STOCHASTIC SIGNALS AND SYSTEMS

WS 2018

ASSIGNMENT 1

RANDOM VARIABLES AND RANDOM NUMBERS

Date: 20/11/2018

by

Shinde Mrinal Vinayak (Matriculation No.: 5021349)
Yama Sanath Kumar(Matriculation No.: 5021387)

guided by

Prof. Dr. -Ing. Dieter Kraus

Contents

List of Figures

Chapter 1

Random variables and random Numbers

Random variables:

Random variable is a way to represent a sample space by assigning one or more numerical values to each outcome. If the outcome produces one value, the probability distribution is univariate and if the outcome produces more than one value the probability distribution is multi variate. If the sample space is finite or countable infinite then random variable is a discrete random variable. A continuous random variable takes uncountably infinite many values in an interval of real numbers.

Probability density function:

It is a statistical expression that defines a probability distribution for a continuous random variable as opposed to a discrete random variable. When the PDF is graphically portrayed, the area under the curve will indicate the interval in which the variable will fall. The total area in this interval of the graph equals the probability of a continuous random variable occurring.

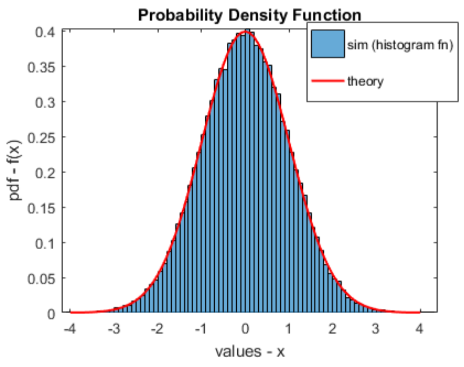


Figure 1.1: Probability density function

1.1 Uniform distribution

Task: Specify the density function, distribution function and determine the expected value and variance of an on the interval *[a, b]* uniformly distributed random variable X.

Solution:

The uniform distribution is one of the simplest probability distributions. It is a continuous distribution, this means that it takes values within a specified range, e.g. between 0 and 1. And sometimes it is also known as a rectangular distribution. It has a constant probability.

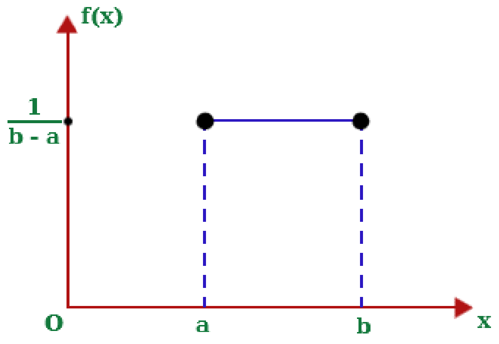


Figure 1.2: Uniform distribution of a random variable

- Probability density function: It is given by f(x) in the interval [a,b and is defined as,]

$$f(x) = \begin{cases} \frac{1}{b-a}, & a \leq x \leq b \\ 0, & \text{otherwise} \end{cases}$$

- Cumulative distribution function: It is method to describe the distribution of random variables. The advantage is that it can be defined for any kind of random variable (discrete, continuous, and mixed). The cumulative distribution function (CDF) of random variable $F_X(X)$ is defined as,

$$F_X(X) = \begin{cases} 0, & \text{for x<a} \\ \frac{x-a}{b-a}, & \text{for } a \leq x \leq b \\ 1, & \text{for x>b} \end{cases}$$

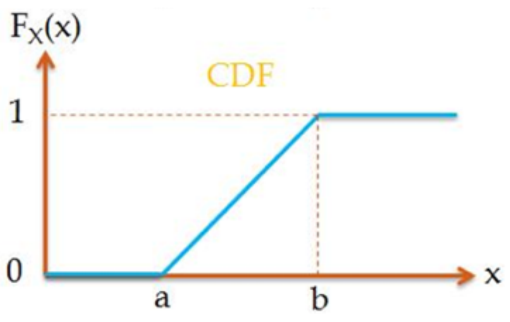


Figure 1.3: Cumulative distribution function of a random variable

- Expected value: It is also called as the mean value. E[X] of the Uniform distributed random variable in the interval [a, b] is given as,

$$\text{E[X]}=\int_{-\infty}^{\infty} Xf(x)dx \qquad \text{f(x)}=\frac{1}{b-a}.$$

$$E[X] = \frac{1}{b-a}\int_a^b x dx$$

$$E[X] = \frac{1}{b-a}\left[\frac{x^2}{2}\right]_a^b$$

$$E[X] = \frac{b^2-a^2}{2(b-a)}$$

$$E[X] = \frac{b+a}{2} = \mu$$

- Variance: `V[X]` of a Uniformly distributed random variable in the interval [a, b] is as,

$$V[X] = \sigma^2 = E(X-\mu)^2 = E(X^2 - 2\mu X + \mu^2)$$

$$V[X] = E(X^2) - 2E(X)\mu + \mu^2$$

$$V[X] = E(X^2) - (E(X))^2$$

Therefore by using,

$$E[X] = \frac{b+a}{2} = \mu,$$

we get,

$$V[X] = \frac{(b-a)^2}{12}$$

1.2 Normal distribution

Task: Specify the density function and the distribution function of a $N(\mu, \sigma^2)$ distributed random variable X. How can the tabulated values of the standard normal distribution function be used to determine values for a $N(\mu, \sigma^2)$ distributed random variable?

Solution:

The normal distribution is an extremely important continuous probability distribution. In probability theory, the normal distribution is also called the Gaussian distribution. There are two parameters in normal distribution, μ is the mean and lies between $-\infty$ and $+\infty$ and σ is the standard deviation.

The normal density function is given by:

$$f_x(x) = \tfrac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\tfrac{(x-\mu)^2}{2\sigma^2}\right\}, \quad X \in R$$

The normal distribution function is given by:

$$F_x(X) = \int\limits_{-\infty}^{x} f(x')dx'$$
$$F_x(X) = \int_{-\infty}^{x} \tfrac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\tfrac{(x-\mu)^2}{2\sigma^2}\right\} dx'$$

If X is a random variable that has a normal distribution with mean, μ and variance, σ^2,we write this equation as $X \sim N\ (\mu, \sigma^2)$. The standard normal distribution with mean 0 and variance 1 and is written as $Y \sim N$ (0,1). Its distribution function is usually denoted by $\phi(x)$.

$$\phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} \exp\left\{-\frac{x'^2}{2}\right\}$$

Let $X \sim N(\mu, \sigma^2)$ and therefore $\frac{(X-\mu)}{\sigma} \sim N(0,1)$ then

$$F_X(x) = P(\{\zeta : -\infty < X(\zeta) \leq x\})$$

$$F_X(x) = P(\{\zeta : -\infty < \frac{X(\zeta)-\mu}{\sigma} \leq \frac{x-\mu}{\sigma}\}) = \phi\left\{\frac{x-\mu}{\sigma}\right\}$$

The following graph represents normal distribution curves of various mean and variance values.

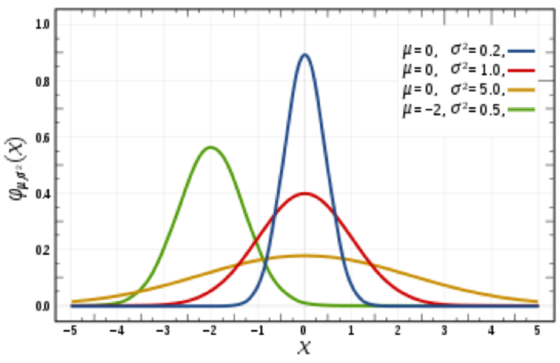


Figure 1.4: Normal Distribution Curve

1.3 Approximation of distribution function

Task: What are the advantages in estimating the distribution function instead of estimating the corresponding density function?

Solution:

Probability density estimation using histograms: The histogram partitions the set [0,1] into several bins and uses the count of the bin as a density estimate. The sample space is split up into bins and then the count of the samples that fall into each bin is taken and is divided by the total number of samples. Keeping the width of the bins fixed, as the number of data sampled increases, then by the law of large numbers, the relative frequency for each bin will converge on the bins probability. When the number of bins is too large, the quantity (bias) will be small while the second quantity (variance) will be large. When the number of bins is too small, the quantity (bias) will be large while the second quantity (variance) will be small. This is called over-smoothing. There exists a bias-variance trade off.

Probability distribution using histograms: Suppose we have one-dimensional samples $x_1 : x_2 ::: x_n$ with a cumulative distribution function F. Then we can define the empirical cumulative distribution function on `n` samples as,

$$F_n(x)^{'} = \frac{1}{n}\sum_{i=1}^{n} 1\{X_{(i)} \leq x\}$$

According to Glivenko-Cantelli Lemma, the empirical distribution converges uniformly to `F(x)`, namely

$$max_n|F_n(x)^{'} - F(x)| \longrightarrow 0$$

The maximum gap between the empirical CDF and true CDF goes. It means we can learn distributions just by collecting enough data.

1.4 Bivariate normal distribution

Task: Specify the density function $f_{x,y}(x, y)$ and the characteristic function $\varphi_{x,y}(x, y)$ of two bivariate normal distributed random variables X and Y with the expected values μ_x, μ_y, the variances ${\sigma_x}^2$, ${\sigma_y}^2$ and the correlation coefficient ρ. Outline the level curves of the density function $\{$*(x,y)*: $f_{x,y}(x, y) =$ const.$\}$ for $\rho = 0$ and $\rho = 1/2$. To what shape degenerate the level curves in case of $\rho = \pm 1$?

Solution: A bivariate distribution is a type of bivariate probability distribution and it represents the generalization of the normal distribution over two dimensions. A two dimensional normal distribution is called as bivariate normal distribution. Two random variables X and Y are said to be bivariate normal, if $aX + bY$ has a normal distribution for all a, b $\in$ R. The density function of a Bivariate Normal Distribution is,

$$f_{XY}(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} exp\left[-\frac{1}{2(1-\rho^2)}.\{(\frac{x-\mu_x}{\sigma_x})^2 - 2\rho\frac{(x-\mu_x)(y-\mu_y)}{\sigma_x\sigma_y} + (\frac{y-\mu_y}{\sigma_y})^2\right]$$

For all $x, y \in R : -1 < \rho < 1; \sigma_x > 0$ and $\sigma_y > 0$

The density function of the Standard Bivariate Normal Distribution can be given as:

$$F_{XY}(x,y) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left[-\frac{1}{2(1-\rho^2)}.\{x^2 - 2\rho xy + y^2\}\right]$$

where, $\sigma \in (-1, 1)$. If ρ=0, then we just say X and Y have the standard bivariate normal distribution.

The characteristic function of the bivariate normal distribution is defined as,

$$\phi_{XY}(t1, t2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{xy}(x,y)exp(j(t1x + t2y)dxdy$$

If the two random variables X and Y are independent, the characteristic function of the bivariate normal distribution given as,

$$\phi_{XY}(t1, t2) = \exp(j\mu_x t1)\exp\left[\frac{(\sigma_x^2/t1^2)}{2}\right]\exp(j\mu_y t2)\exp\left[\frac{(\sigma_x^2/t2^2)}{2}\right]$$

The level curves for the density function are shown in Figure 1.5.

The values of the parameters used in the density function is shown in the table.

μ_x	μ_y	σ_x	σ_x
1.4569	0.5012	0.9435	0.6797

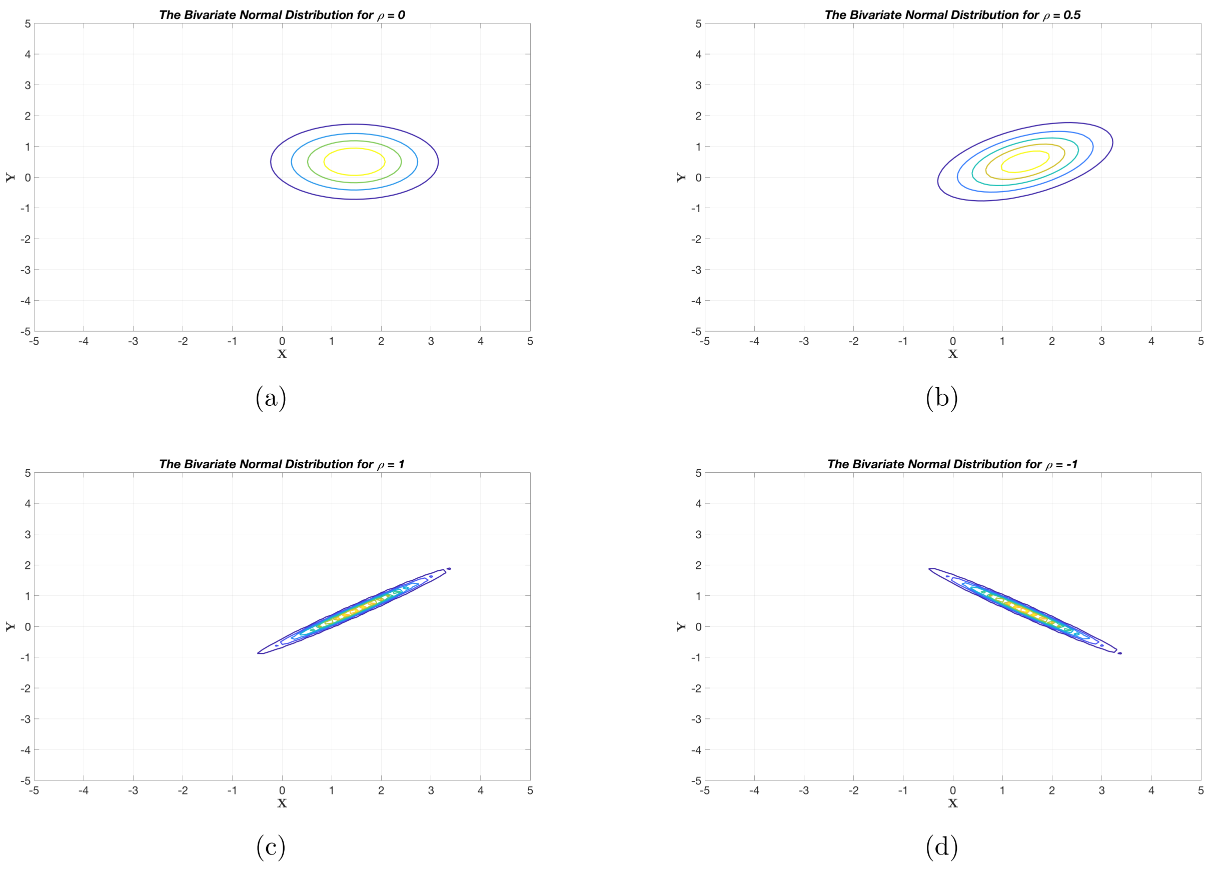


Figure 1.5: Level curve for different values of ρ

The first plot shows the case where the correlation ρ is equal to zero. This special case is called the circular normal distribution. Here, we have a perfectly aligned ellipse. As ρ increases, we can see that the curve becomes flattened in the perpendicular direction.

1.5 Monte-Carlo-method

Task: Develop a method for estimating the constant π by using on the interval [0, 1] uniformly distributed random numbers. (Note: Consider pairs of these random numbers like coordinates of random points in the plane. How large is approximately the relative frequency that a point is lying inside the unit circle?)

Solution: Monte Carlo simulation also known as MC simulation, is a method from Stochastics, in which a very large number of similar random experiments is the basis. It is an attempt made to numerically solve analytical problems that cannot be solved or only be solved with the help of probability theory. The constant π can be estimated using Monte Carlo method as follows.

1. We start shooting random points (x, y) within a unit square such that x, y belongs to [0 1]. The Figure 1.6 depicts a unit square along with a unit circle which is centred at (0, 0) and has radius 1.

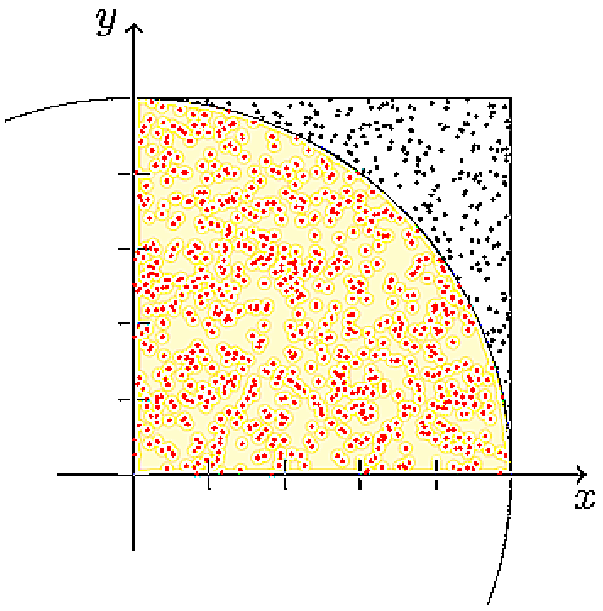


Figure 1.6: Monte Carlo Method (example)

2. The random points (x, y) that lie inside that unit circle are coloured pink and those that lie on the unit square outside the circle are coloured black.

3. We keep a track of the total number of points N_{total} and the number of points inside the quarter unit circle N_{circle}

4. If we divide N_{circle} by N_{total}, we should get a value that is approximate to the ratio of area of

the quarter circle to the area of the unit square.

$$\frac{N_{circle}}{N_{total}} = \frac{\textit{Area of quarter circle}}{\textit{Area of unit square}}$$

$$\frac{N_{circle}}{N_{total}} = \frac{\frac{1}{4}\pi r^2}{r^2} = \frac{1}{4}\pi$$

Therefore,

$$\pi = 4 * \frac{N_{circle}}{N_{total}}$$

5. Thus π can be estimated using the above equation. When we only have a small number of points, the estimation is not very accurate, but when we have hundreds of thousands of points, we get much closer to the actual value within around 2 decimal places of accuracy.

It may take a long period to determine the value of π using this experiment. As shown in the above figure, let the value of the radius of the circle be 1. We can calculate the distance from the origin using Pythagorean Theorem and the generated two random variables x and y. By knowing the value of the radius, the points hitting outside the circle can be easily differentiated from the points hitting within the circle.

To estimate the value of π in the interval [0,1] through the unit circle, two variables x and y has to be considered. Then the equilibrium can be given as,

$$\int_0^1 \int_0^{\sqrt{1-x^2}} 1 dy\ dx \approx K/N$$

where, K = number of points within the circle, N = sample numbers

Chapter 2

Exercises with Matlab

2.1 Mean value, variance and standard deviation

Task: Generate 1000 on the interval [0, 1] uniformly distributed random numbers. Display the numbers by using the MATLAB-instruction `plot.` Before generating the random numbers the initial value has to be set to 0 via the option `state` in the MATLAB function `rand`. Save the vector of random numbers in `dat1_1`. Calculate the mean value, variance and standard deviation of the random sequence in three different ways.

Using the MATLAB

1. loop `for ... end`,
2. functions `sum` and `length` and
3. functions `mean` and `cov`.

Compare the results. Take a look at the m-files `mean` and `cov` with the instruction type. **Solution:** We can create 1000 uniformly distributed random numbers by using the function `rand`. We have written separate functions in-order to have a clean code. First, the function `randomSequence` is written to generate uniformly distributed random numbers, then three separate functions called to calculate the mean value, variance and standard deviation in the mentioned ways. Lastly, the `compare` function aids to compare the result.

MATLAB code:

```
clear all
close all
clc

N = 1000;
y = randomSequence(N);
[m1,v1,sd1] = forLoop(y.n, y.x);
[m2,v2,sd2] = usingSumAndLength(y.x);
[m3,v3,sd3] = usingMeanAndConv(y.x);
compare(m1,m2,m3,v1,v2,v3,sd1,sd2,sd3);

function y = randomSequence(n)
disp('Generating uniformly distributed random numbers ...')
rand('state',0);
x = rand(1,n);
figure()
plot(x)
set(gca,'Title',text('String','Uniformly distributed random numbers',...
                     'FontAngle', 'italic', 'FontWeight', 'bold'),...
         'xlabel',text('String', 'random numbers', 'FontAngle','italic'),...
         'ylabel',text('String', 'random values', 'FontAngle','italic'), ...
         'FontSize',15)
save dat1_1;
y = load('dat1_1');
disp('Saving the random sequence in dat1_1')
disp('done')
end

function [m1,v1,sd1] = forLoop(n,x)
fprintf('\n Calculation using loop for...end')
m = 0;
for i = 1:n
    m = m + x(1,i);
end
m1 = m/n;
fprintf('\nMean: %f', m1);
v = 0;
for j = 1:n
    v = (v + (x(1,j)-m1)^2);
end
v1 = v/(n-1);
fprintf('\nVariance: %f', v1);
sd1 = sqrt(v1);
fprintf('\nStandard deviation: %f', sd1);
end

function [m2,v2,sd2] = usingSumAndLength(x)
fprintf('\n\n Calculation using sum and length functions')
Summation = sum(x);
L = length(x);
m2 = Summation/L;
fprintf('\nMean: %f', m2);
v2 = (sum((x-m2).^2))/(L-1);
fprintf('\nVariance: %f', v2);
sd2 = (sqrt(v2));
fprintf('\nStandard deviation: %f', sd2);
end

function [m3,v3,sd3] = usingMeanAndConv(x)
fprintf('\n\n Calculation using mean and conv functions')
m3 = mean(x);
```

```
v3 = cov(x);
sd3 = (sqrt(v3));
fprintf('\nMean: %f', m3);
fprintf('\nVariance: %f', v3);
fprintf('\nStandard deviation: %f', sd3);
end

function compare(m1,m2,m3,v1,v2,v3,sd1,sd2,sd3)
fprintf('\n\n Comparision of all three methods')
tolerance = 0.0001;
m = (m1+m2)/2;
v = (v1+v2)/2;
sd = (sd1+sd2)/2;
    if abs(m-m3) < tolerance
        fprintf('\nMean obtained from either of the methods are the same.')
    else
        fprintf('\nMean obtained for each method is different')
    end
    if abs(v-v3) < tolerance
        fprintf('\nVariance obtained from either of the methods are the same.')
    else
        fprintf('\nVariance obtained for each method is different')
    end
    if abs(sd-sd3) < tolerance
        fprintf('\nStandard deviation obtained from either of the methods are the same.')
    else
        fprintf('\nStandard deviation obtained for each method is different')
    end
end
```

Output: The plot of uniformly distributed random numbers over the interval [0,1] is depicted in Figure 2.1.

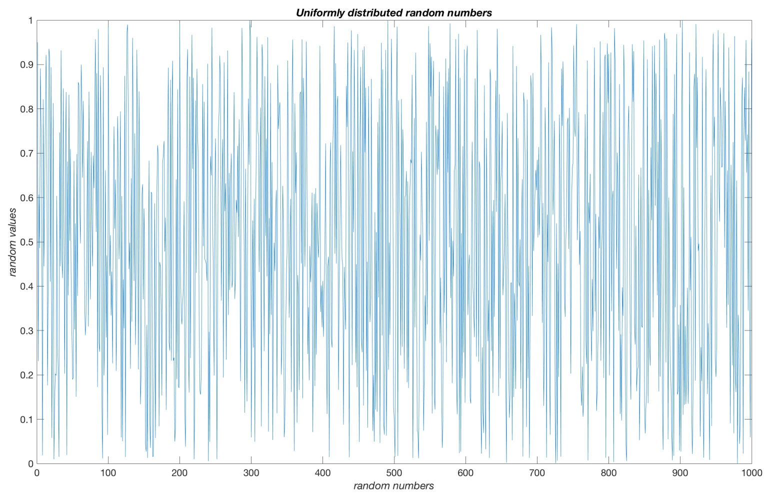


Figure 2.1: Uniformly distributed random variables in the interval [0,1]

The output of the code is shown below.

```
Generating uniformly distributed random numbers ...
Saving the random sequence in dat1_1
done

 Calculation using loop for...end
Mean: 0.502108
Variance: 0.083717
Standard deviation: 0.289339

 Calculation using sum and length functions
Mean: 0.502108
Variance: 0.083717
Standard deviation: 0.289339

 Calculation using mean and conv functions
Mean: 0.502108
Variance: 0.083717
Standard deviation: 0.289339

 Comparision of all three methods
Mean obtained from either of the methods are the same.
Variance obtained from either of the methods are the same.
Standard deviation obtained from either of the methods are the same.
```

Inference: As observed from the above output, the mean, variance and the standard deviation calculated using the three different methods yields the same result.

2.2 Density function and histogram

Task: Write a function `density`, that estimates the density function of a random variable from a number of observations by applying the histogram function `hist`. Use the uniformly distributed random sequence from `dat1_1`. Show the estimates as bar graph and as line graph. Compare the theoretical density function with the result by drawing the theoretical density also into the line graph figure. Repeat the experiment for 1000 standard normal distributed random numbers, where the initial value has to be set to 0. Save this random vector in `dat1_2`. (Note: The function `density` shall be applicable to any random sequences. Therefore take into account the interrelationship between the scaling and the length of the random sequence and the width of the interval.)

Solution: `density` function is written which allows the user to choose the number of `bins`. The uniform density function as well as the normal density function is estimated by using the function `hist`. Line graph of both theoretical and estimated values are plotted in-order to compare the two.

MATLAB code:

```
clear all
close all
clc

load dat1_1;
bins = input('Number of bins:');
[rho1,location1] = density(x,bins);
theoretical_uniform = ones(bins);
disp('getting line graph and bar graph for uniform density function...')
plotGraphs(rho1,location1,theoretical_uniform,'Uniform density function');
disp('done')
randn('state',0);
x1 = randn(1000,1);
save dat1_2 x1;
```

```
[rho2,location2] = density(x1,bins);

theoretical_normal = (1/(2*pi)^0.5)*exp(-0.5*((location2).^2));

disp('getting bar and line graph for standard normal density function...')

plotGraphs(rho2,location2,theoretical_normal, ...

    'Standard normal density function');

disp('done')


function [rho,location] = density(randomVariable,bins)

[height,location] = hist(randomVariable,bins);

delta = location(2)-location(1);

rho = height/(delta*length(randomVariable));

end


function plotGraphs(rho,location,theoretical,str)

figure()

bar(location,rho,'FaceColor',[0.9290, 0.6940, 0.1250], ...

    'EdgeColor','k','LineWidth',1)

set(gca,'Title',text('String',str,'FontAngle', 'italic', ...

    'FontWeight', 'bold'), ...

        'xlabel',text('String', 'range', 'FontAngle','italic'),...

        'ylabel',text('String', 'density','FontAngle','italic'), ...

        'FontSize',18)

hold on

grid on

plot(location,rho,location,theoretical,'--','LineWidth',2.5);

legend('Estimated value (bar graph)','Estimated value (line graph)', ...

    'Theoritical value');

hold off

end
```

Output: The estimated line and bar graph is plotted along with the theoretical graph for comparison.

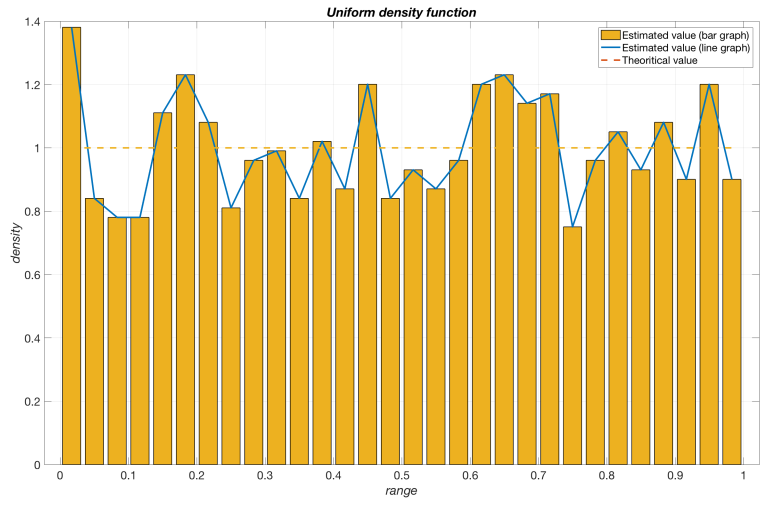


Figure 2.2: Uniform density function

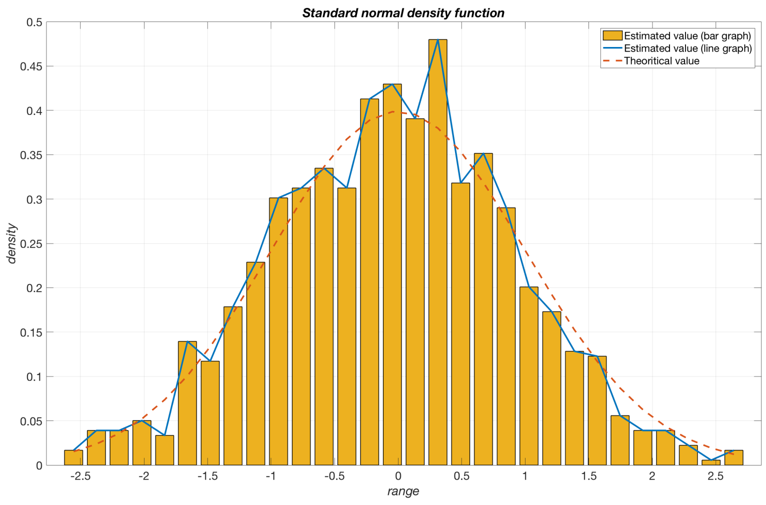


Figure 2.3: Normal density function

The output of the code is shown below.

```
Number of bins:30
getting line graph and bar graph for uniform density function...
done
getting bar and line graph for standard normal density function...
done
```

Inference: The values that are estimated using the histogram nearly coincide with the values estimated by the theoretical density function.

2.3 Distribution function and frequency

Task: Estimate the distribution function of the random sequences given in `dat1_1` and `dat1_2`. For this, generate with the MATLAB instruction `linspace` a rampe within the interval [0, 1] and use it for drawing the estimate. (Note: Sort the data.)

Solution: the function `linespace` is used to generate a ramp within a given interval. we used this function to estimate the distribution function of random sequences that are stored in `dat_1.mat` and texttttdat_2.mat. The previously stored data needed to be sorted. In this case, we used the function `sort` to sort the data in increasing order.

MATLAB code:

```
clear all
close all
clc

if exist('dat1_1.mat','file')
    load dat1_1;
    fprintf('Dimension of x, (N) = %d',length(x));
else
    warning('dat1_1 doesnot exist')
    return
end
if exist('dat1_2.mat','file')
    load dat1_2;
    fprintf('\nDimension of x1, (N1) = %d',length(x1));
else
    fprintf('\n')
    warning('dat1_2 doesnot exist')
    return
 end
y = linspace(0,1,length(x));
```

```
y1 = linspace(0,1,length(x1));
z = sort(x);
z1 = sort(x1);
figure()
plot(z,y,'LineWidth',2)
set(gca,'Title',text('String','Distribution function',...
                    'FontAngle', 'italic', 'FontWeight', 'bold'),...
        'xlabel',text('String', '$\mathbf{X}$', 'Interpreter', 'latex'),...
        'ylabel',text('String', '$\mathbf{F(X)}$', 'Interpreter', 'latex'), ...
        'FontSize',15)
hold on
plot(z1,y1,'LineWidth',2)
xlabel('X')
ylabel('F(X)')
grid on
legend('Uniform distribution','Normal distribution')
```

Output: `dat_1.mat` had the values for uniform distribution while `dat_2.mat` consisted of values for the normal distribution. The plot of both the distribution is shown in Figure 2.4.

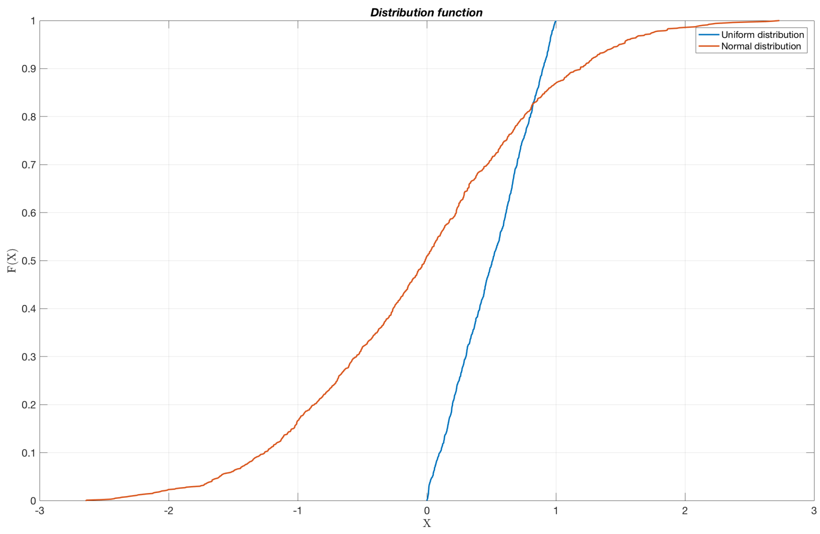
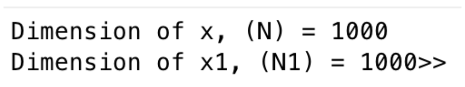


Figure 2.4: Distribution function of random sequences

In-order to get affirmation of the previously stored data, `exist` function was used to check the presence of the files and the number of random numbers were printed.



```
Dimension of x, (N) = 1000
Dimension of x1, (N1) = 1000>>
```

Inference: The plotted graph shows that the previously stored data is correct.

2.4 Generation of a bivariate normal distribution

Task: Set the initial value of `randn` to 0. Generate through `z1=randn(1000,2)` a matrix of normal distributed random numbers. Multiply `z1` from the right with the matrix `D = [1 .5; 0 .5]`, thus `z2 = z1 * D`. Separate the result through `x = z2(:,1)` and `y = z2(:,2)` into two vectors and add 1.5 to each element of `x` and 0.5 to each element of y. Store the vectors by using the MATLAB instruction `save dat1_3 x y`.

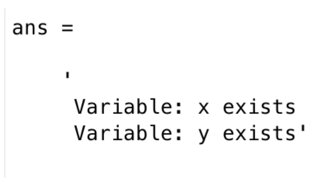
Solution: The `dat1_3` was saved by following the instructions step-by-step as mentioned in the task. To check for validation, `whos` function was used and the variable names were printed.

MATLAB code:

```
clear all;
close all;
clc

randn('state',0);
z1 = randn(1000,2);
D = [1 0.5; 0 0.5];
z2 = z1 * D;
x = z2(:,1) + 1.5;
```

```
y = z2(:,2) + 0.5;
save dat1_3 x y;
if exist('dat1_3.mat','file')
    load dat1_3;
    details = whos(matfile('dat1_3.mat'));
    sprintf('\nVariable: %1s exists',details.name)
else
    fprintf('\n')
    warning('Some error occured')
    return
 end
```

Output: The output after execution of the code is shown below.



```
ans =

    '
     Variable: x exists
     Variable: y exists'
```

Inference: Variable `x` and variable `y` exists. Thus we can say that we have successfully implemented the mentioned task.

2.5 Bivariate normal distribution

Task: Load `dat1_3` that includes the two vectors `x` and `y` of length 1000 as samples of the two bivariate normally distributed random variables `X` and `Y`. Estimate for this dataset μ_x, μ_y, σ_x^2, σ_y^2 and ρ by using the sample cross- and covariance function. Calculate the bivariate density function $f_{x,y}(x, y)$ with the estimated parameters and display it as a 3dimensional and a level curve graphic. (Note: Use the MATLAB-functions mesh and contour.)

Solution: First, a function `getParameters` was written in-order to get the values of μ_x, μ_y, σ_x^2, σ_y^2 and ρ. Later, these values were required to calculate the Bivariate normal distribution. The 3D view of the distribution was plotted using `mesh`.

MATLAB code:

```
clear all
close all
clc

load dat1_3;
a = -5:0.125:5;
[X,Y] = meshgrid(a);
F_xy = bivariateFunction(x,y,X,Y);
figure()
mesh(X,Y,F_xy,'LineWidth',1)
set(gca,'Title',text('String','Bivariate Normal Distribution',...
                     'FontAngle', 'italic', 'FontWeight', 'bold'),...
         'xlabel',text('String', '$\mathbf{X}$', 'Interpreter', 'latex'),...
         'ylabel',text('String', '$\mathbf{Y}$', 'Interpreter', 'latex'), ...
         'zlabel',text('String', 'density', 'FontAngle', 'Italic', ...
         'FontWeight', 'bold'),'FontSize',15)
grid on
figure()
contour(X,Y,F_xy,'LineWidth',2.5)
set(gca,'Title',text('String','The Contour of The Bivariate Normal Distribution',...
                     'FontAngle', 'italic', 'FontWeight', 'bold'),...
         'xlabel',text('String', '$\mathbf{X}$', 'Interpreter', 'latex'),...
         'ylabel',text('String', '$\mathbf{Y}$', 'Interpreter', 'latex'), ...
         'FontSize',15)
grid on

function F_xy = bivariateFunction(x,y,X,Y)
[mu_x,mu_y,sigma_x,sigma_y,rho] = getParameters(x,y);
A = 1/(2.*pi.*sigma_x.*sigma_y.*sqrt(1-rho.^2));
B = 1/(2.*(1-rho.^2));
```

```
C = ((X-mu_x)/sigma_x).^2;
D = (2.*rho.*(X-mu_x).*(Y-mu_y))/(sigma_x.*sigma_y);
E = ((Y-mu_y)/sigma_y).^2;
F_xy = A.*exp(-B.*(C-D+E));
end

function [mu_x,mu_y,sigma_x,sigma_y,rho] = getParameters(x,y)
mu_x = mean(x);
mu_y = mean(y);
covarianceMatrix = cov(x,y);
sigma_x = sqrt(covarianceMatrix(1,1));
sigma_y = sqrt(covarianceMatrix(2,2));
rho = (covarianceMatrix(1,2) / ...
    sqrt(covarianceMatrix(1,1)*covarianceMatrix(2,2)));
end
```

Output: Figure 2.5 is the result from the execution of the code. ρ found by using the formula for these data sets was 0.6529. Hence it is a little more flattened in the perpendicular direction as compared to when ρ is 0. At 0, the graph is perfectly symmetric bell shaped and parallel to the horizontal line.

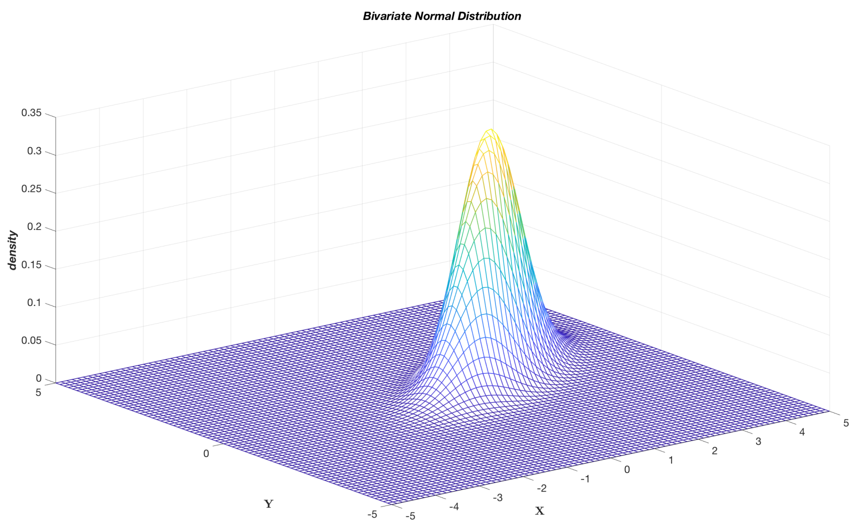


Figure 2.5: 3 dimensional view of Bivariate normal distribution

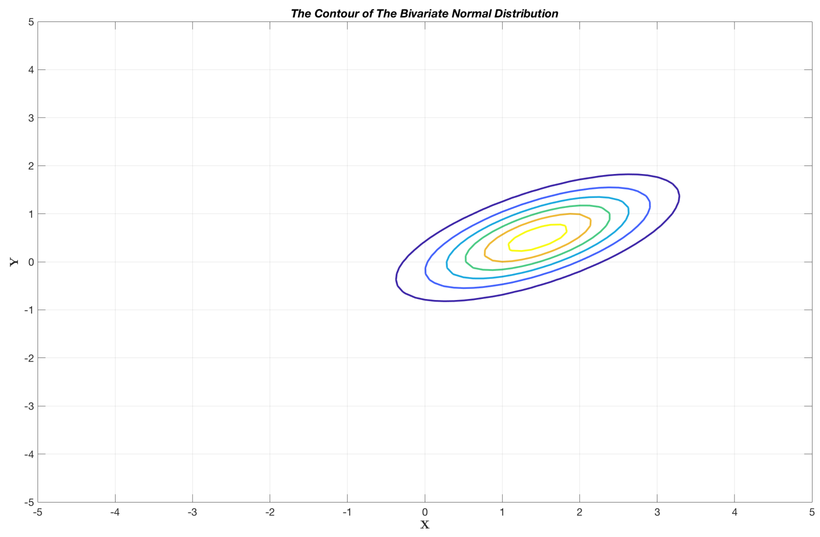


Figure 2.6: Contour of the Bivariate normal distribution

Inference: As ρ increases, that bell-shaped curve becomes flattened on the 45 degree line. At ρ equals 0.65, we can see that the curve extends out towards minus 3 and plus 3 and becomes flattened in the perpendicular direction.

2.6 Monte-Carlo-Method for approximation of π

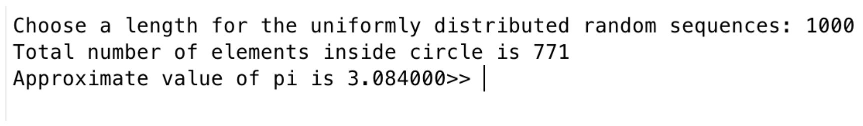
Task: Generate two on the interval [0, 1] uniformly distributed random sequences `x` and `y` of length 1000. The initial value has to be set to 0 again. How many points lie within the unit circle? Estimate therewith the constant π .

Solution: First, `x` and `y` in the range of [0,1] are created using `rand`. Further, the code is written to find the number of points within a circle of radius 1. We have allowed the users to choose the length of the uniformly distributed random sequence. We aim to find the relationship between the actual value of π and the length of the sequence chosen.

MATLAB code:

```
clear all

close all

clc


rand('state',0)

N = input('Choose a length for the uniformly distributed random sequences: ');

randomValues = rand(N,2);

x = randomValues(:,1);

y = randomValues(:,2);

no_insideCircle = 0;

for i = 1:N

    radius = sqrt(x(i,1).^2 +y(i,1).^2);

    if radius <= 1

        no_insideCircle = no_insideCircle + 1;

    end

end

fprintf('Total number of elements inside circle is %d',no_insideCircle)

approxPi = 4.*(no_insideCircle/N);

fprintf('\nApproximate value of pi is %f',approxPi);
```

Output: The output of the code is shown below. We considered the length of the sequence to be 1000 as mentioned in the task.



```
Choose a length for the uniformly distributed random sequences: 1000
Total number of elements inside circle is 771
Approximate value of pi is 3.084000>> |
```

We further wrote another program to find how the length of the sequence affects the value of π.

```
clear

N_all = round(10.^(1:.5:7));

r = 1;

errors = NaN(size(N_all));

for k = 1:numel(N_all)

    N = N_all(k);
    x = rand(1,N);
    y = rand(1,N);
    r2 = x.^2 + y.^2;
    area = mean(r2<=r^2)*4;
    error_all(k) = pi-area;
end
semilogx(N_all, error_all, 'o-','LineWidth',2);
set(gca,'xlabel',text('String', 'length of random sequence', ...
    'FontAngle','italic'),'ylabel',text('String', ...
    'error (true value - estimated value)', 'FontAngle','italic'), ...
    'FontSize',15)
grid
```

We notice from Figure 2.7 that as the length of the sequence increases, we get a better approximation of π.

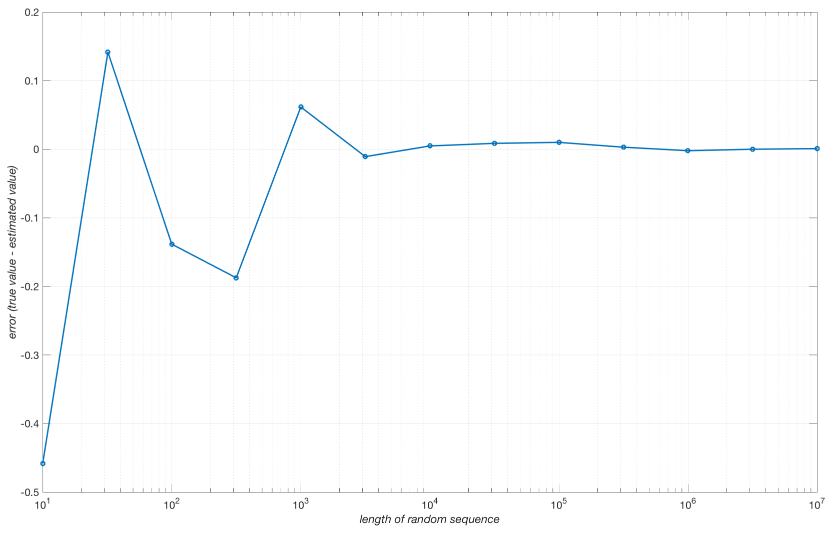


Figure 2.7: Length of random sequence versus the error value

Inference: From Figure 2.7, it is proved that as we increase the length of the sequence from 1000 to 100000 (100 times), we have a much better estimation of π.

Bibliography

[1] Prof. Dr.-Ing. Dieter Kraus, *"Stochastic Signals and Systems- Probability Theory* lecture notes.

[2] http://mathworld.wolfram.com/UniformDistribution.html

[3] https://onlinecourses.science.psu.edu/stat505/node/33/