MATLAB Simulation Project 2

The Gaussian approximation

Probability Engineering

Date: Spring 1403

Master: PhD. A Zaimbashi

Written By:
Alireza Sotoodeh (401412056)

Shahid Bahonar
University of Kerman

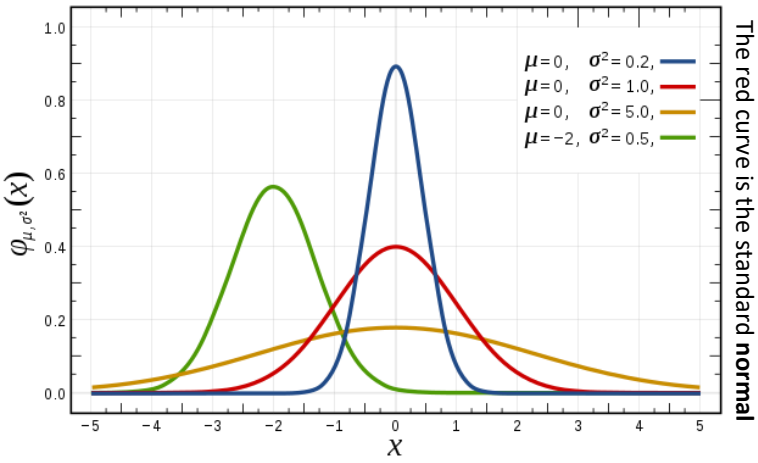
Alireza Sotoodeh Master: PhD. A Zaimbashi

Introduction to Gaussian Distribution

In probability theory and statistics, a normal distribution or Gaussian distribution is a type of continuous probability distribution for a real-valued random variable. The general form of its probability density function (**PDF**) is:

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

In statistics, the Gaussian distribution, also known as the normal distribution, is defined by two parameters: the mean (**μ**) and the standard deviation (**σ**). The mean (**μ**) represents the expectation, median, and mode of the distribution. The standard deviation (**σ**) is a measure of the amount of variation or dispersion in the set of values. The square of the standard deviation (**σ²**) is known as the variance. A random variable that follows a Gaussian distribution is said to be **normally distributed** and is referred to as a **normal deviate**.



The red curve is the standard **normal**

Normal distributions are important in statistics and are often used in the natural and social sciences to represent real-valued random variables whose distributions are not known. Their importance is partly due to the central limit theorem. It states that, under some conditions, the average of many samples (observations) of a random variable with finite mean and variance is itself a random variable—whose distribution converges to a normal distribution as the number of samples increases. Therefore, physical quantities that are expected to

be the sum of many independent processes, such as measurement errors, often have distributions that are nearly normal.

Moreover, Gaussian distributions have some unique properties that are valuable in analytic studies. For instance, any linear combination of a fixed collection of independent normal deviates is a normal deviate. Many results and methods, such as propagation of uncertainty and least squares parameter fitting, can be derived analytically in explicit form when the relevant variables are normally distributed.

The standard normal distribution, also known as the unit normal distribution, is a special case of the normal distribution. It occurs when the mean μ = 0 and the standard deviation σ = 1. The probability density function, which describes the distribution, is used in this case:

Notation	$\mathcal{N}(\mu, \sigma^2)$
Parameters	$\mu \in \mathbb{R}$ = mean (location) $\sigma^2 \in \mathbb{R}_{>0}$ = variance (squared scale)
Support	$x \in \mathbb{R}$
PDF	$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$
CDF	$\Phi\left(\frac{x-\mu}{\sigma}\right) = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right]$
Quantile	$\mu + \sigma\sqrt{2}\,\operatorname{erf}^{-1}(2p-1)$
Mean	μ
Median	μ
Mode	μ
Variance	σ^2
MAD	$\sigma\sqrt{2/\pi}$
Skewness	0
Excess kurtosis	0
Entropy	$\frac{1}{2}\log(2\pi e\sigma^2)$
MGF	$\exp(\mu t + \sigma^2 t^2/2)$
CF	$\exp(i\mu t - \sigma^2 t^2/2)$
Fisher information	$\mathcal{I}(\mu, \sigma) = \begin{pmatrix} 1/\sigma^2 & 0 \\ 0 & 2/\sigma^2 \end{pmatrix}$ $\mathcal{I}(\mu, \sigma^2) = \begin{pmatrix} 1/\sigma^2 & 0 \\ 0 & 1/(2\sigma^4) \end{pmatrix}$
Kullback–Leibler divergence	$\frac{1}{2}\left\{\left(\frac{\sigma_0}{\sigma_1}\right)^2 + \frac{(\mu_1-\mu_0)^2}{\sigma_1^2} - 1 + \ln\frac{\sigma_1^2}{\sigma_0^2}\right\}$
Expected shortfall	$\mu - \sigma\frac{\frac{1}{\sqrt{2\pi}} e^{\frac{-\left(q_p\left(\frac{X-\mu}{\sigma}\right)\right)^2}{2}}}{1-p}$[1]

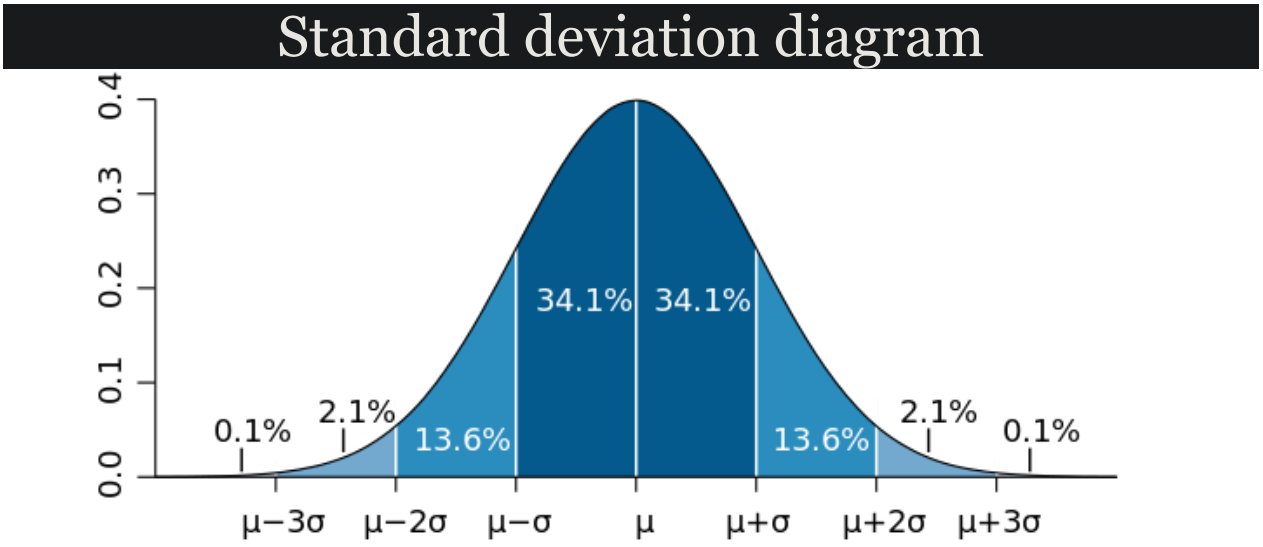
$$\varphi(z) = \frac{e^{-z^2/2}}{\sqrt{2\pi}}.$$

The variable ***Z*** has a mean of 0 and a variance and standard deviation of 1. The density $\varphi(z)$ has its peak $1/\sqrt{2\pi}$ at z = 0 and inflection points at z = +1 and z= -1.

Notation:

The probability density of the standard Gaussian distribution (also known as the standard normal distribution, which has a mean of zero and a variance of one) is often represented by the Greek letter phi (ϕ). The alternative form of the Greek letter phi, represented as φ, is also frequently used.

The normal distribution is often referred to as N(μ, σ^2) or $\mathcal{N}(\mu, \sigma^2)$. Thus, when a random variable X is normally distributed with mean μ and standard deviation σ, one may write X ~ N(μ, σ^2) or X ~ $\mathcal{N}(\mu, \sigma^2)$.



For the normal distribution, the values less than one standard deviation away from the mean account for 68.27% of the set; while two standard deviations from the mean account for 95.45%; and three standard deviations account for 99.73%.

Source: en.wikipedia.org

Alireza Sotoodeh Master: PhD. A Zaimbashi

MATLAB simulation project 1_code this project

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%{
Autor: Alireza Sotoodeh(Student ID:401412056)
Subject: Gaussian Random Variable Generation
%}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
clc
close all
%-------------------------------------------------------------------------
n=1000;                % Number of samples (The first main variable)
mu=10;                   % Mean (The second Main variable)
main_var = 3;              % variance = sigma^2
sigma=sqrt(main_var);        % Standard deviation (sqrt of variance)
%-------------------------------------------------------------------------
xlim1=0;                  %if xlim1 and xlim2 == 0 then no resizing for plot
xlim2=25                  %if xlim1 and xlim2 == 0 then no resizing for plot
ylim1=0;                  %if ylim1 and ylim2 == 0 then no resizing for plot
ylim2=0;                  %if ylim1 and xlim2 == 0 then no resizing for plot
h = 0.003;               % Distance between two text
Text_shift= 0;           % Position for shifting the text to right and left
Steps_for= n/10 ;          %for loop plotting mean and variance plots
%-------------------------------------------------------------------------
% We can generate a vector of random numbers by using: randn(1, n)
X = randn(1, n); % Generate n samples from standard normal distribution
mean_X = mean(X); % Calculate the sample mean
var_X = var(X); % Calculate the sample variance
disp(['Mean of X (approx 0): ', num2str(mean_X)]); % For X
disp(['Variance of X (approx 1): ', num2str(var_X)]);
%-------------------------------------------------------------------------
Y = sigma * X + mu; % Transform X
mean_Y = mean(Y); % Calculate the sample mean of Y
var_Y = var(Y); % Calculate the sample variance of Y
disp(['Mean of Y (approx ', num2str(mu), '): ', num2str(mean_Y)]); % For Y
disp(['Variance of Y (approx ', num2str(main_var), '): ',
num2str(var_Y)]);
%-------------------------------------------------------------------------
% Calculating and ploting histogram
num_bins = 100; % Number of bins for the histogram
[counts, edges] = histcounts(Y, num_bins, 'Normalization', 'pdf');
centers = (edges(1:end-1) + edges(2:end)) / 2; % Calculate bin centers
bar(centers, counts, 'FaceColor', [0.7 0.7 0.7]); % Plot histogram
f = @(y) 1/sqrt(2*pi*sigma^2) * exp(-(y-mu).^2 / (2*sigma^2));% Plot
hold on;
fplot(f, [min(Y) max(Y)], 'LineWidth', 2);
%-------------------------------------------------------------------------
%can change the length of the horizontal and vertical of the graph!
if ~(xlim1 == 0 && xlim2 == 0)
    xlim([xlim1, xlim2]);
end
if ~(ylim1 == 0 && ylim2 == 0)
    ylim([ylim1, ylim2]);
end
%-------------------------------------------------------------------------
%{
Display text and lines for different values!
%}
text_pos = max(Y)+ Text_shift;
% Display the Number of samples (The first main variable)
text(text_pos, max(counts)-(1*h), ['Number of samples: ', num2str(n)]);
% Display the main variance (sigma^2) on the plot as vertical lines
line([sigma^2 sigma^2], ylim, 'Color', 'k', 'LineStyle', '-');
text(text_pos, max(counts)-(2*h), ['Main Variance: ', num2str(sigma^2)]);
% Display the main mean (mu) on the plot as vertical lines
line([mu mu], ylim, 'Color', 'k', 'LineStyle', '-.');
text(text_pos, max(counts)-(3*h), ['Main Mean: ', num2str(mu)]);
% Display the mean and variance of X on the plot as vertical lines
line([mean_X mean_X], ylim, 'Color', 'r', 'LineStyle', '--');
text(text_pos, max(counts)-(4*h), ['Mean of X (\approx 0): ', ...
    num2str(mean_X)]);line([mean_X+sqrt(var_X) mean_X+sqrt(var_X)], ...
    ylim,'Color', 'b','LineStyle', '--');
text(text_pos, max(counts)-(5*h), ['Variance of X (\approx 1): ', ...
    num2str(var_X)]);
% Display the mean and variance of Y on the plot as vertical lines
line([mean_Y mean_Y], ylim, 'Color', 'g', 'LineStyle', '--');
text(text_pos, max(counts)-(6*h), ['Mean of Y(\approx ', ...
    num2str(mu), '): ',num2str(mean_Y)]);
line([var_Y var_Y], ylim, 'Color', 'm', ...
    'LineStyle', '--');
text(text_pos, max(counts)-(7*h), ['Variance of Y(\approx ', ...
    num2str(main_var), '): ', num2str(var_Y)]);
```

The rest of code ...

```
%-------------------------------------------------------------------------
%{
Display -1sigma and +1sigma on the plot as vertical lines (where 68% data
exist
%}
line([mean_Y-sigma mean_Y-sigma], ylim, 'Color', 'b', 'LineStyle', '-');
text(text_pos, max(counts)-(8*h), ['$-1\sigma$: ', ...
    num2str(mean_Y-sigma)],'Interpreter', 'latex');
line([mean_Y+sigma mean_Y+sigma], ylim, 'Color', 'b', 'LineStyle', '-');
text(text_pos, max(counts)-(9*h), ['$+1\sigma$: ', ...
    num2str(mean_Y+sigma)], 'Interpreter', 'latex');
hold off;
%-------------------------------------------------------------------------
h_legend = legend('Histogram','Gaussian', 'Main Variance', 'Main Mean',...
'Mean of X', 'Variance of X:', 'Mean of Y', 'Variance of Y:', ...
'$-1\sigma$','$+1\sigma$');
set(h_legend,'Interpreter','latex');
ylabel('pdf')
xlabel('y')
set(get(gca,'XLabel'),'Interpreter','latex', 'FontSize',10)
set(get(gca,'YLabel'),'Interpreter','latex', 'FontSize',10)
set(get(gca,'legend'),'Interpreter','latex', 'FontSize',9)
%-------------------------------------------------------------------------
%{
Add additional figures to show the value of mean of X converges to 0 and
variance of X converges to 1 for every n and also mean of Y to main mean
(for every n) and variance of Y to number of sample (for every n) using
for loop
%}
n_values = 1:Steps_for:n; % Vector of sample sizes
mean_X_values = zeros(size(n_values)); %  to store mean of X
var_X_values = zeros(size(n_values)); %  to store variance of X
mean_Y_values = zeros(size(n_values)); %  to store mean of Y
var_Y_values = zeros(size(n_values)); %  to store variance of Y

for i = 1:length(n_values)
    n_i = n_values(i); % Current sample size
    X_i = randn(1, n_i); % Generate n_i samples
    Y_i = sigma * X_i + mu; % Transform X_i
    mean_X_values(i) = mean(X_i); % store the sample mean of X_i
    var_X_values(i) = var(X_i); % store the sample variance of X_i
    mean_Y_values(i) = mean(Y_i); %  and store the sample mean of Y_i
    var_Y_values(i) = var(Y_i); %  store the sample variance of Y_i
end

% Plot the mean and variance of X and Y as functions of n
figure;
subplot(2, 2, 1);
plot(n_values, mean_X_values);
title('Mean of X vs. n');
xlabel('n');
ylabel('Mean of X');
text(max(n_values)*0.7, max(mean_X_values)*0.7, ['Converges to 0']);

subplot(2, 2, 2);
plot(n_values, var_X_values);
title('Variance of X vs. n');
xlabel('n');
ylabel('Variance of X');
text(max(n_values)*0.7, max(var_X_values)*0.7, ['Converges to 1']);

subplot(2, 2, 3);
plot(n_values, mean_Y_values);
title('Mean of Y vs. n');
xlabel('n');
ylabel('Mean of Y');
text(max(n_values)*0.7, max(mean_Y_values)*0.7, ['Converges to ', ...
    num2str(mu)]);

subplot(2, 2, 4);
plot(n_values, var_Y_values);
title('Variance of Y vs. n');
xlabel('n');
ylabel('Variance of Y');
text(max(n_values)*0.7, max(var_Y_values)*0.7, ['Converges to ', ...
    num2str(main_var)]);
```

be continued ...

Appendix: MATLAB Code Used for This Project

Note:

Instead of using "Hist," I have used the bar function. However, I encountered difficulty in finding a solution due to the change from displaying numbers to displaying values on the horizontal bar of the histogram!

The rest of code …

```
%-------------------------------------------------------------------------
%{
writing other tests for understanding better based on the internet and A.I
%}

%{
 Test for Normality: You can use a statistical test like the Shapiro-Wilk
 test or the Anderson-Darling test to check if your generated random
variable follows a normal distribution.
%}
[h,p] = adtest(Y); % Anderson-Darling test
if h == 0
    disp(['The generated random variable follows a normal ' ...
        'distribution (Anderson-Darling test).']);
else
    disp(['The generated random variable does not follow a ' ...
        'normal distribution (Anderson-Darling test).']);
end


%{
Test for Mean: You can use a t-test to check if the mean of your
generated random variable is equal to the expected mean (mu), and an
 F-test to check if the variance is equal to the expected
variance (main_var).
%}
[h,p] = ttest(Y, mu); % t-test
if h == 0
    disp(['The mean of the generated random variable is ' ...
        , num2str(mu), ' (t-test).']);
else
    disp(['The mean of the generated random variable is not ' ...
        , num2str(mu), ' (t-test).']);
end
[h,p] = vartest(Y, main_var); % F-test
if h == 0
    disp(['The variance of the generated random variable is ' ...
        , num2str(main_var), ' (F-test).']);
else
    disp(['The variance of the generated random variable is not ' ...
        , num2str(main_var), ' (F-test).']);
end

%{
Test for Independence: You can use the autocorrelation function to check
if the generated random numbers are independent.
Yes, for a sequence of independent and identically distributed (i.i.d.)
random variables, the autocorrelation should be near 0 for all lags
except 0.
%}
lags = 0:100;
autocorr_Y = zeros(size(lags));
for i = lags
    autocorr_Y(i+1) = mean((Y(1:end-i)-mean_Y).*(Y(i+1:end)-mean_Y));
end
autocorr_Y = autocorr_Y / autocorr_Y(1);
figure;
stem(lags, autocorr_Y);
title('Autocorrelation of Y(should be near 0)');
xlabel('Lag');
ylabel('Autocorrelation');
```

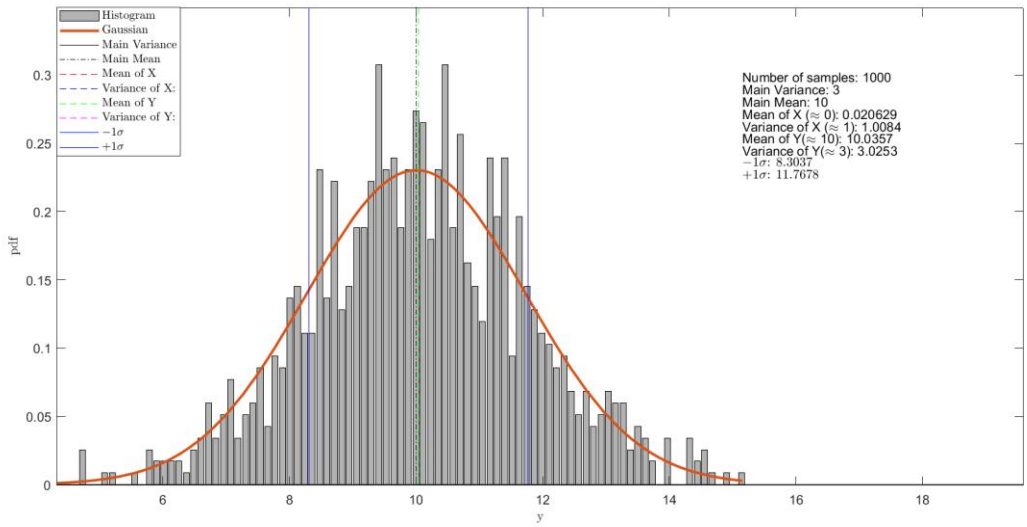
Note:

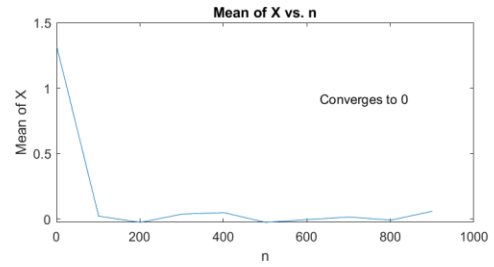
On this page, I have included additional test code for this distribution. (will show the result in the command window)

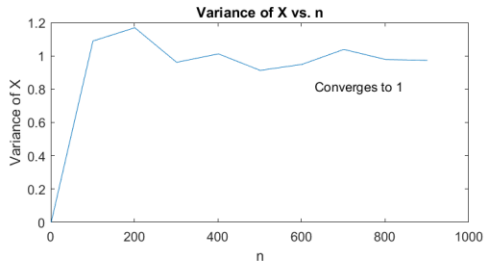
Alireza Sotoodeh Master: PhD. A Zaimbashi

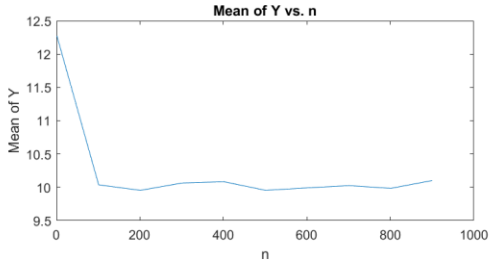
1. Prove that the random rate and variance above are zero and one, respectively.

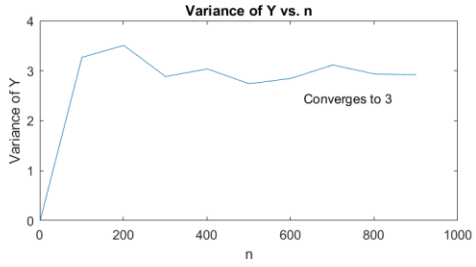
In order to do so we only need to run the code and check it for different values of N, mu and main_var! (for each example, the code generates 3 graph)



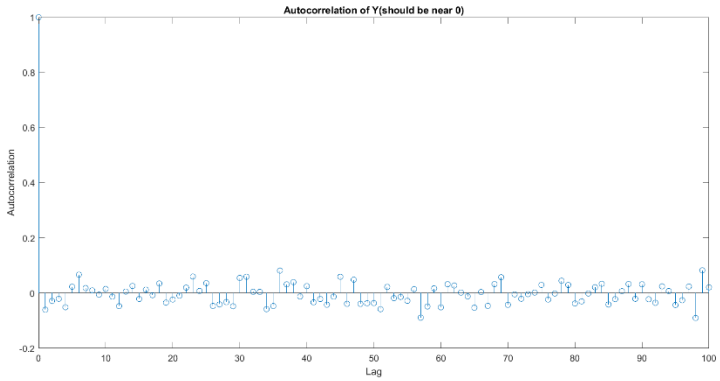




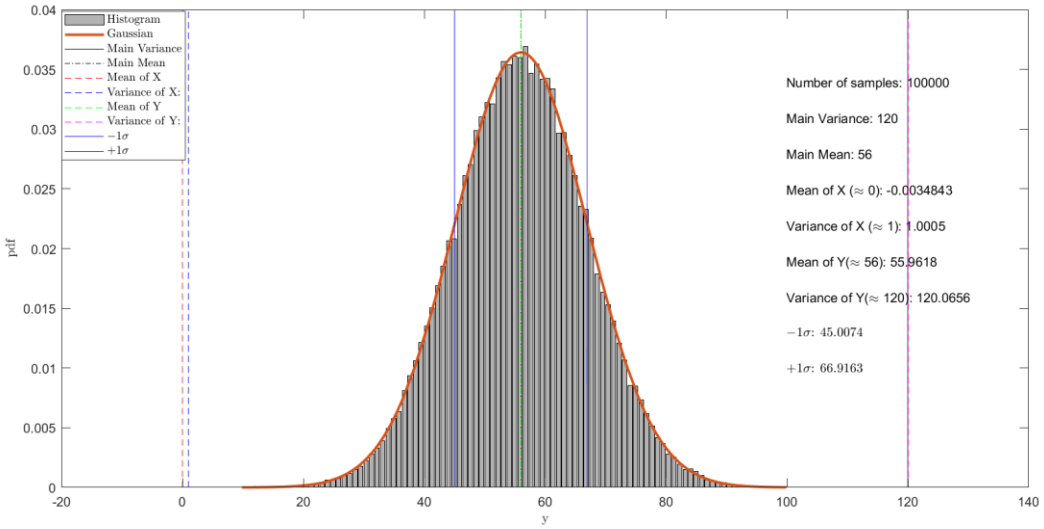


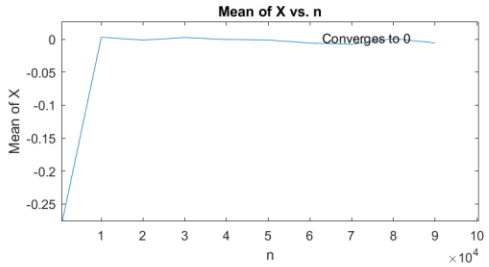


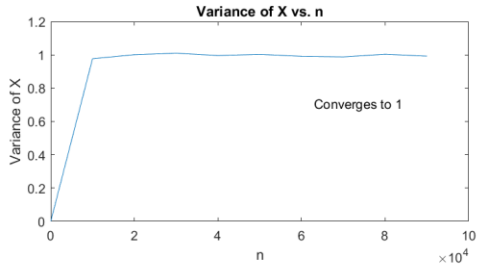
Converges to 10

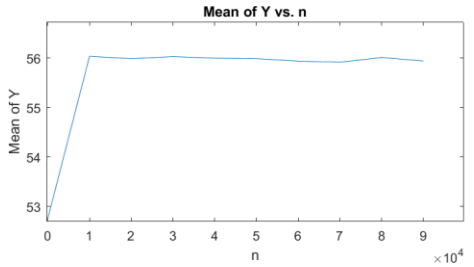


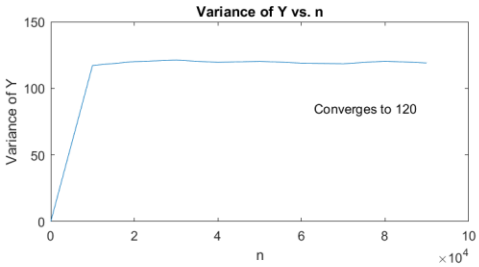
Alireza Sotoodeh Master: PhD. A Zaimbashi

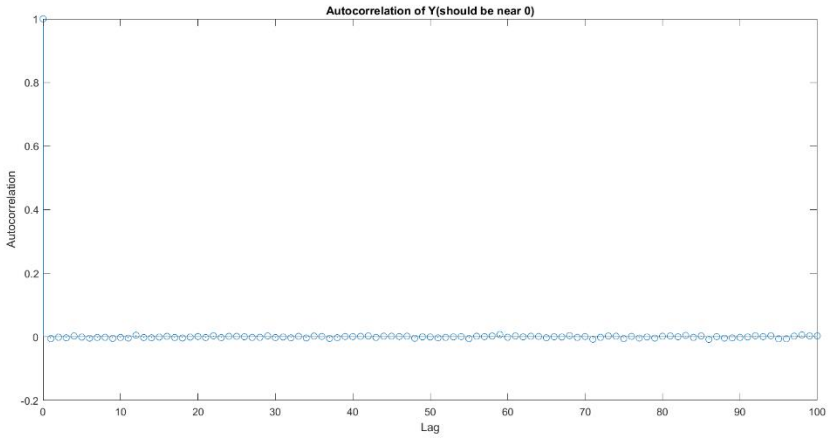




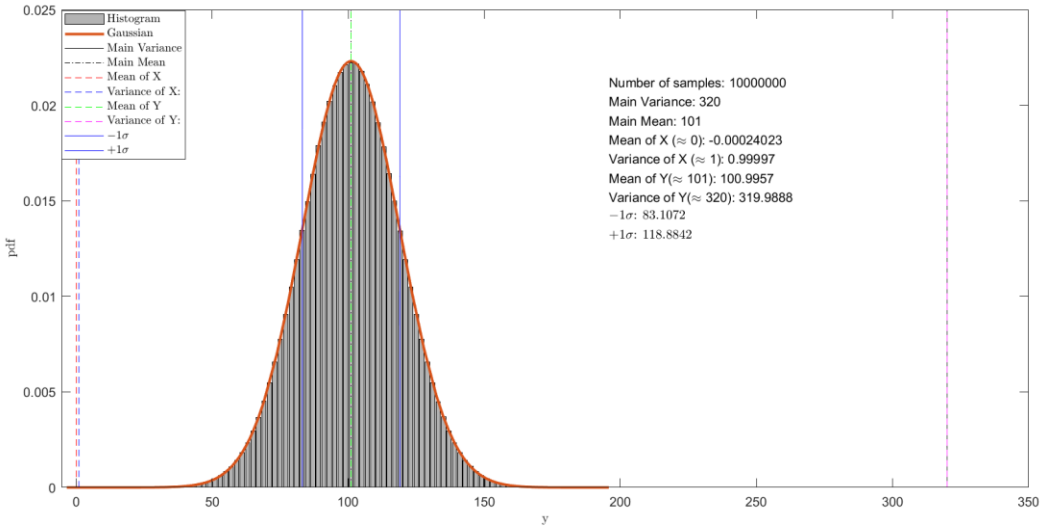


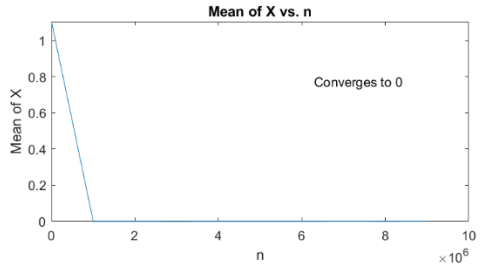


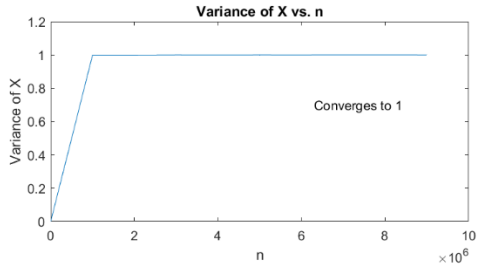


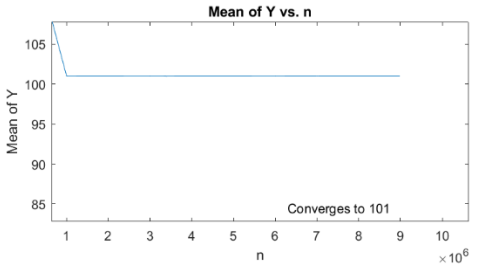


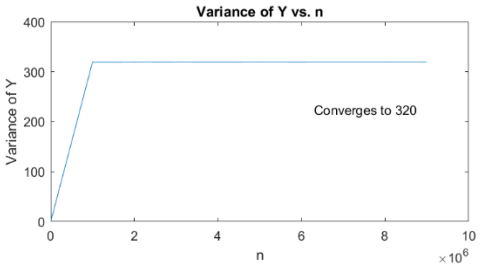
Alireza Sotoodeh Master: PhD. A Zaimbashi

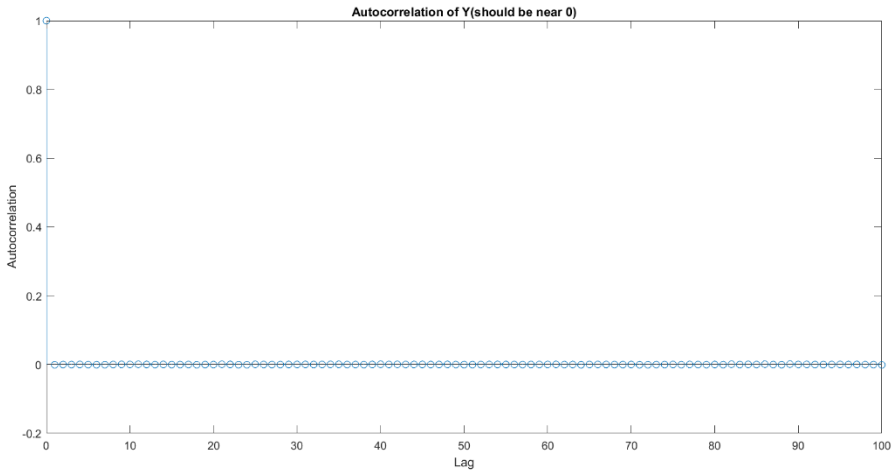




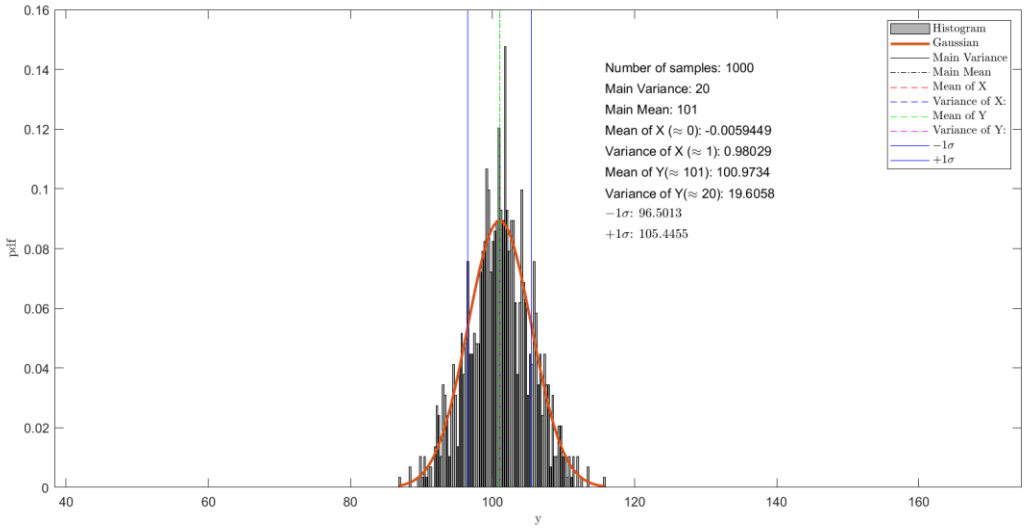


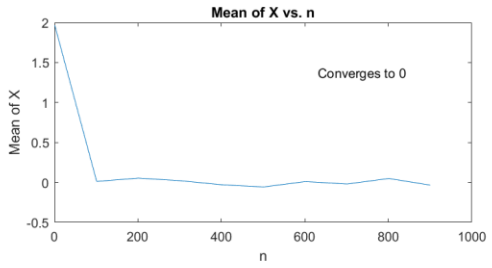


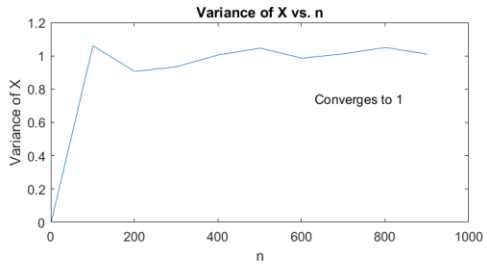


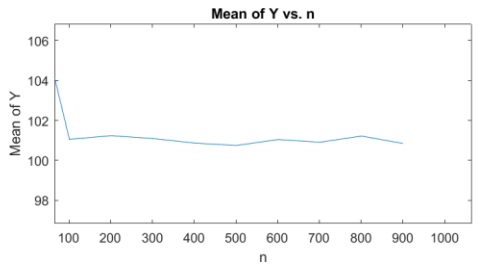


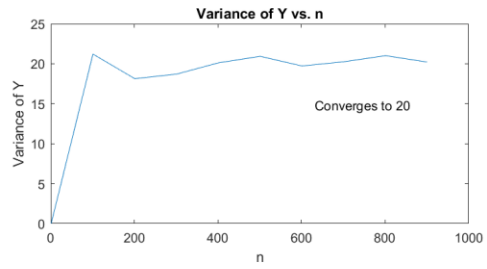
Alireza Sotoodeh Master: PhD. A Zaimbashi

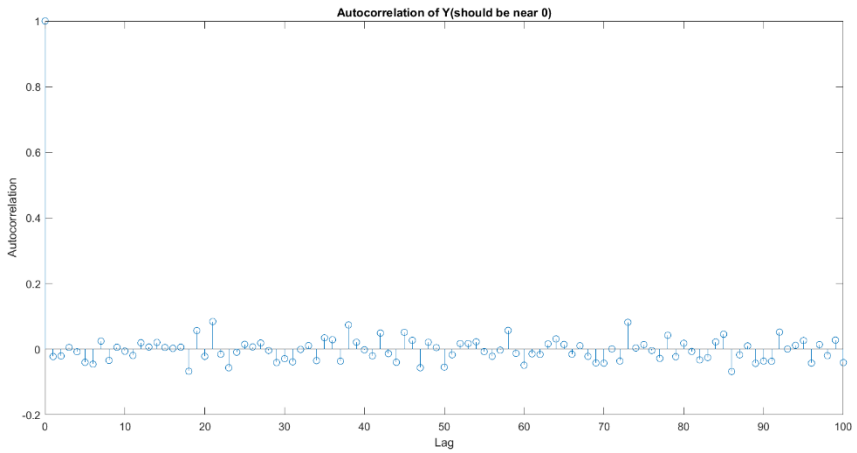












be continued …

In the code, we also provided some details and additional test in the command window: (like you see for our last example)

```
Mean of X (approx 0): -0.0059449
Variance of X (approx 1): 0.98029
Mean of Y (approx 101): 100.9734
Variance of Y (approx 20): 19.6058
The generated random variable follows a normal distribution (Anderson-Darling test).
The mean of the generated random variable is 101 (t-test).
The variance of the generated random variable is 20 (F-test).
>>
```

By conducting various tests, you will notice that the mean is approximately equal to 0 and the variance is consistently around 1.

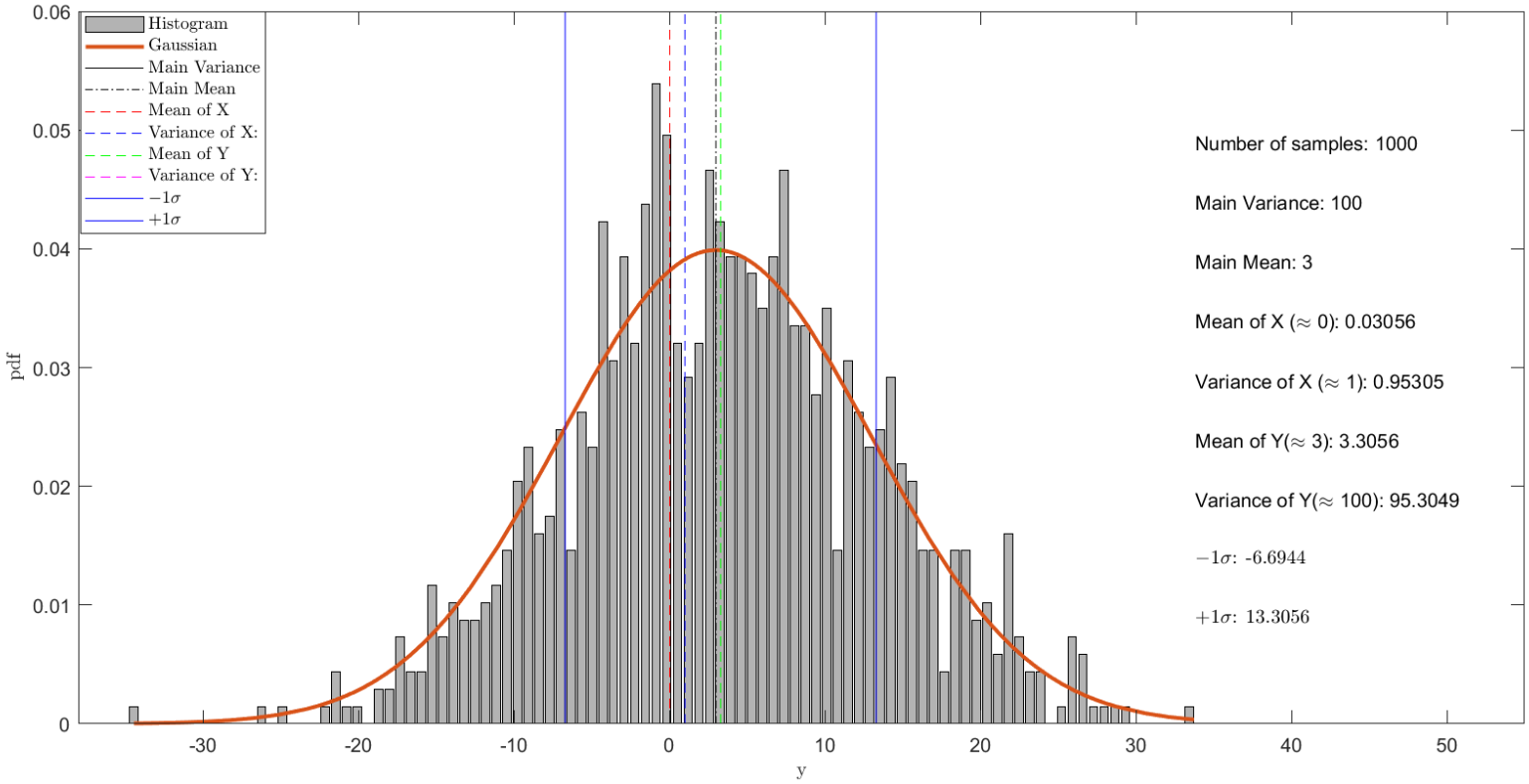
As previously stated, a normal distribution is observed when the mean **μ** is 0 and the standard deviation is 1. Therefore, it is reasonable to conclude that as the number of samples increases, our histogram, which depicts a normal distribution, will more closely resemble our Gaussian approximation. This means that the mean should approach 0 and the variance should approach 1.

2. In our class, we learned that you can change the mean and variance of a random variable X with the following transformation: Y=a X+ b with this in mind, generate a Gaussian random variable with a mean of 3 and a variance of 100, and verify your claim according to clause 1.

In the provided MATLAB code, the first section discusses the distribution and emphasizes the importance of having a mean value close to 0 and a variance value close to 1 for the variable X.
In this section, we will focus on the relationship between Y and X, specifically the equation Y = a X + b which stands for:

$$y = \sigma X + \mu$$

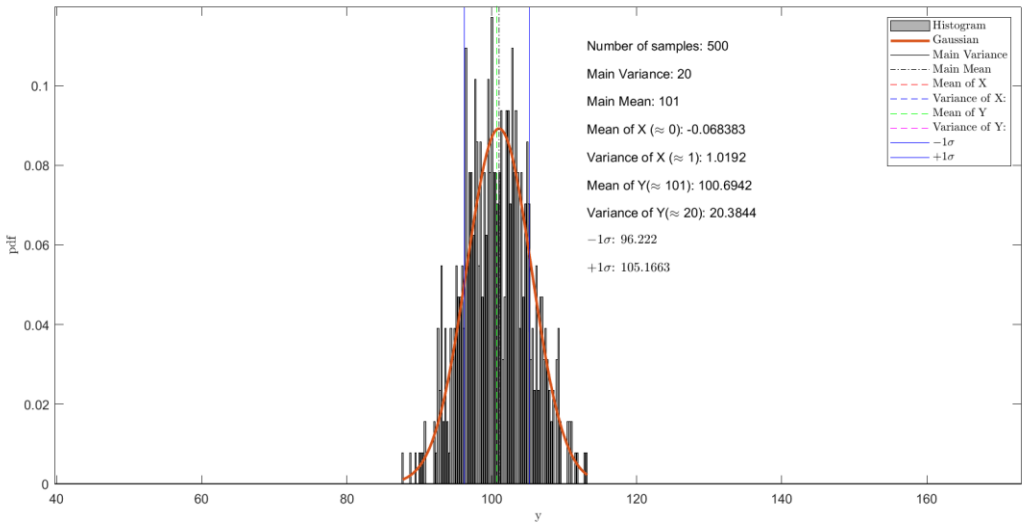
This equation demonstrates that the mean of Y should be similar to the chosen mean for X (mean of distribution), and the variance of Y should be similar to the chosen variance for X (variance of distribution).

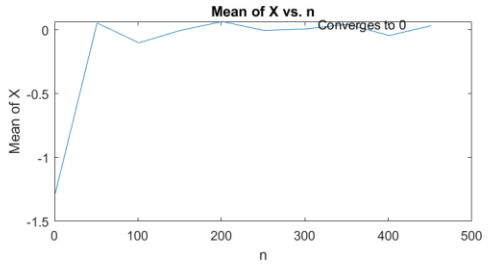


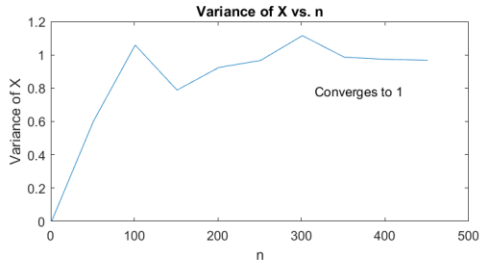
As expected, the mean and variance values of y are similar to the mean and variance values we selected for our distribution.
We can conduct additional tests by increasing the number of samples on subsequent pages.

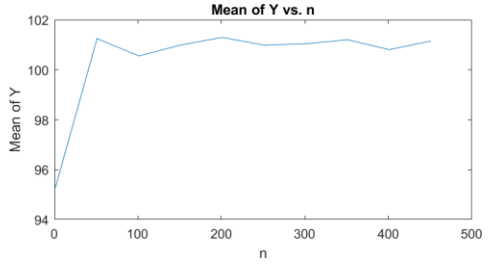
(additional test in future pages)

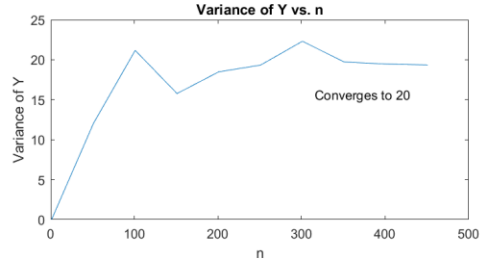
Alireza Sotoodeh Master: PhD. A Zaimbashi

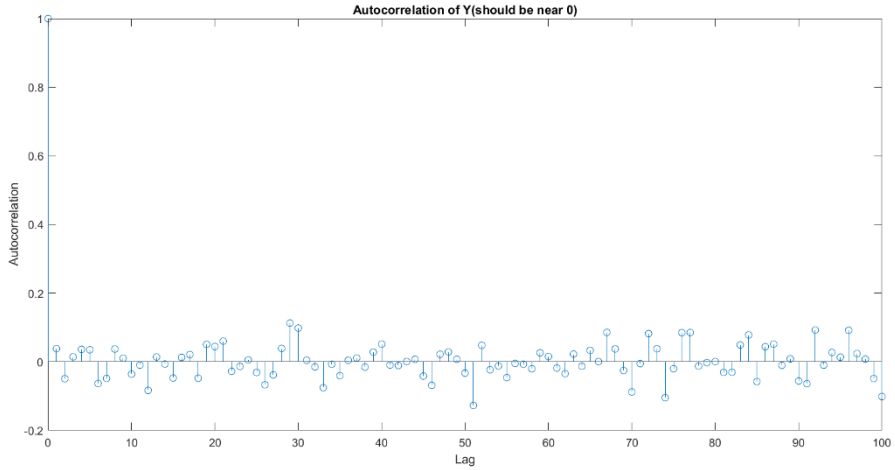




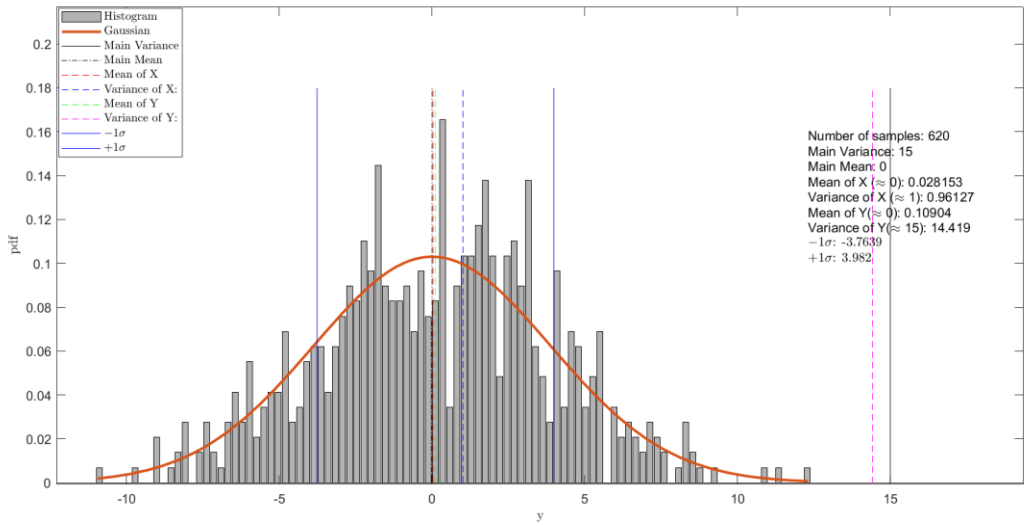


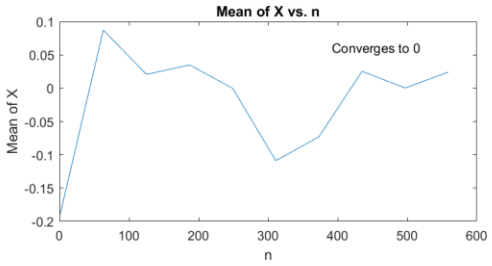


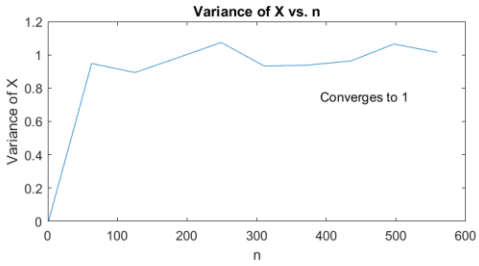


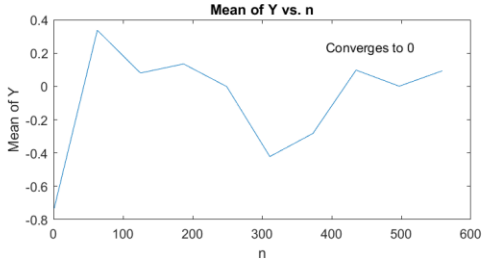


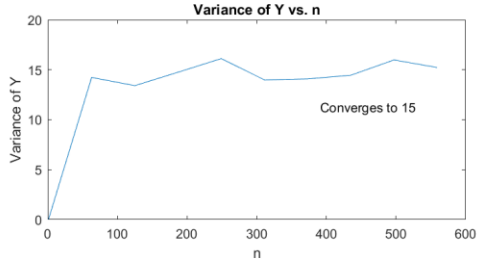
Alireza Sotoodeh Master: PhD. A Zaimbashi

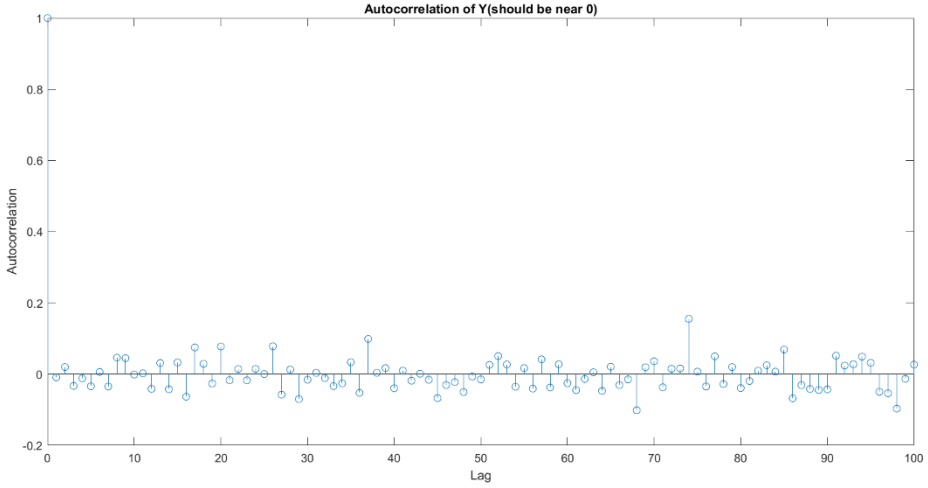




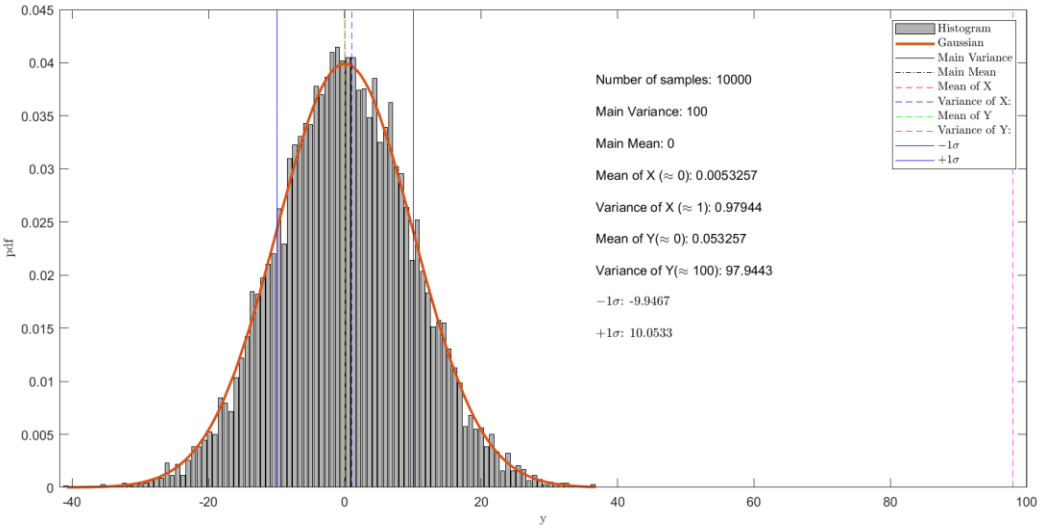


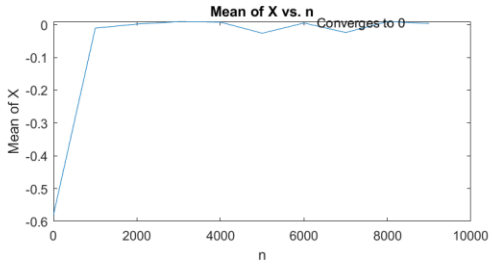


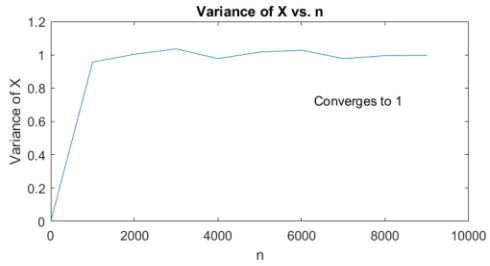


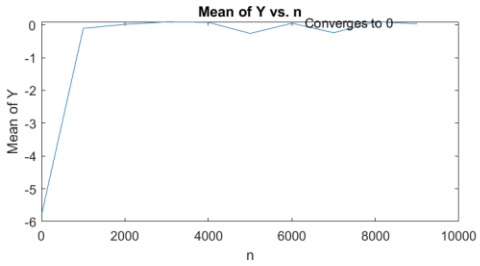


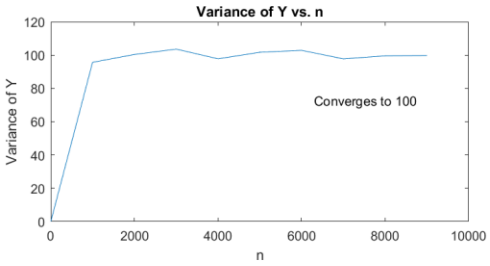
Alireza Sotoodeh Master: PhD. A Zaimbashi

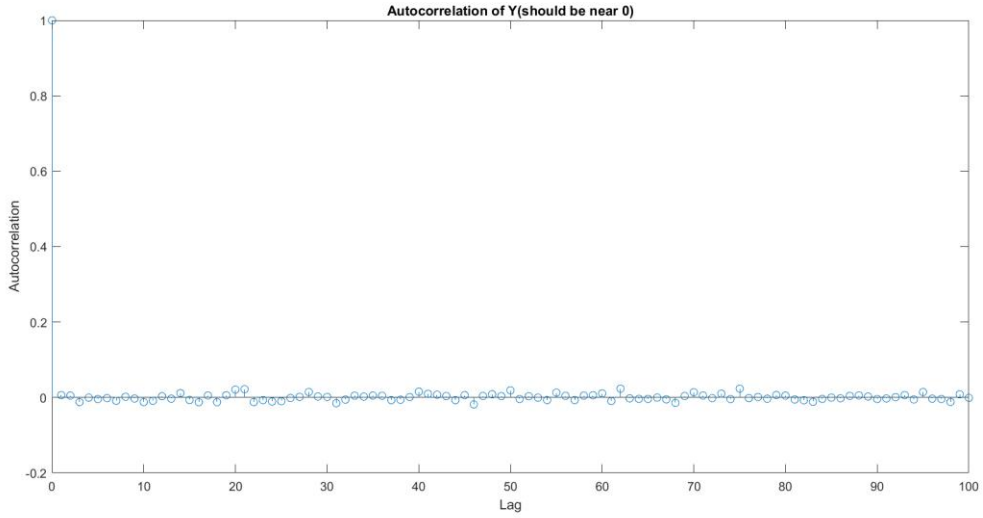












As we mentioned we in the code, we also provided some details and additional test in the command window: (like you see for our last example)

```
Mean of X (approx 0): 0.0053257
Variance of X (approx 1): 0.97944
Mean of Y (approx 0): 0.053257
Variance of Y (approx 100): 97.9443
The generated random variable follows a normal distribution (Anderson-Darling test).
The mean of the generated random variable is 0 (t-test).
The variance of the generated random variable is 100 (F-test).
fx >>
```

By increasing the number of samples, we can see that the mean and variance of y become more similar to the overall mean and variance, the same as our Gaussian approximation to the normal distribution!

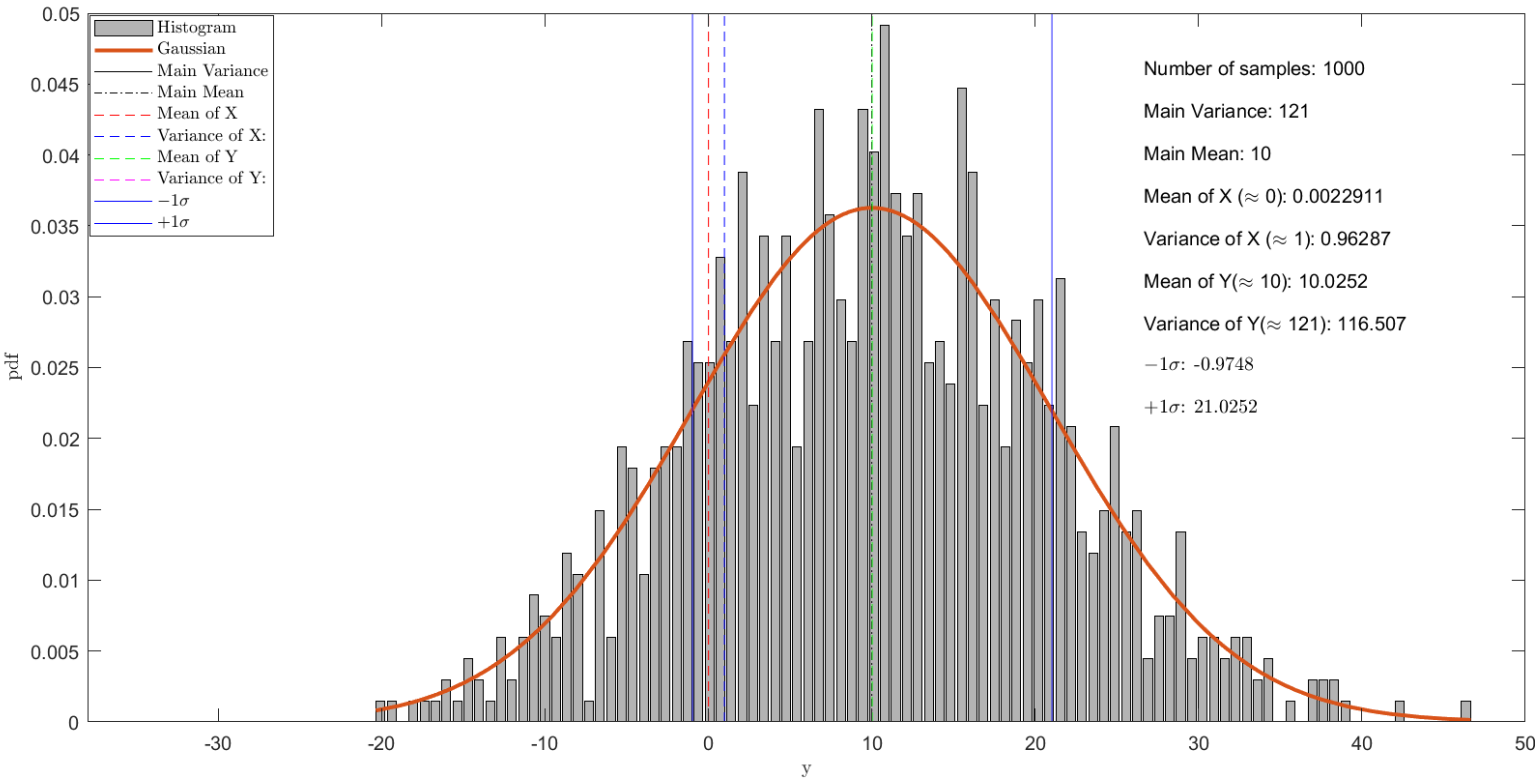
- Changin value of μ : shifting the figure to right and left
- Changin value of σ: the distribution will become wider.

3. Generate a Gaussian data set with a mean of 3 and a variance of 100. Use the hist or bar command to plot the probability density function (PDF) of this data. Note that the area under the PDF should be one, so apply this point in the hist command according to the class notes. On the other hand, we know that a Gaussian distribution with mean μ and variance σ² has the following PDF:

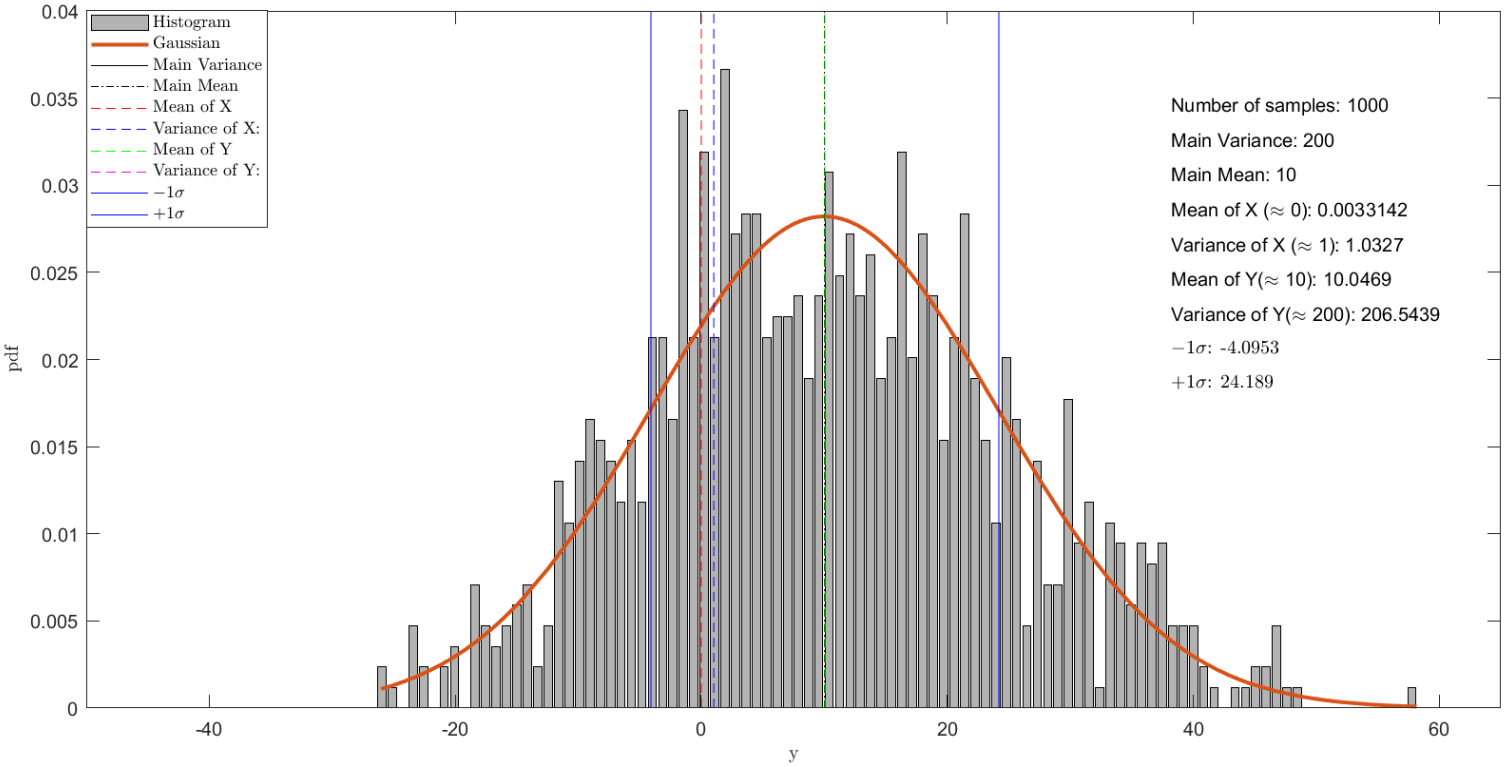
$$f_Y(y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-\mu)^2}{2\sigma^2}}$$

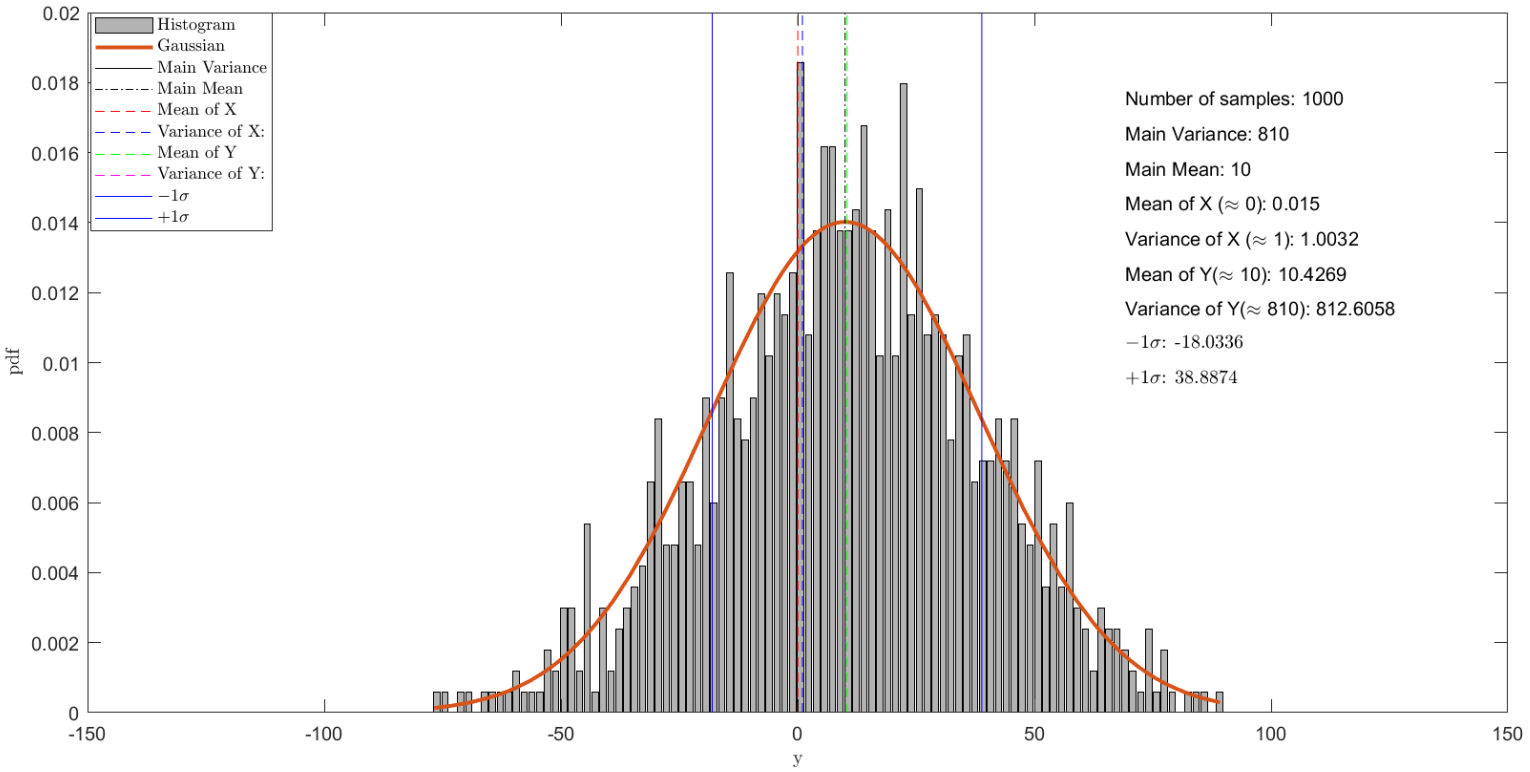
By plotting the two figures obtained from the above equation and the hist command, check the correctness of your work."

In future tests and those already conducted in this report, one can observe a noticeable difference between the histogram representing the distribution and the Gaussian approximation.

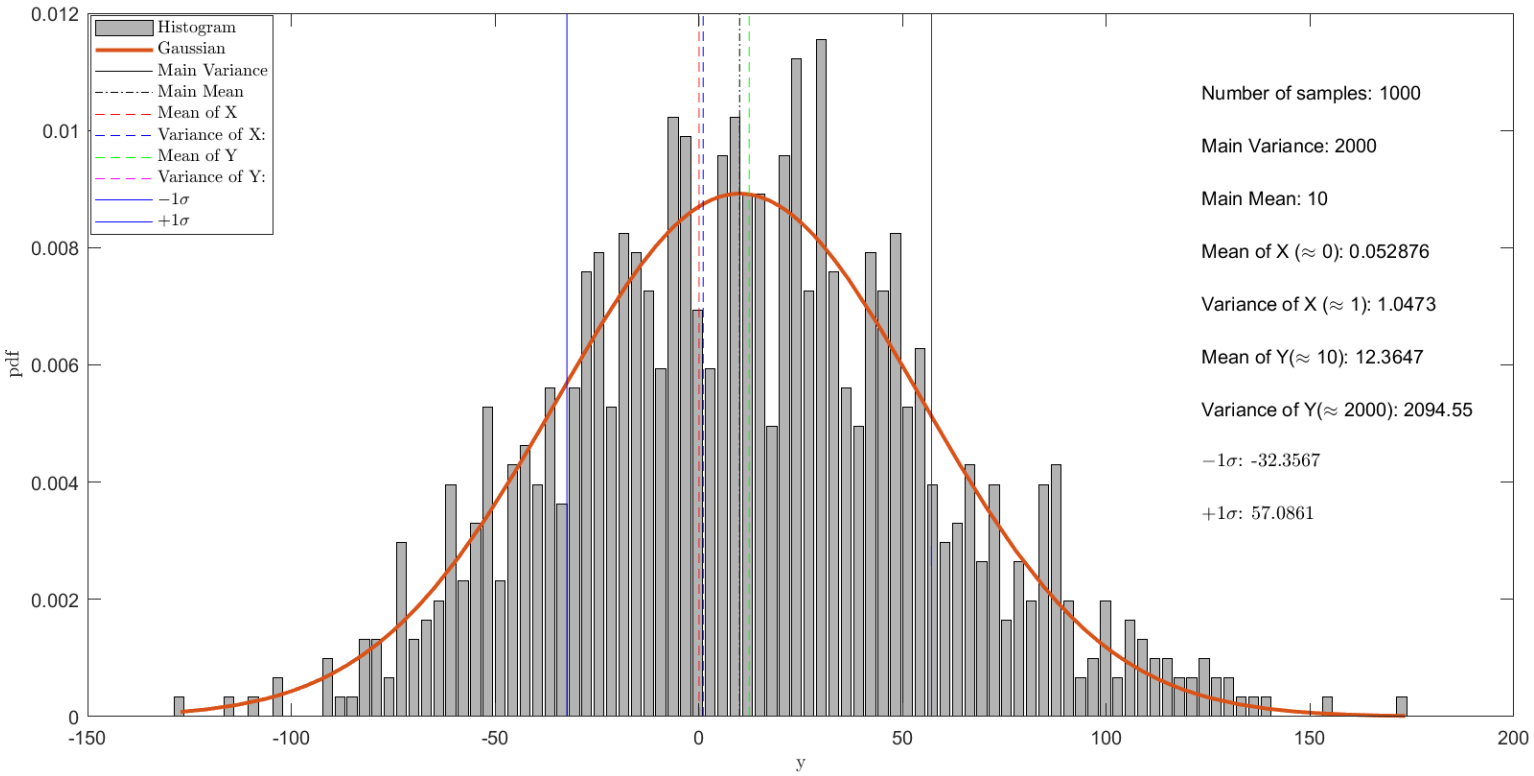


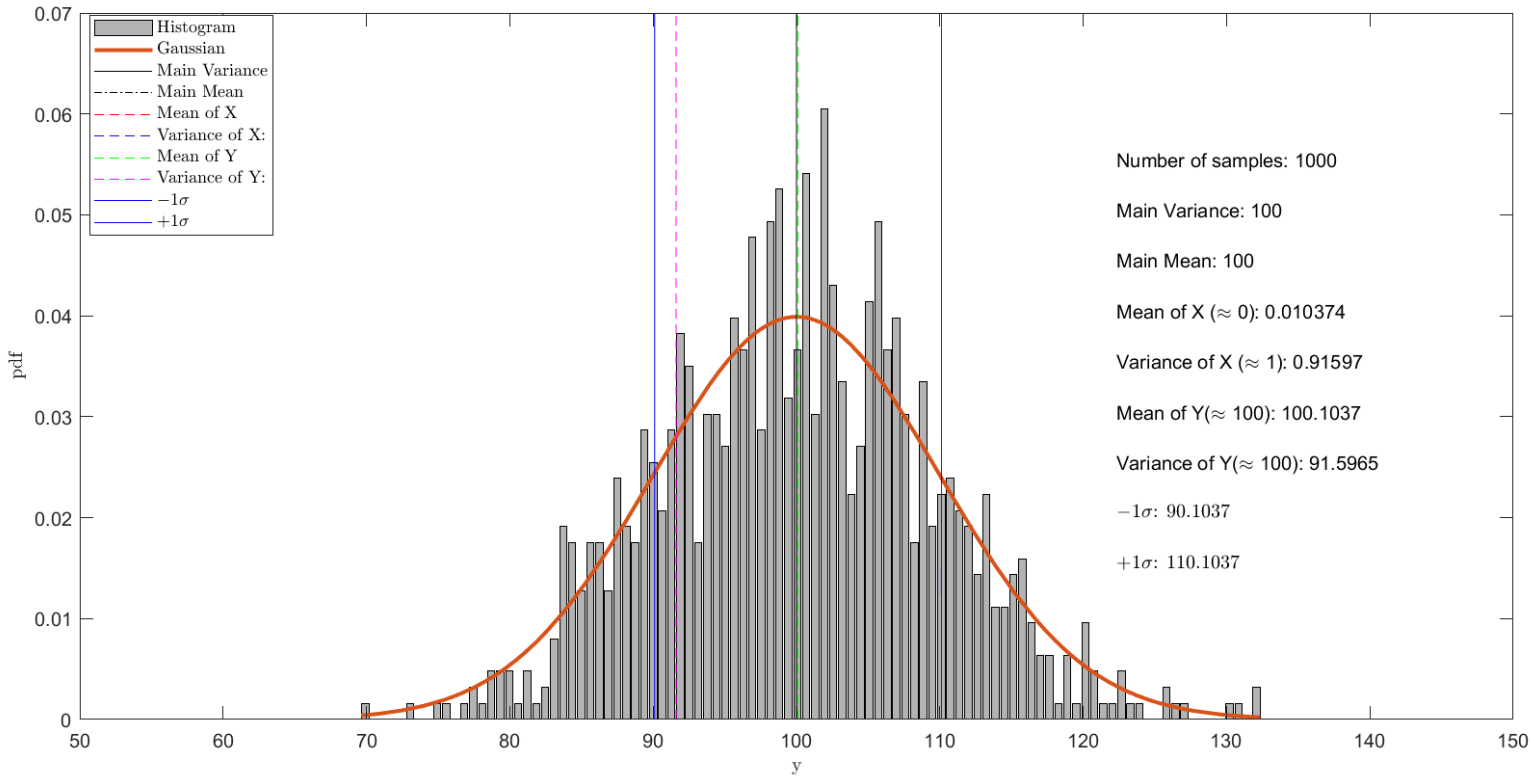
be continued …

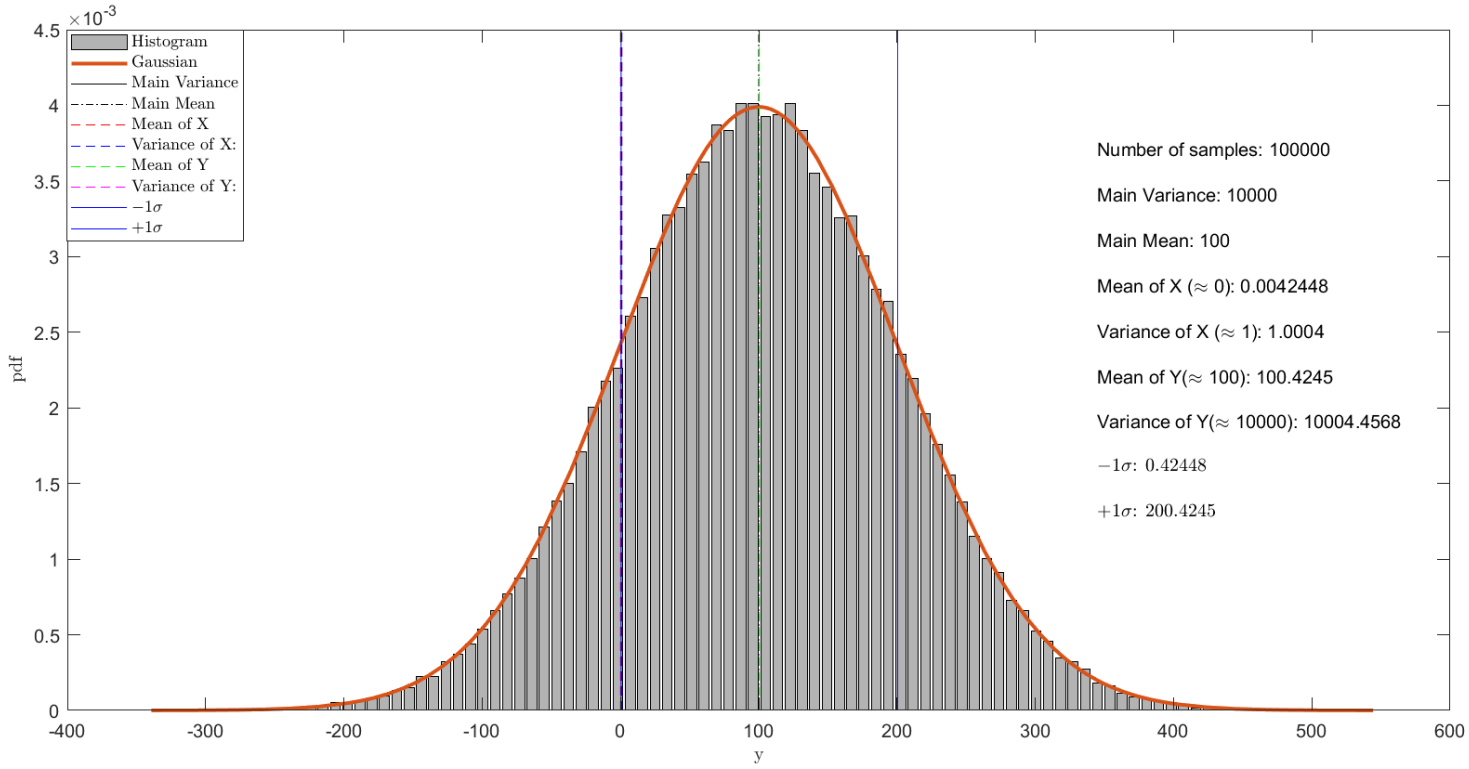




be continued ...


Histogram
Gaussian
Main Variance
Main Mean
Mean of X
Variance of X:
Mean of Y
Variance of Y:
-1σ
$+1\sigma$
Number of samples: 1000
Main Variance: 2000
Main Mean: 10
Mean of X (≈ 0): 0.052876
Variance of X (≈ 1): 1.0473
Mean of Y(≈ 10): 12.3647
Variance of Y(≈ 2000): 2094.55
-1σ: -32.3567
$+1\sigma$: 57.0861
pdf
y


Histogram
Gaussian
Main Variance
Main Mean
Mean of X
Variance of X:
Mean of Y
Variance of Y:
-1σ
$+1\sigma$
Number of samples: 1000
Main Variance: 100
Main Mean: 100
Mean of X (≈ 0): 0.010374
Variance of X (≈ 1): 0.91597
Mean of Y(≈ 100): 100.1037
Variance of Y(≈ 100): 91.5965
-1σ: 90.1037
$+1\sigma$: 110.1037
pdf
y



In the ideal scenario, the approximation and distribution would be equal in a large number of samples, regardless of the values of variance and mean.
If our number of sample be great distribution and approximation are equal!

4. E{x}

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%{
Autor: Alireza Sotoodeh(Student ID:401412056)
Subject:  A Study on Mean Convergence
%}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
clc
close all
%-------------------------------------------------------------------------
n=1000;                  % Number of samples
mu = 5;                  % Mean
variance = 100;          % variance = sigma^2
%-------------------------------------------------------------------------
x = linspace(0,n,n);
y = mu * ones(size(x)); %filled with the mean value
plot(x, y, 'r'); % Plot the mu line
hold on;
xlabel('x');
ylabel('Mean');
title('Line representing mean value');
grid on;
avg_X = zeros(1, n/10); % Initialize an array to store the average of X
i = 1;
for N=10:10:n;
X = sqrt(variance)*randn(1,N) + mu;
avg_X(i) = mean(X); % Calculate the average of X
if mod(N,1000) == 0 % Update the plot every 1000 iterations
plot(10:10:N, avg_X(1:i), 'b-'); % Plot the average of X
drawnow; % Update the plot immediately
end
if abs(avg_X(i) - mu) < 0.01  % Command window
fprintf(['The average of X for N=%d is approximately equal ' ...
'to the mean: %f\n'], N, avg_X(i)); % Print the average of X
end
i = i + 1;
end
plot(10:10:n, avg_X, 'b-'); % Plot the final average of X
%-------------------------------------------------------------------------
legend('Mean Value', 'Average of X');
```

Appendix: MATLAB Code Used for This Project

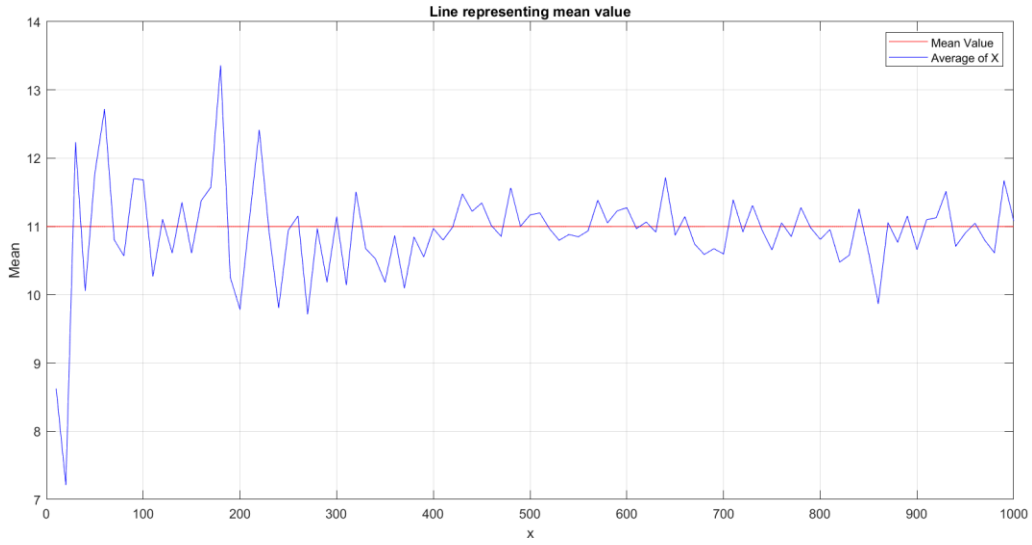
As we mentioned earlier if n(number of samples) is great then we expect a mean of x (which is a random value distribution) should Converge to the main mean we consider for distribution!

(tests in future pages)

N = 1000

Mu =11

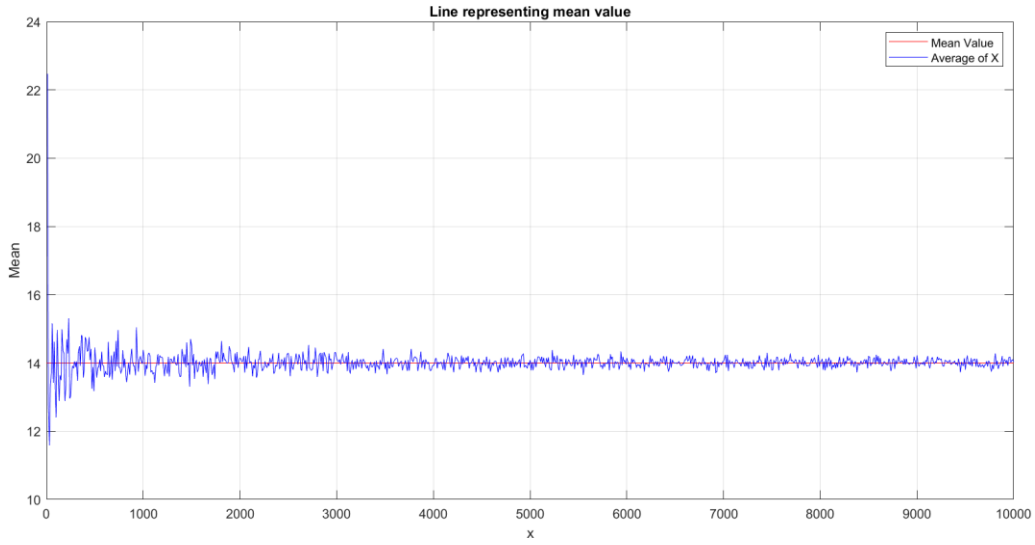
Var=100



N = 10000

Mu =14

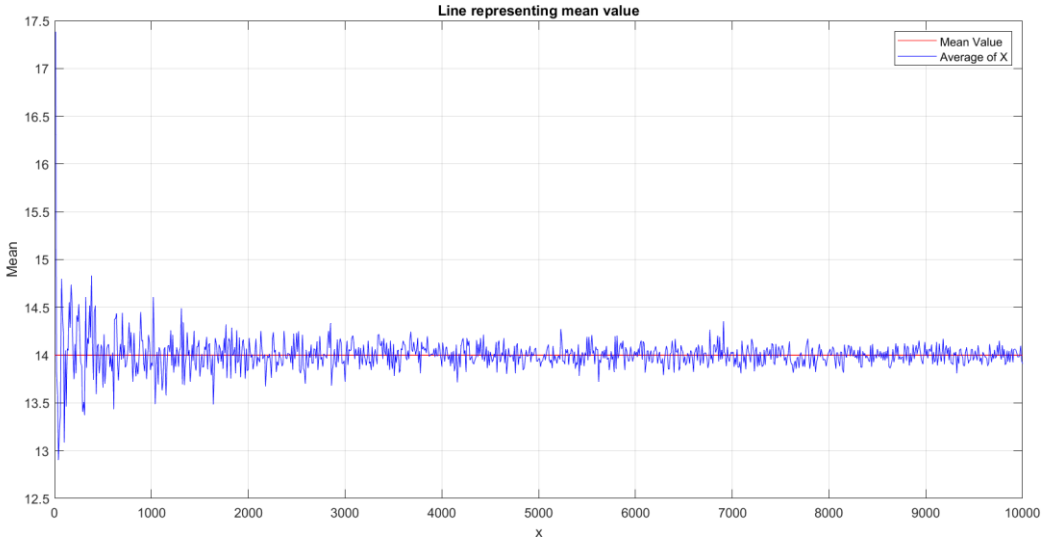
Var=100



N = 10000

Mu =14

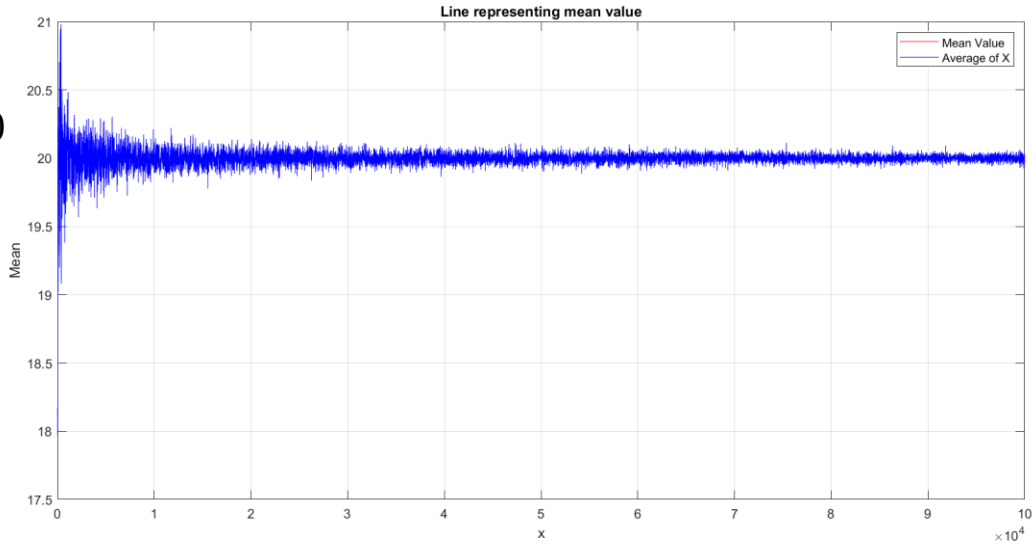
Var=50



N = 100000

Mu =20

Var=50



As we expected based on increasing the values of n, it's more likely to see, average x craves to its true value or mean regardless of value variance!

5. E[(X - \mu)^2] (variance)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%{
Autor: Alireza Sotoodeh(Student ID:401412056)
Subject:  A Study on Variance Convergence
%}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
clc
close all
%-------------------------------------------------------------------------
n=100;                    % Number of samples
mu = 5;                    % Mean
variance = 100;            % variance = sigma^2
%-------------------------------------------------------------------------
x = linspace(0,n,n);
y = variance * ones(size(x)); %filled with the variance value
plot(x, y, 'r'); % Plot the line
hold on;
xlabel('x');
ylabel('Variance');
title('Line representing variance value, Variance = E[(X - \mu)^2]');
grid on;
var_X = zeros(1, n/10); % Initialize an array to store the variance of X
i = 1;
for N=10:10:n;
    X = sqrt(variance)*randn(1,N) + mu;
    var_X(i) = var(X); % Calculate the variance of X
    if mod(N,1000) == 0 % Update the plot every 1000 iterations
        plot(10:10:N, var_X(1:i), 'b-'); % Plot the variance of X
        drawnow;
    end
    if abs(var_X(i) - variance) < 0.01  % Command window
        fprintf(['The variance of X for N=%d is approximately equal ' ...
            'to the variance: %f\n'], N, var_X(i)); % print variance of X
    end
    i = i + 1;
end
plot(10:10:n, var_X, 'b-'); % Plot the final variance of X
%-------------------------------------------------------------------------
legend('Variance Value', 'Variance of X');
```

Appendix: MATLAB Code Used for This Project

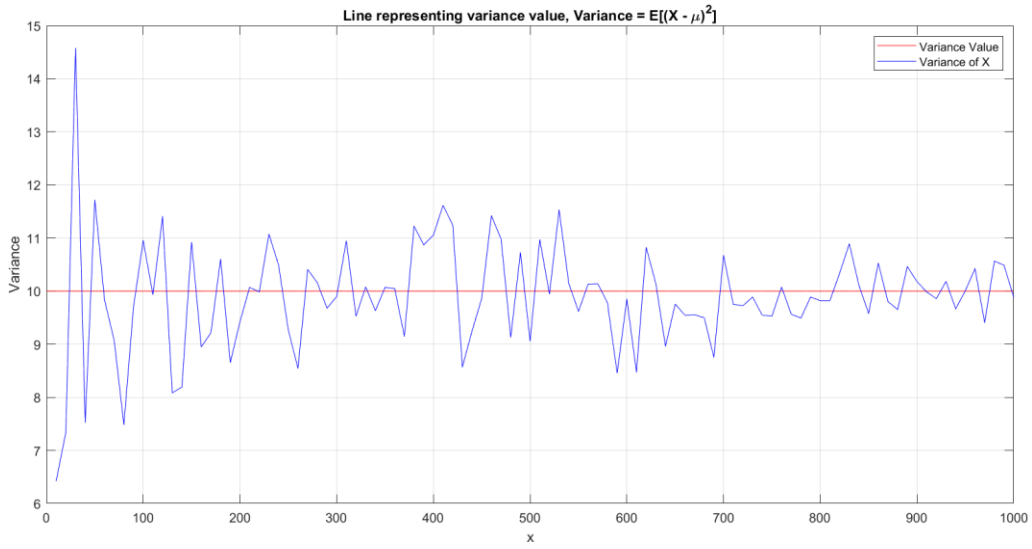
As we mentioned earlier if n(number of samples) is great then we expect a variance of x (which is a random value distribution) should Converge to the main variance we consider for distribution!

(tests in future pages)

N = 1000

Mu =5

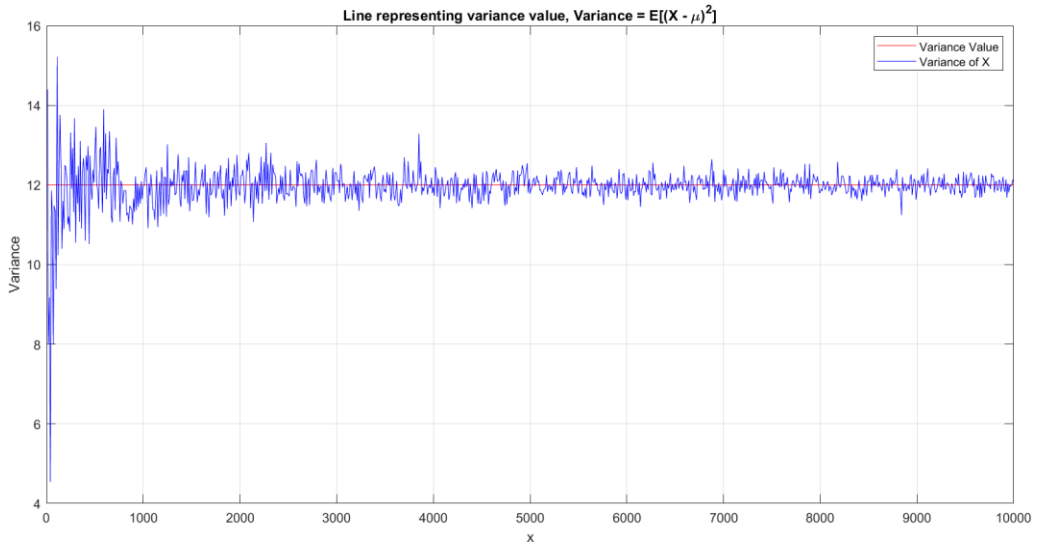
Var=10



N = 10000

Mu =23

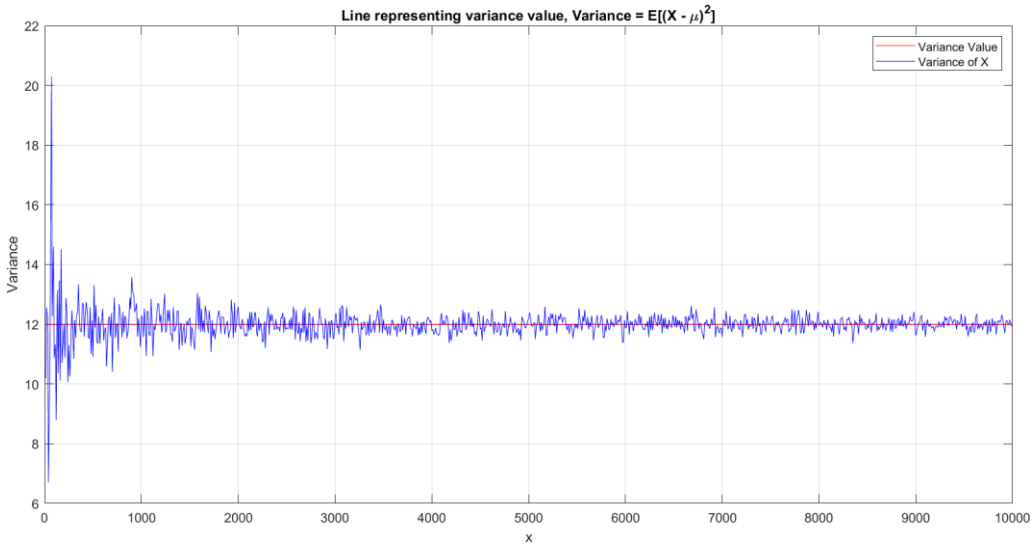
Var=12



N = 10000

Mu =0

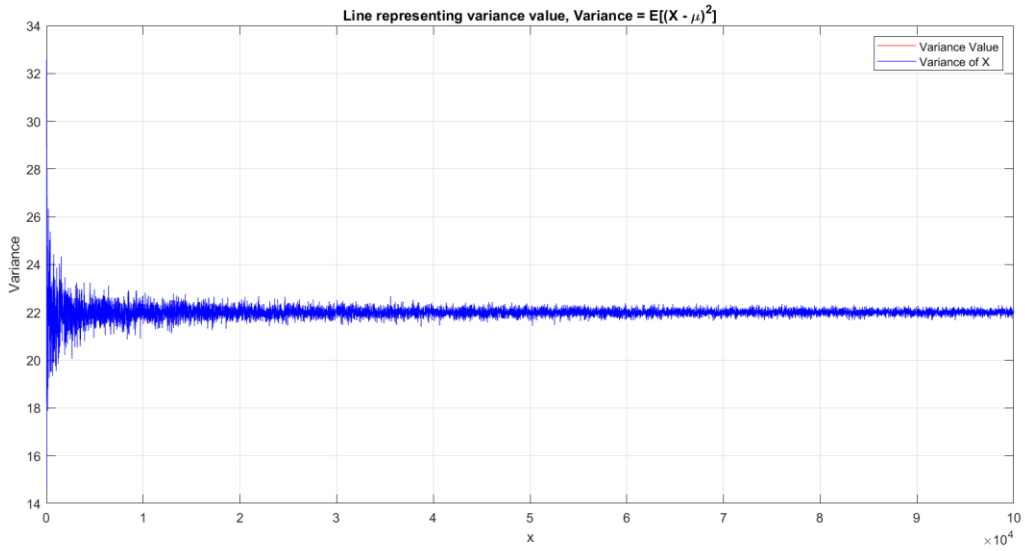
Var=12



N = 100000

Mu =0

Var=22



As we expected based on increasing the values of n, it's more likely to see, the variance of x craves to its true value regardless of value mu!

the project with an additional grade: $E\{x^3\}$

Appendix: MATLAB Code Used for This Project

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%{
Autor: Alireza Sotoodeh(Student ID:401412056)
Subject:  A Study on Cubic Mean Convergence
%}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
clc
close all
%-------------------------------------------------------------------------
n=1000;                % Number of samples
mu = 5;                  % Mean
variance = 100;          % variance = sigma^2
%-------------------------------------------------------------------------
x = linspace(0,n,n);
y = mu * ones(size(x)); %  filled with the mean value
plot(x, y, 'r'); % Plot the line
hold on;
xlabel('x');
ylabel('Mean');
title('Line representing mean value');
grid on;
avg_X_cubed = zeros(1, n/10); % to store the average of X^3
i = 1;
for N=10:10:n;
    X = sqrt(variance)*randn(1,N) + mu;
    avg_X_cubed(i) = mean(X.^3); % Calculate the average of X^3
    if mod(N,1000) == 0 % Update the plot every 1000 iterations
        plot(10:10:N, avg_X_cubed(1:i), 'b-'); % Plot the average of X^3
        drawnow;
    end
    if abs(avg_X_cubed(i) - mu^3) < 0.01 %Command window_Print mean of X^3
        fprintf(['The average of X^3 for N=%d is approximately equal' ...
            ' to the cube of the mean: %f\n'], N, avg_X_cubed(i));
    end
    i = i + 1; % Increment the counter
end
plot(10:10:n, avg_X_cubed, 'b-'); % Plot the final average of X^3
%-------------------------------------------------------------------------
legend('Mean Value', 'Average of X Cubed');
```

This script is a simulation of the Law of Large Numbers for the cubic mean. It generates a sequence of random numbers, calculates their cubic mean.

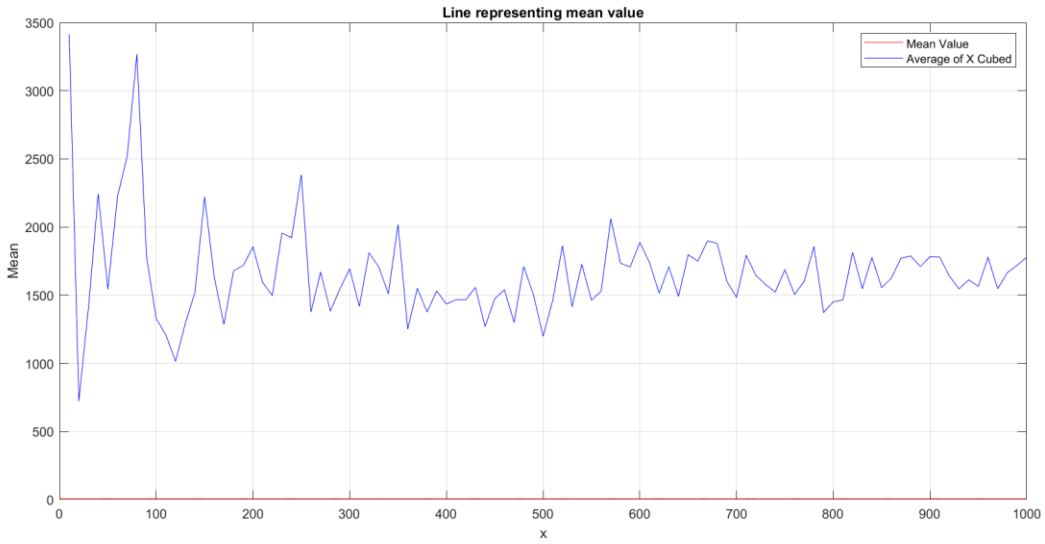
What happened by changing different values? :

- **Changing mu or variance:** This will change the distribution of your random numbers, which will in turn affect the cubic mean. The larger the variance, the more spread out your data will be, and it might take longer for the cubic mean to converge to the cube of the mean.
- **Changing n:** This changes the number of samples you' re taking. The larger n is, the more accurate your estimate of the cubic mean will be, according to the Law of Large Numbers. However, it will also make your script run slower because it has to generate and process more data.

N = 1000

Mu =5

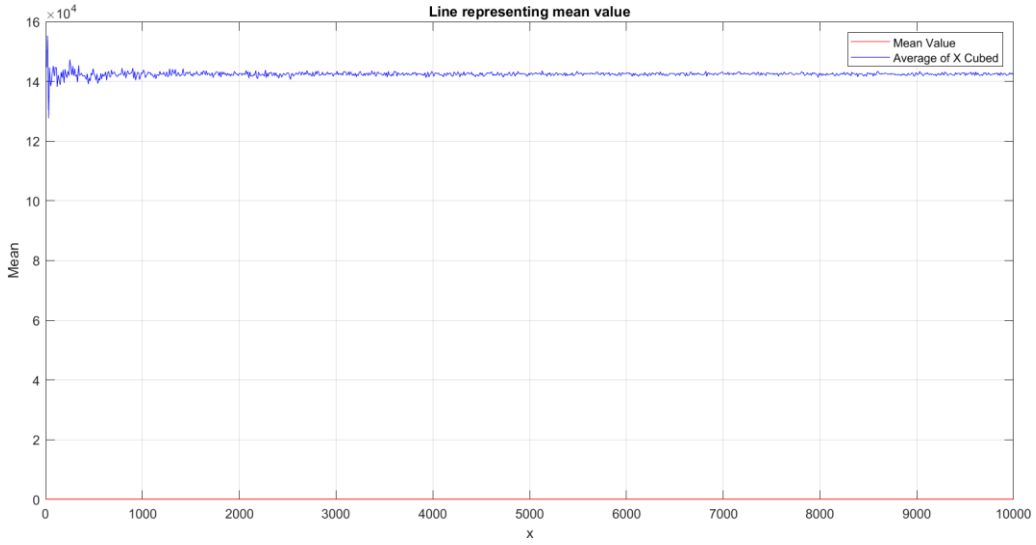
Var=100



N = 10000

Mu =52

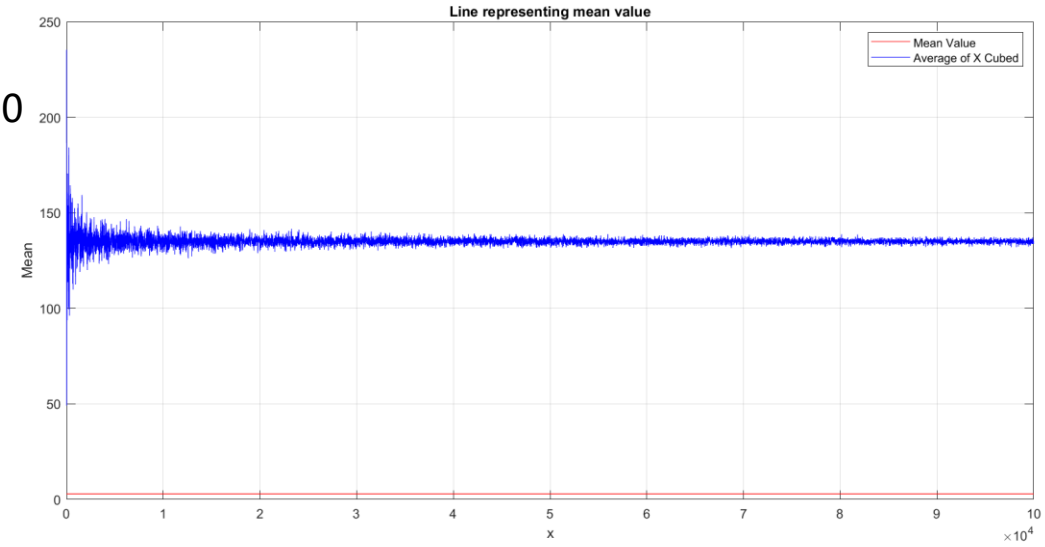
Var=12



N = 100000

Mu =3

Var=12



Thanks for reading 😊

Hope the best

Alireza Sotoodeh

Spring 1403