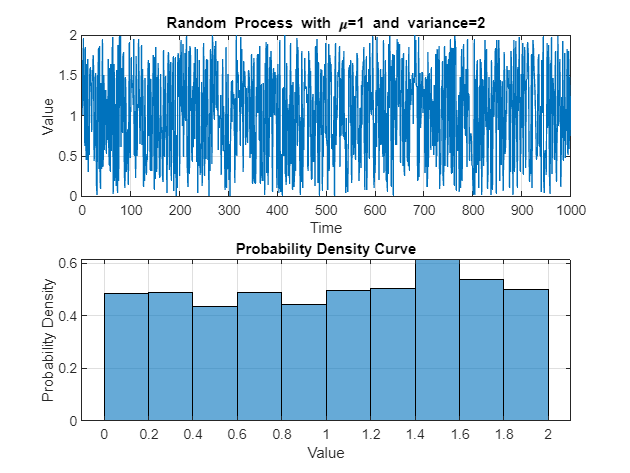
end

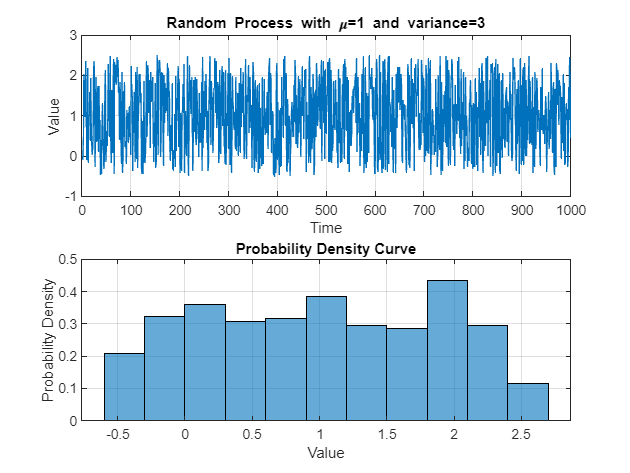












Time Averages and Variances for each simulation:

Simulation 1 - Time Average: 0.016643, Variance: 0.086960

Simulation 2 - Time Average: 0.002316, Variance: 0.329511

Simulation 3 - Time Average: 0.030596, Variance: 0.776681

Simulation 4 - Time Average: 1.007712, Variance: 0.083500

Simulation 5 - Time Average: 1.027138, Variance: 0.334084

Simulation 6 - Time Average: 1.009716, Variance: 0.748596

Ensemble Averages:

At time instance 1: 0.429959

At time instance 100: 0.028048

At time instance 1000: 0.176949

clear all; close all; clc;

num\_samples = 10000; % Number of samples

num\_simulations = 3; % Number of simulations (random processes)

length\_data = 1000; % Number of time instances

% Generate the time axis (x-axis)

time\_axis = 1:num\_samples;

% Initialize matrix to store simulation data

simulation\_data = zeros(num\_simulations, num\_samples);

% Generate simulation data

for sim = 1:num\_simulations

% Generate random samples from a normal distribution with mean 0 and variance 1

random\_samples = randn(1, num\_samples);

simulation\_data(sim, :) = random\_samples;

end

% Plot the random processes and calculate their mean and variance

figure;

for sim = 1:num\_simulations

subplot(num\_simulations, 2, 2\*sim - 1);

plot(time\_axis, simulation\_data(sim, :));

xlabel('Time');

ylabel('Value');

title(['Random Process ', num2str(sim) ' with \mu=0 and \sigma^2=1']);

grid on;

% Calculate and display mean and variance

process\_mean = mean(simulation\_data(sim, :));

process\_variance = var(simulation\_data(sim, :));

disp(['Random Process ', num2str(sim), ' - Mean: ', num2str(process\_mean), ', Variance: ', num2str(process\_variance)]);

% Plot the histogram and probability density curve

subplot(num\_simulations, 2, 2\*sim);

histogram(simulation\_data(sim, :), 'Normalization', 'pdf');

xlabel('Value');

ylabel('Probability Density');

title(['PDF of Random Process ', num2str(sim)]);

grid on;

end

% Calculate Ensemble Averages for specific time instances

time\_instants = [1, 100, 1000]; % Specific time instances for analysis

ensemble\_averages = mean(simulation\_data(:, time\_instants), 1);

% Display Ensemble Averages

disp('Ensemble Averages:');

for i = 1:length(time\_instants)

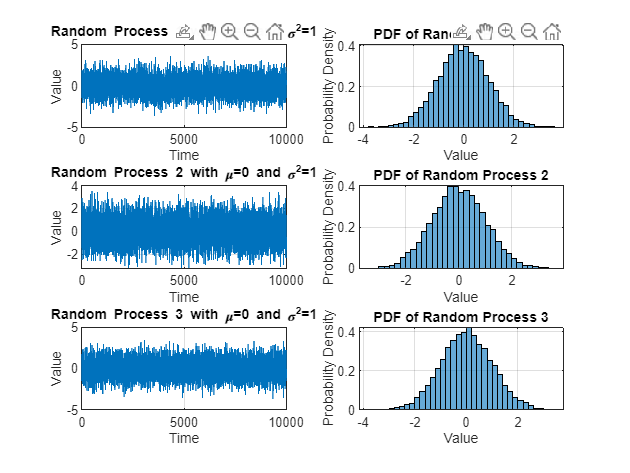
fprintf('At time instance %d: %f\n', time\_instants(i), ensemble\_averages(i));

end

Random Process 1 - Mean: -0.011009, Variance: 0.9906

Random Process 2 - Mean: 0.0083925, Variance: 1.0186

Random Process 3 - Mean: 0.011323, Variance: 0.98942



Ensemble Averages:

At time instance 1: 0.273020

At time instance 100: -1.187097

At time instance 1000: -0.209893

clear all; close all; clc;

% Parameters

num\_samples = 1000; % Number of samples

mean\_values = [0, 0, 0, 1, 1, 1]; % Different mean values

variance\_values = [1, 2, 3, 1, 2, 3]; % Different variance values

% Generate the time axis (x-axis)

time\_axis = 1:num\_samples;

% Initialize matrix to store simulation data

num\_simulations = length(mean\_values);

simulation\_data = zeros(num\_simulations, num\_samples);

% Loop through different mean and variance values

for i = 1:num\_simulations

mean\_val = mean\_values(i);

sigma = sqrt(variance\_values(i));

% Generate random samples from a normal distribution with specified mean and variance

random\_samples = sigma \* randn(1, num\_samples) + mean\_val;

simulation\_data(i, :) = random\_samples;

% Create a new figure for each random process

figure;

% Plot the random process

subplot(2, 1, 1);

plot(time\_axis, random\_samples);

xlabel('Time');

ylabel('Value');

title(['Random Process with \mu=', num2str(mean\_val), ' and \sigma^2=', num2str(sigma^2)]);

grid on;

% Plot the histogram and probability density curve

subplot(2, 1, 2);

histogram(random\_samples, 'Normalization', 'pdf');

xlabel('Value');

ylabel('Probability Density');

title('Probability Density Curve');

grid on;

end

% Calculate Time Averages and Variances for each simulation

time\_averages = mean(simulation\_data, 2);

variances = var(simulation\_data, 0, 2);

% Display Time Averages and Variances for each simulation

disp('Time Averages and Variances for each simulation:');

for sim = 1:num\_simulations

fprintf('Simulation %d - Time Average: %f, Variance: %f\n', sim, time\_averages(sim), variances(sim));

end

% Calculate Ensemble Averages for specific time instances

time\_instants = [1, 100, 1000]; % Specific time instances for analysis

ensemble\_averages = mean(simulation\_data(:, time\_instants), 1);

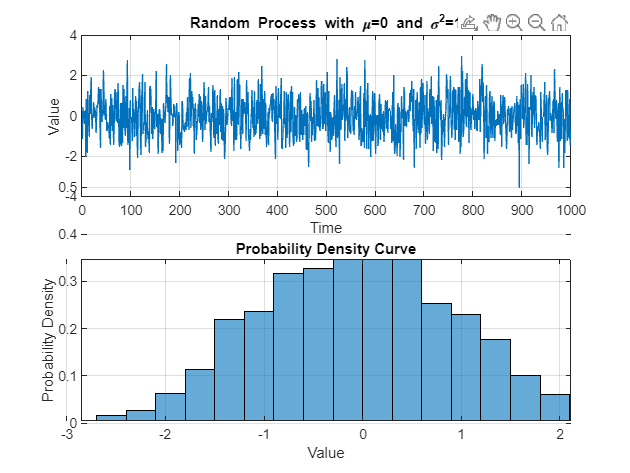
% Display Ensemble Averages

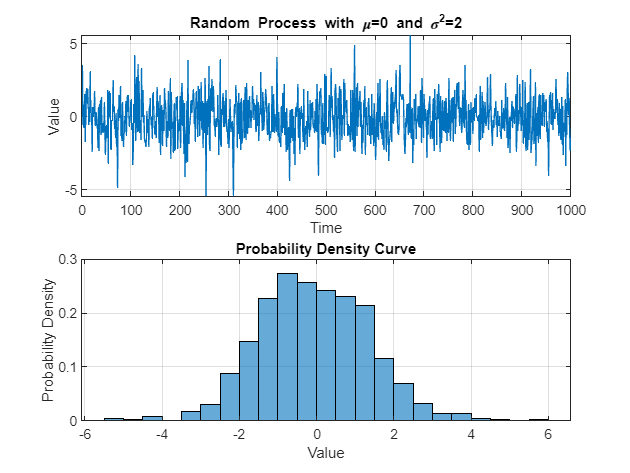
disp('Ensemble Averages:');

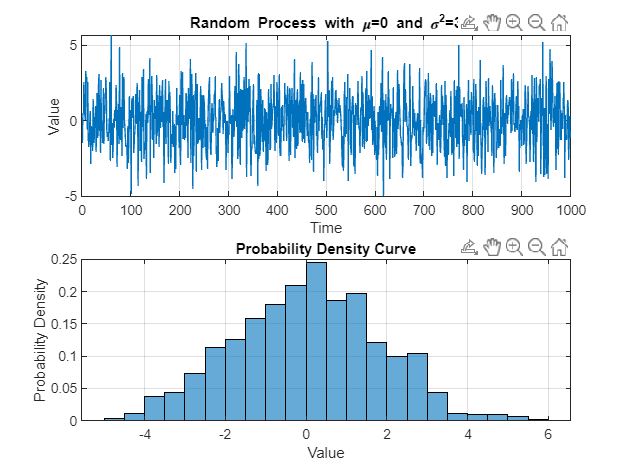
for i = 1:length(time\_instants)

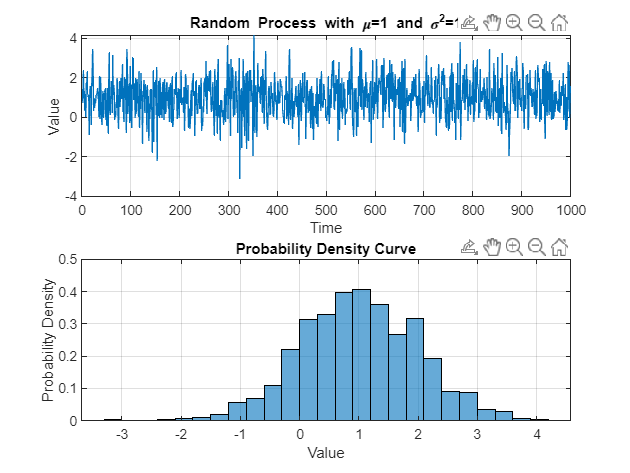
fprintf('At time instance %d: %f\n', time\_instants(i), ensemble\_averages(i));

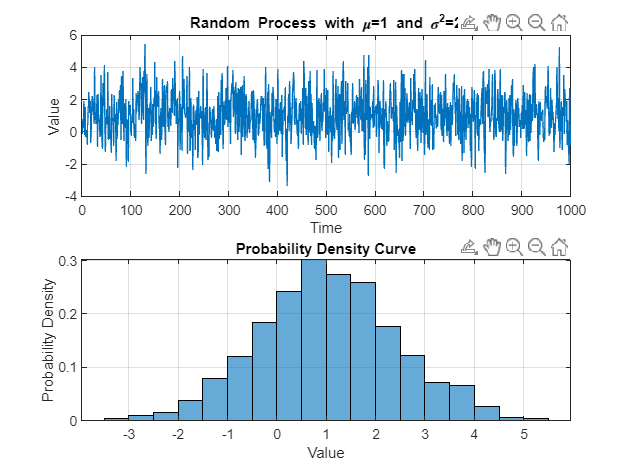
end

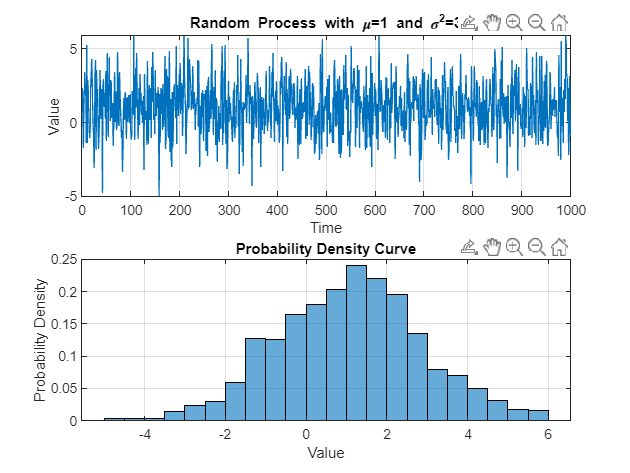












Time Averages and Variances for each simulation:

Simulation 1 - Time Average: -0.033927, Variance: 0.998411

Simulation 2 - Time Average: -0.065927, Variance: 2.021845

Simulation 3 - Time Average: 0.036310, Variance: 3.254506

Simulation 4 - Time Average: 1.008753, Variance: 0.995511

Simulation 5 - Time Average: 1.042579, Variance: 1.958888

Simulation 6 - Time Average: 1.067744, Variance: 3.246753

Ensemble Averages:

At time instance 1: 0.767493

At time instance 100: 0.371001

At time instance 1000: -0.512337

clear all; close all; clc;

%% Generate Transmitted Signal x(t)

Fs = 1000; % Sampling frequency (Hz)

T = 1; % Time duration of the signal (s)

t = 0:1/Fs:T-1/Fs; % Time vector

pi = 3.147;

% Generate Transmitted Signal x(t) (1 kHz sinusoidal signal with unit power)

x\_t = sqrt(2) \* cos(2\*pi\*1000\*t);

% SNR values in dB

SNR\_values = [0, 10, 20];

for SNR\_dB = SNR\_values

%% Calculate Noise Power (Variance) from SNR

SNR\_linear = 10^(SNR\_dB / 10); % Convert SNR to linear scale

noise\_variance = var(x\_t) / SNR\_linear; % Calculate noise power (variance)

%% Generate Gaussian Noise Signal n(t)

gaussian\_noise = sqrt(noise\_variance) \* randn(size(t)); % Gaussian noise with mean 0 and calculated variance

%% Generate Received Signal y(t)

y\_t = x\_t + gaussian\_noise;

%% Plot Transmitted and Received Signals

figure;

subplot(3, 1, 1);

plot(t, x\_t);

title('Transmitted Signal x(t)');

xlabel('Time');

ylabel('Amplitude');

subplot(3, 1, 2);

plot(t, gaussian\_noise);

title(['Gaussian Noise Signal n(t) for SNR=' num2str(SNR\_dB) ' dB']);

xlabel('Time');

ylabel('Amplitude');

subplot(3, 1, 3);

plot(t, y\_t);

title(['Received Signal y(t) for SNR=' num2str(SNR\_dB) ' dB']);

xlabel('Time');

ylabel('Amplitude');

sgtitle(['Effect of Noise on Received Signal for SNR=' num2str(SNR\_dB) ' dB']);

end

