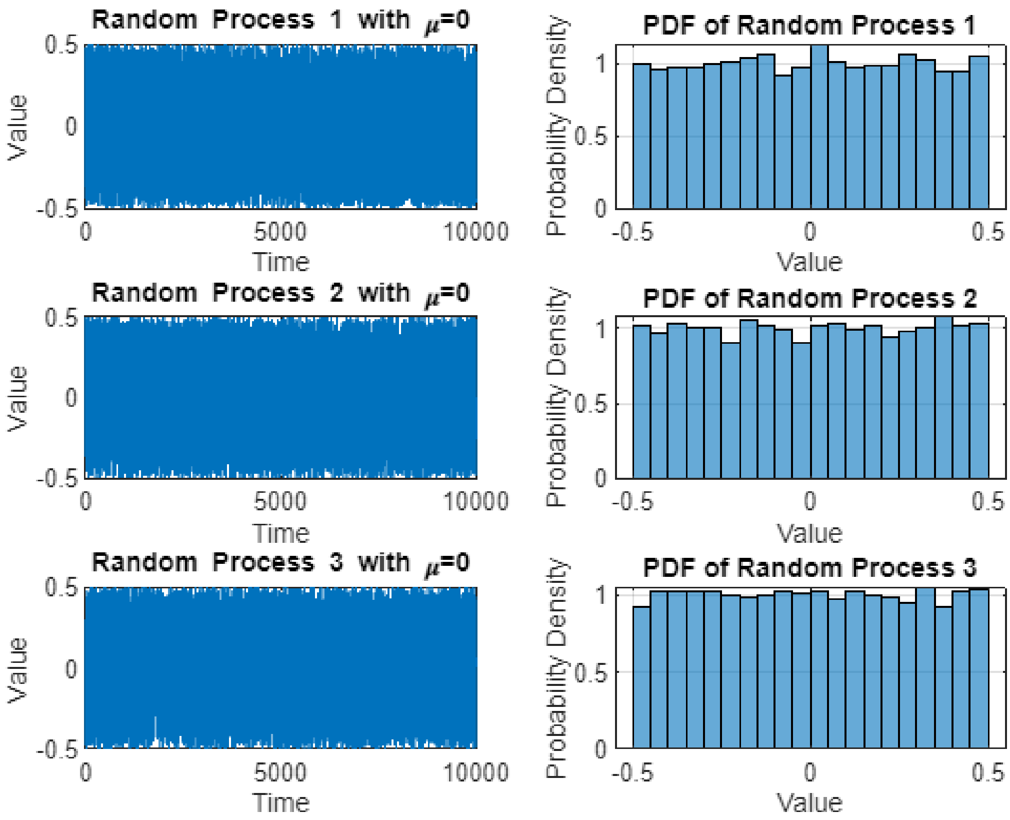
```
clear all; close all; clc;


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
```

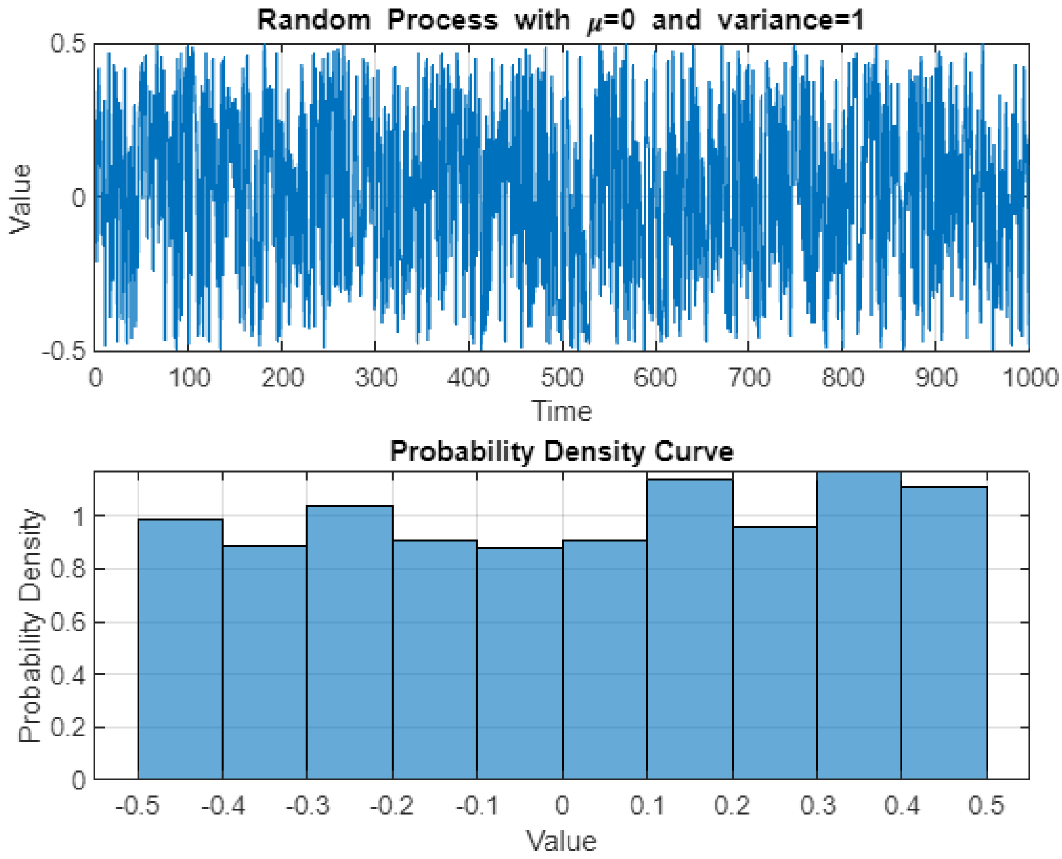


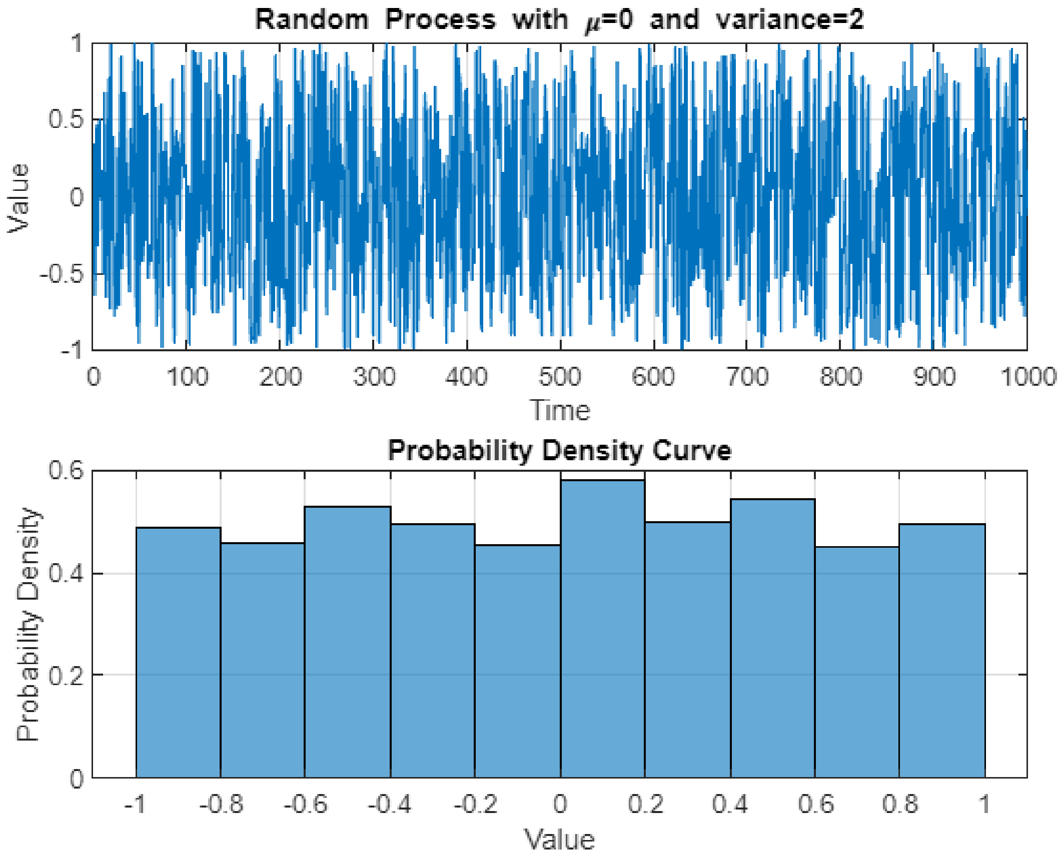

```
Ensemble Averages:
At time instance 1: 0.038166
At time instance 100: 0.044562
At time instance 1000: 0.103993
```

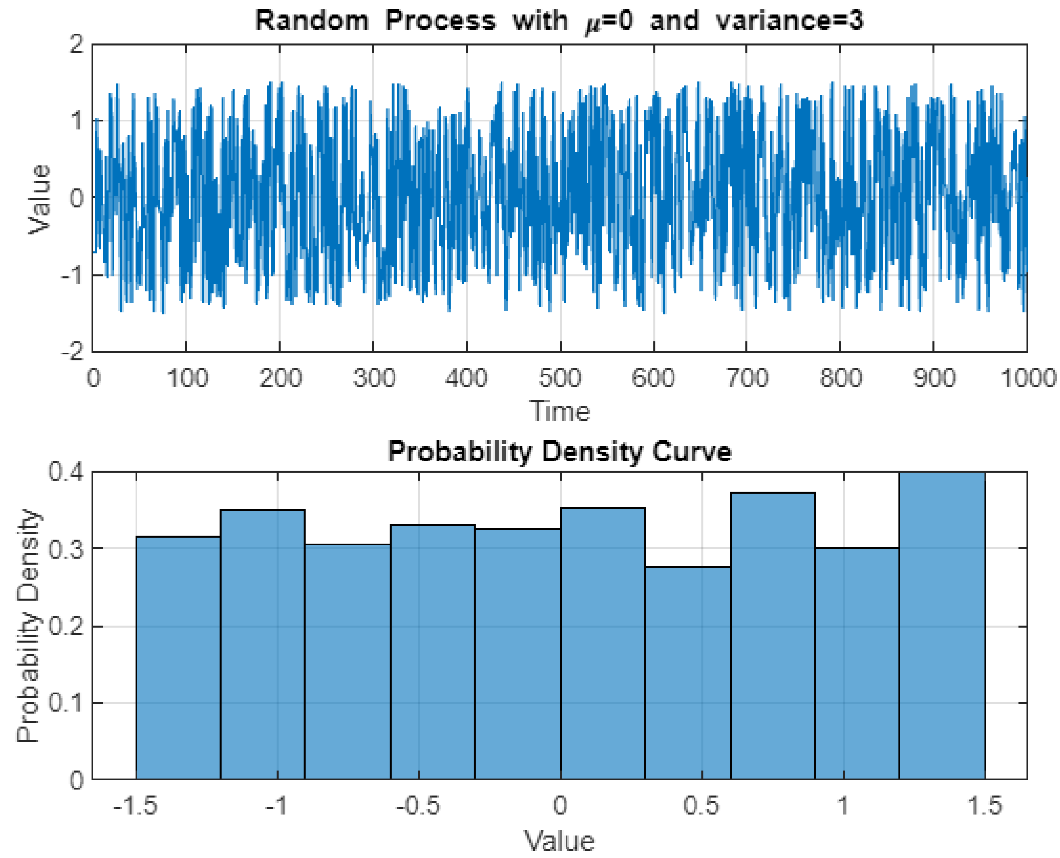
```
clear all; close all; clc;


% Parameters
num_samples = 1000;   % Number of samples
mean_values = [0, 0, 0, 1, 1, 1];  % Different mean values
```

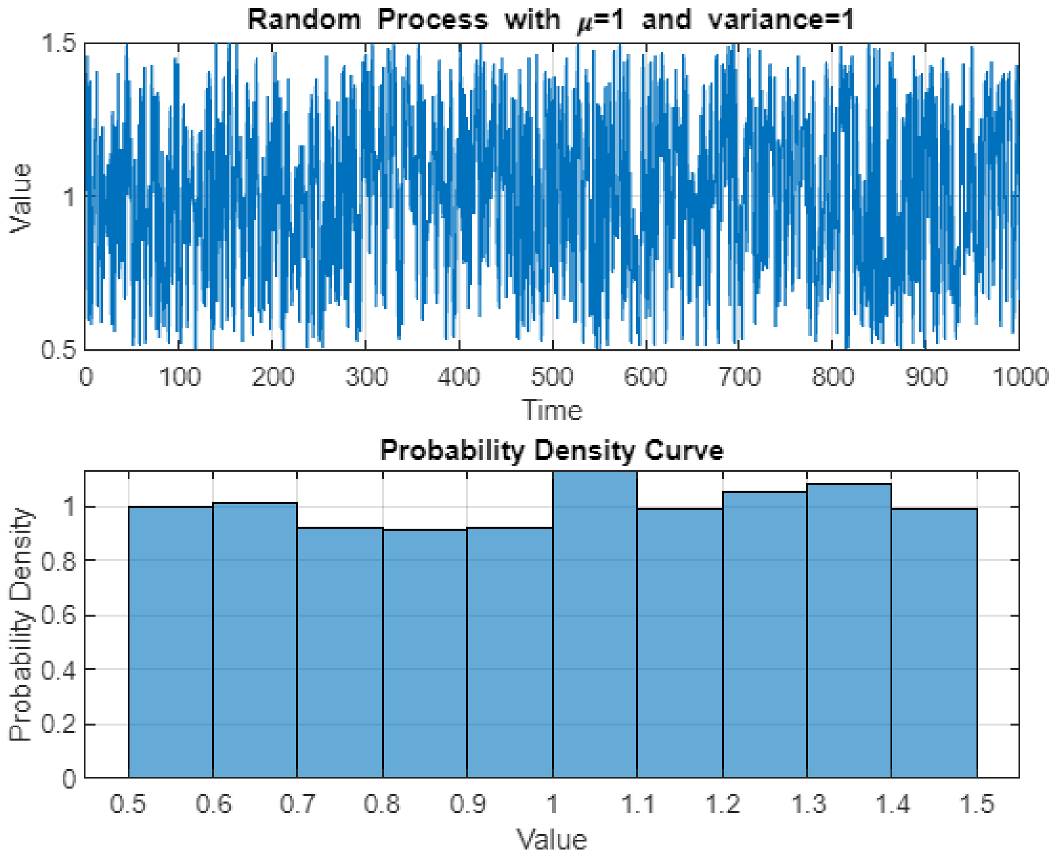
```
variance_values = [1, 2, 3, 1, 2, 3];  % Different variance values


% 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);
```

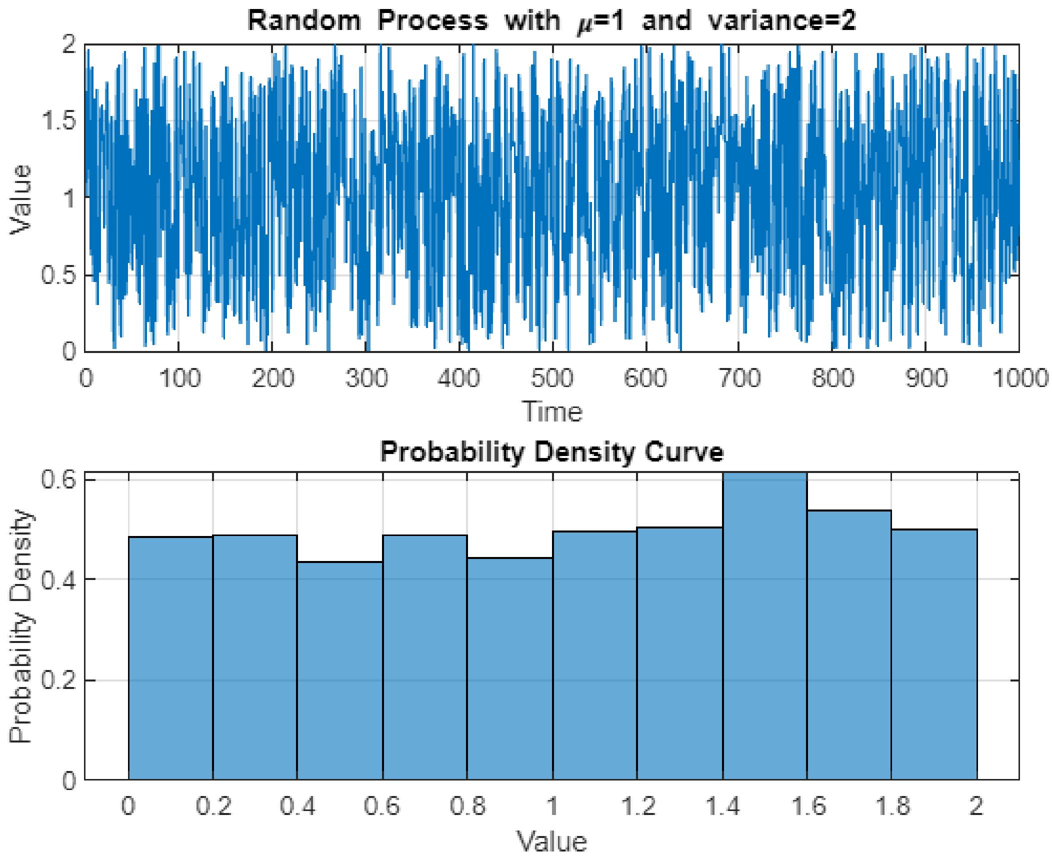
```
% Display Time Averages and Variances for each simulation
disp('Time Averages and Variances for each simulation:');
for sim = 1:num_simulations
    fprintf('Simulation %d - Time Average: %f, Variance: %f\n', sim,
time_averages(sim), variances(sim));
end


% 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
```

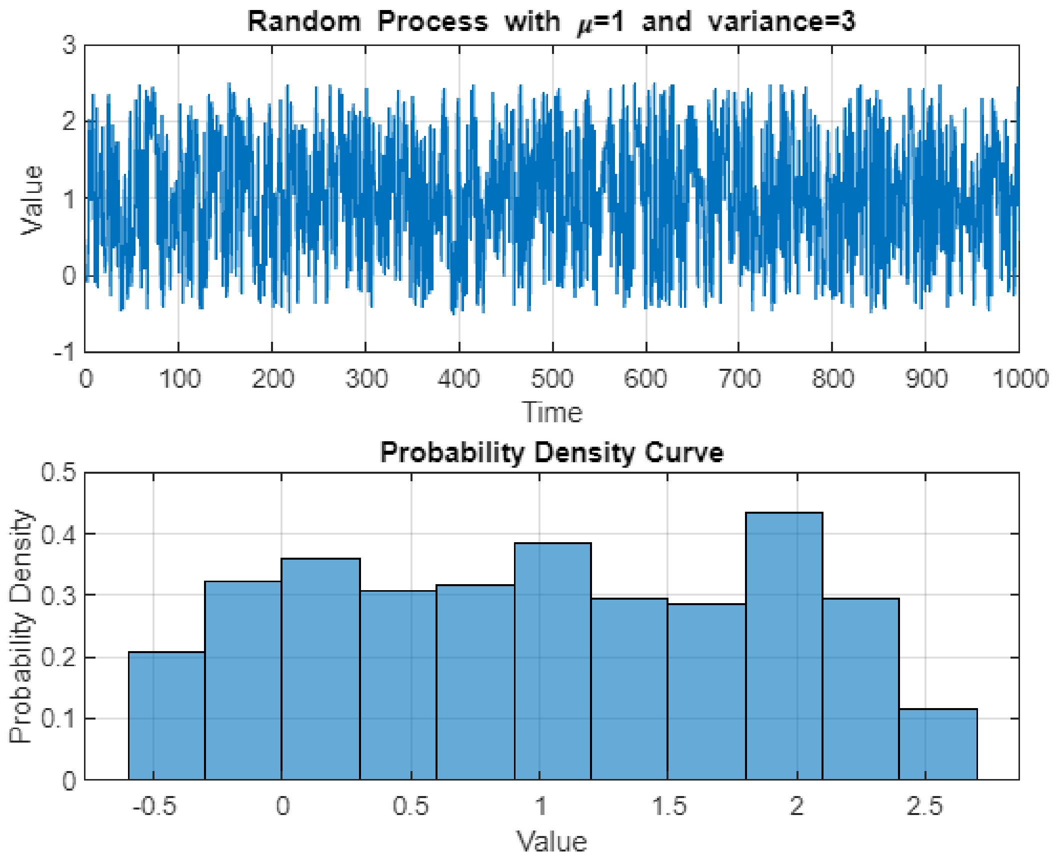



Random Process with μ=0 and variance=2
Value
Time
1
0.5
0
-0.5
-1
0 100 200 300 400 500 600 700 800 900 1000
Probability Density Curve
Probability Density
Value
0.6
0.4
0.2
0
-1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1


Random Process with μ=0 and variance=3
Value
Time
2
1
0
-1
-2
0 100 200 300 400 500 600 700 800 900 1000
Probability Density Curve
Probability Density
Value
0.4
0.3
0.2
0.1
0
-1.5 -1 -0.5 0 0.5 1 1.5


Random Process with μ=1 and variance=1
Value
Time
0 100 200 300 400 500 600 700 800 900 1000
0.5 1 1.5
Probability Density Curve
Probability Density
Value
0.5 0.6 0.7 0.8 0.9 1 1.1 1.2 1.3 1.4 1.5
0 0.2 0.4 0.6 0.8 1


Random Process with μ=1 and variance=2
Value
Time
0 100 200 300 400 500 600 700 800 900 1000
0 0.5 1 1.5 2
Probability Density Curve
Probability Density
Value
0 0.2 0.4 0.6 0.8 1 1.2 1.4 1.6 1.8 2
0 0.2 0.4 0.6



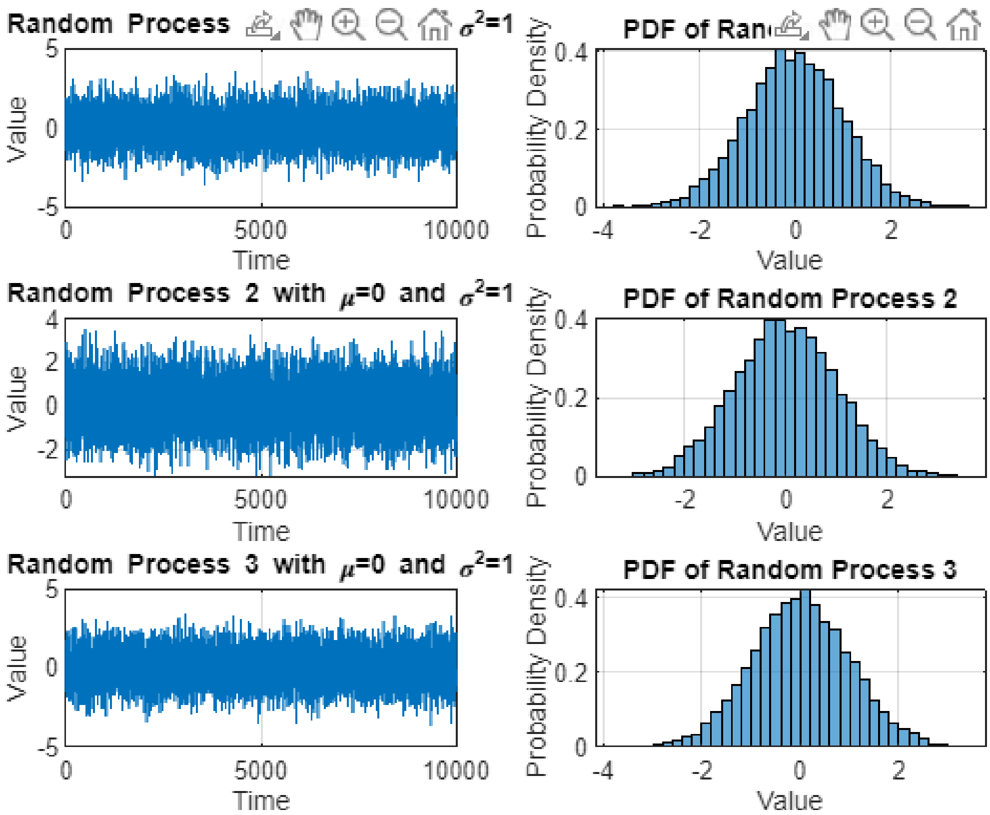

```
Time Averages and Variances for each simulation:
Simulation 1 - Time Average: 0.016643, Variance: 0.086960
Simulation 2 - Time Average: 0.002316, Variance: 0.329511
Simulation 3 - Time Average: 0.030596, Variance: 0.776681
Simulation 4 - Time Average: 1.007712, Variance: 0.083500
Simulation 5 - Time 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
```

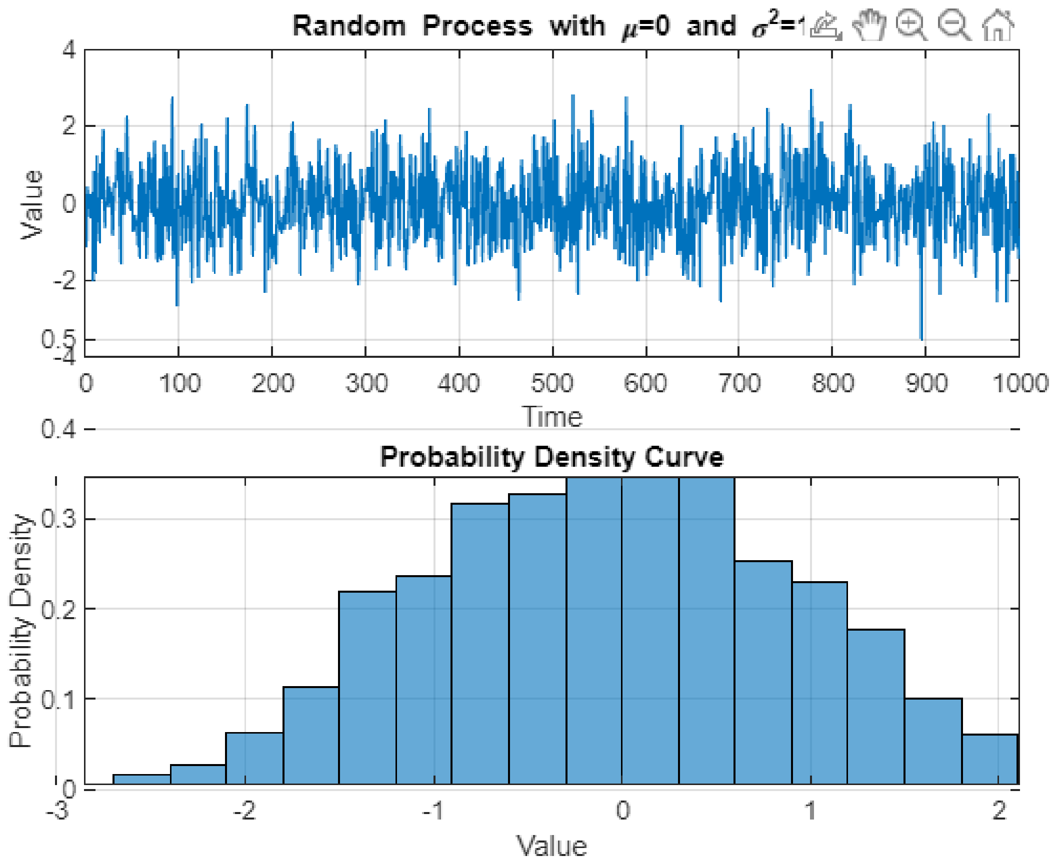


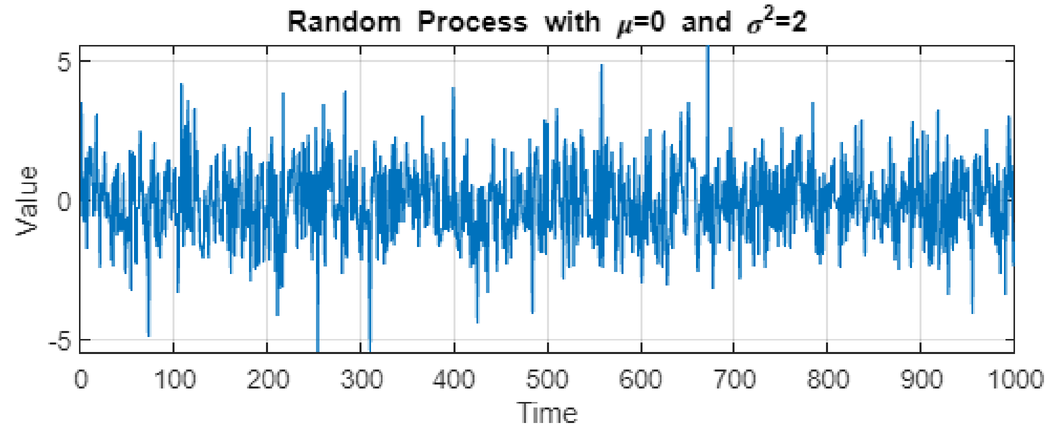

```
Ensemble Averages:
At time instance 1: 0.273020
At time instance 100: -1.187097
At time instance 1000: -0.209893
```

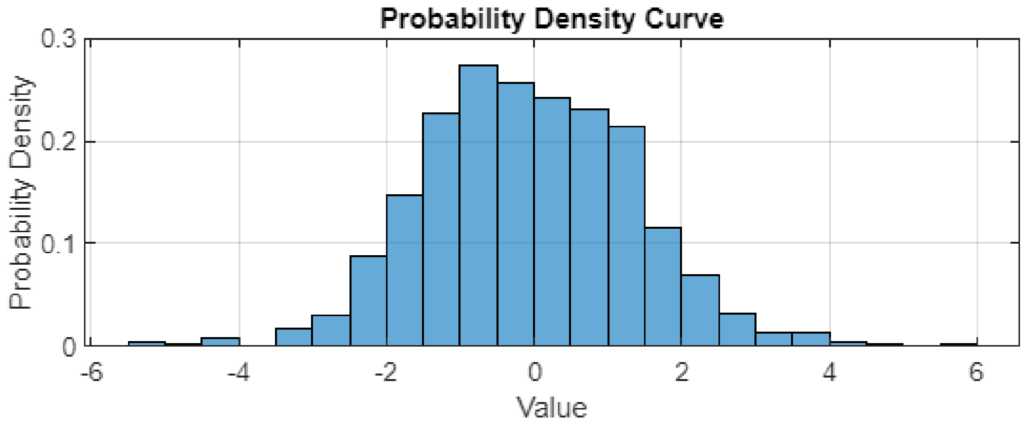
```
clear all; close all; clc;

% 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);
```

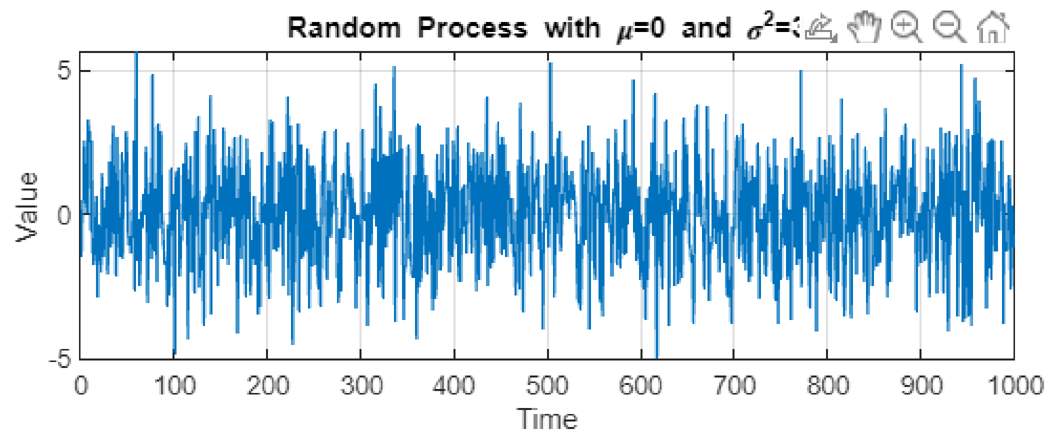
```
% Loop through different mean and variance values
for i = 1:num_simulations
    mean_val = mean_values(i);
    sigma = sqrt(variance_values(i));

    % 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);
```

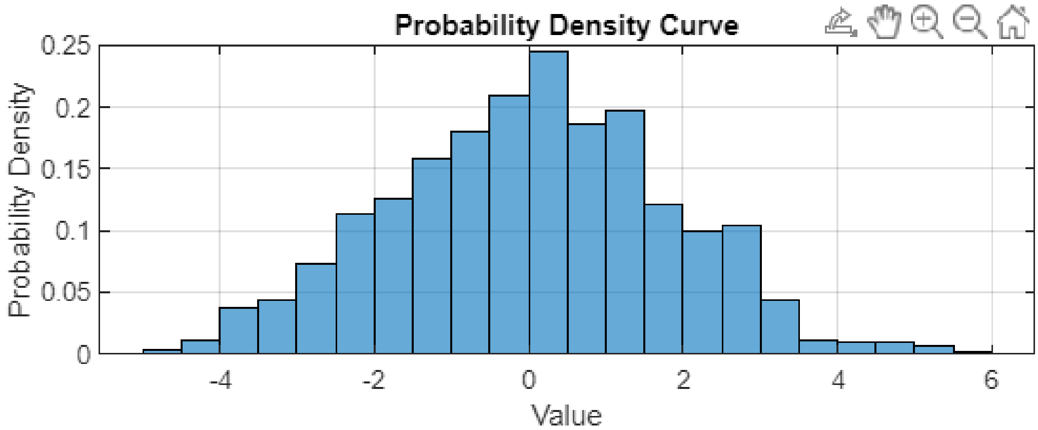
```
% Display Ensemble Averages
disp('Ensemble Averages:');
for i = 1:length(time_instants)
    fprintf('At time instance %d: %f\n', time_instants(i),
ensemble_averages(i));
end
```

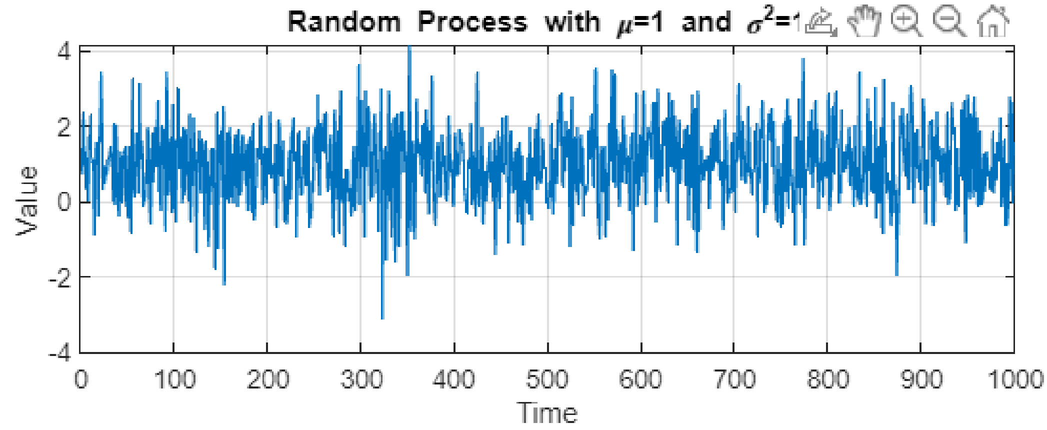


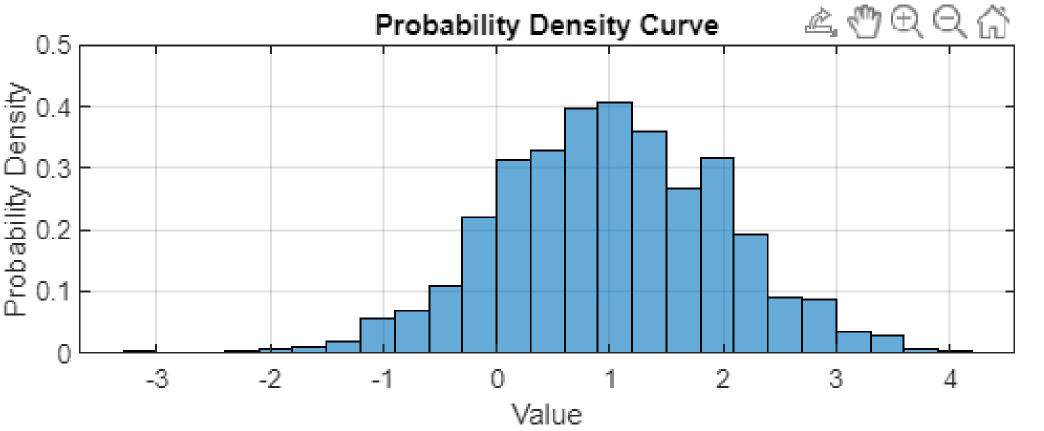

Random Process with μ=0 and σ²=2
Value
Time

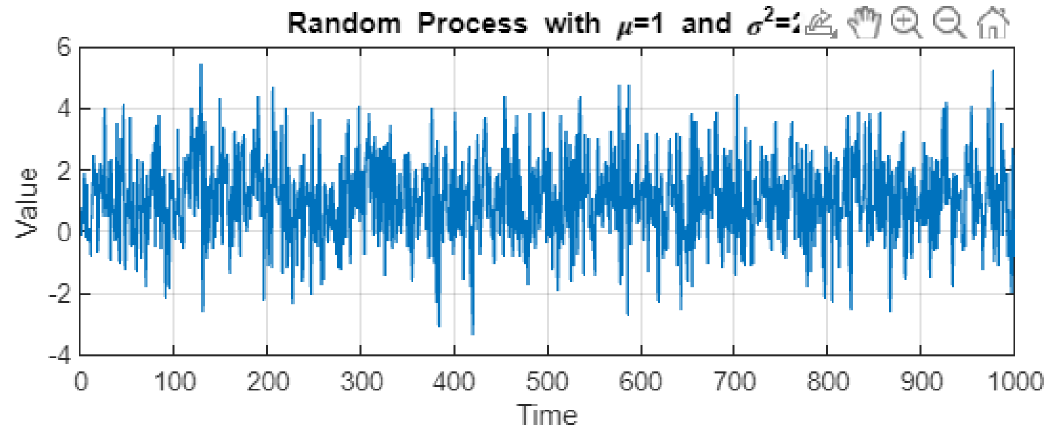

Probability Density Curve
Probability Density
Value

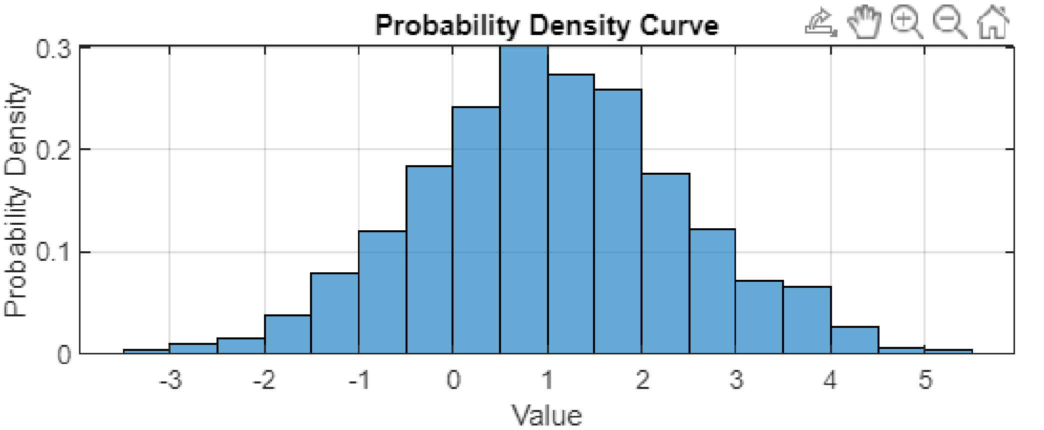

Random Process with μ=0 and σ²=
Value
Time

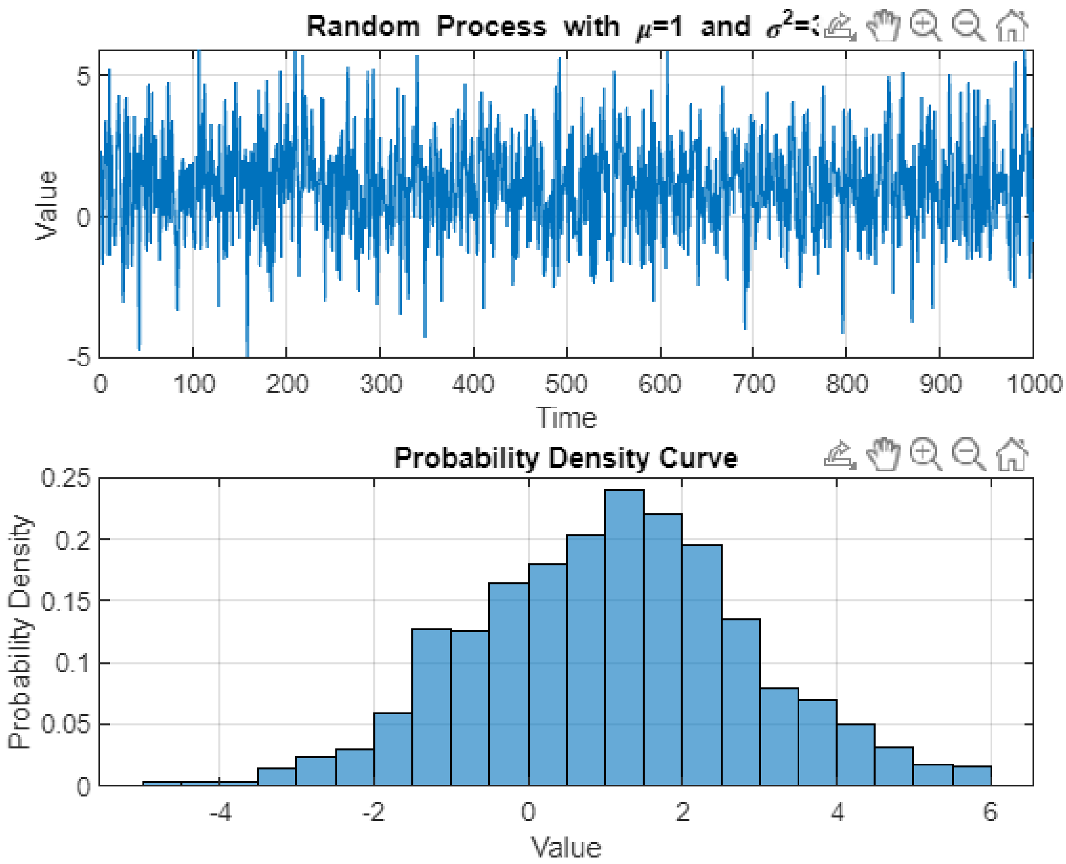

Probability Density Curve
Probability Density
Value


Random Process with μ=1 and σ²=
Value
Time
0
100
200
300
400
500
600
700
800
900
1000
4
2
0
-2
-4


Probability Density Curve
Probability Density
Value
0.5
0.4
0.3
0.2
0.1
0
-3
-2
-1
0
1
2
3
4


Random Process with μ=1 and σ²=
Value
Time
6
4
2
0
-2
-4
0
100
200
300
400
500
600
700
800
900
1000

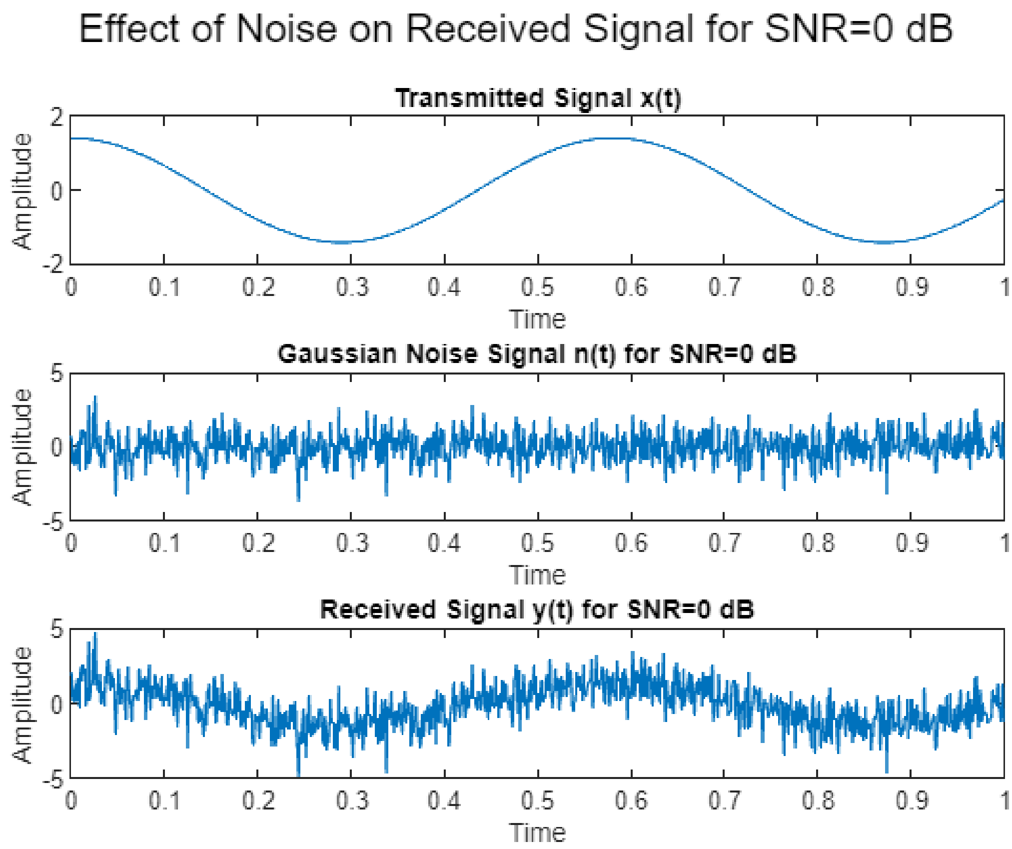

Probability Density Curve
Probability Density
Value
0.3
0.2
0.1
0
-3
-2
-1
0
1
2
3
4
5

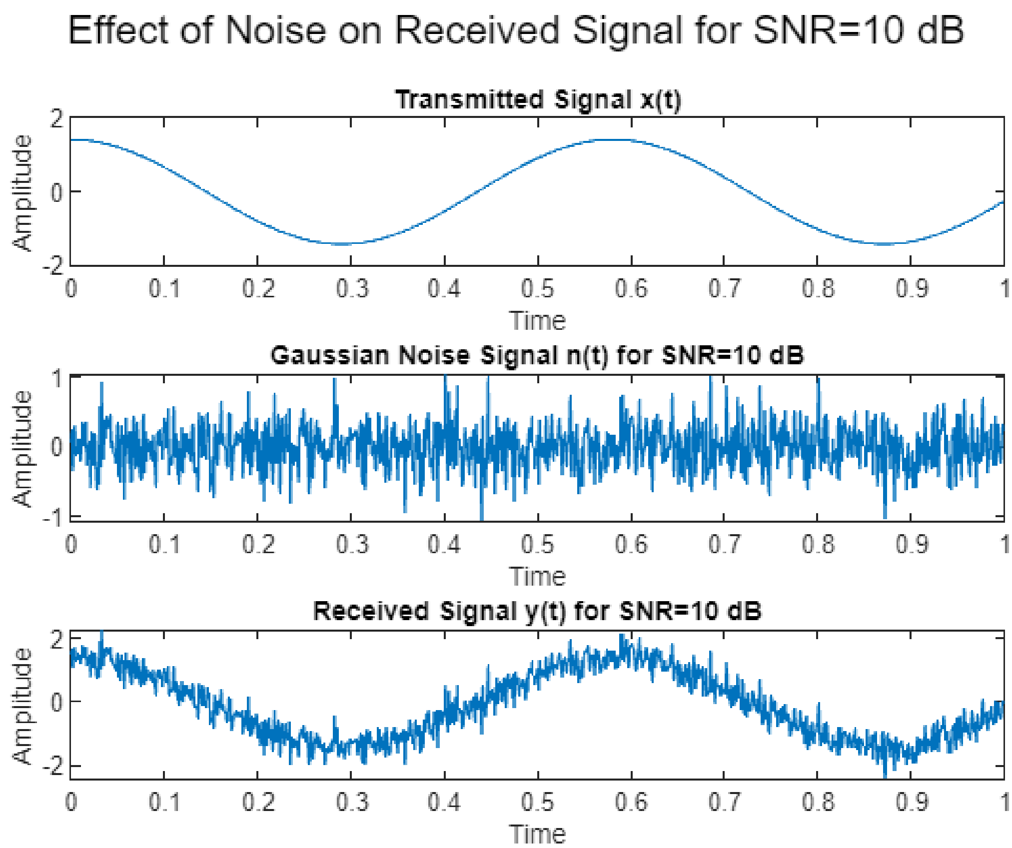


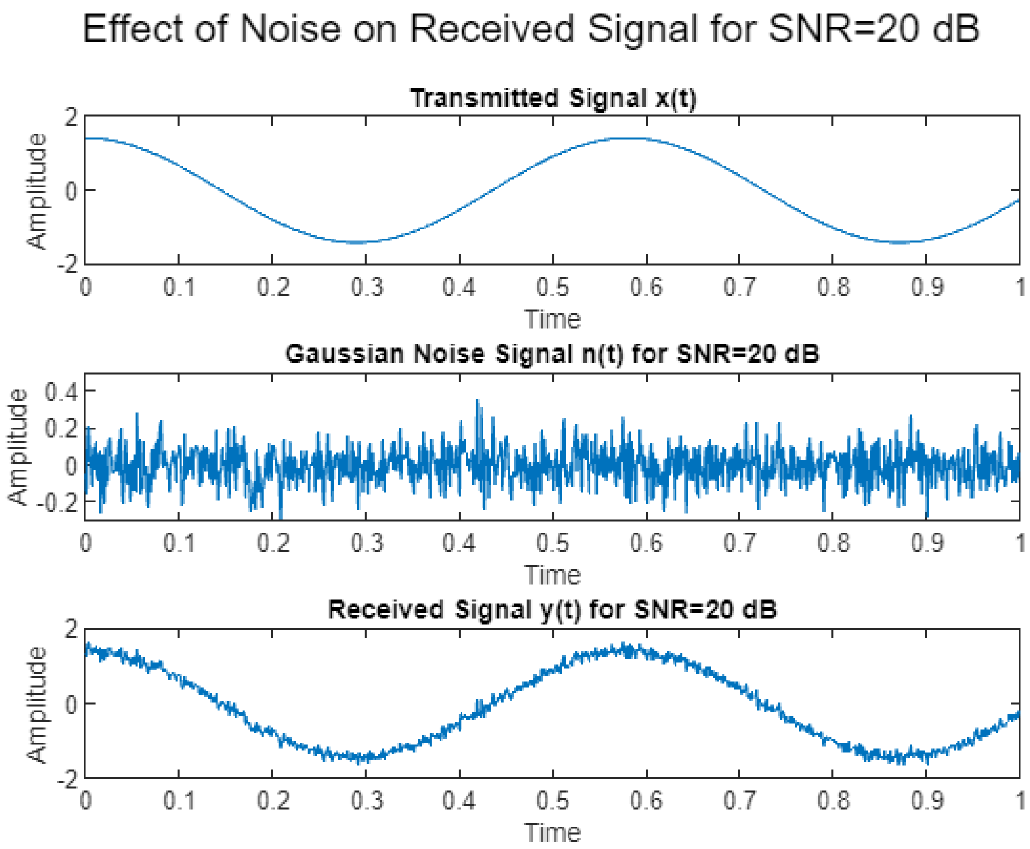

```
Time Averages and Variances for each simulation:
Simulation 1 - Time Average: -0.033927, Variance: 0.998411
Simulation 2 - Time Average: -0.065927, Variance: 2.021845
Simulation 3 - Time Average: 0.036310, Variance: 3.254506
Simulation 4 - Time Average: 1.008753, Variance: 0.995511
Simulation 5 - 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
```


Effect of Noise on Received Signal for SNR=0 dB
Transmitted Signal x(t)
Gaussian Noise Signal n(t) for SNR=0 dB
Received Signal y(t) for SNR=0 dB
Amplitude
Time
2
0
-2
5
-5
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Effect of Noise on Received Signal for SNR=10 dB
Transmitted Signal x(t)
Gaussian Noise Signal n(t) for SNR=10 dB
Received Signal y(t) for SNR=10 dB
Amplitude
Time
2
0
-2
1
-1
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9


Effect of Noise on Received Signal for SNR=20 dB
Transmitted Signal x(t)
Amplitude
2
0
-2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Time
Gaussian Noise Signal n(t) for SNR=20 dB
Amplitude
0.4
0.2
0
-0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Time
Received Signal y(t) for SNR=20 dB
Amplitude
2
0
-2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Time