Predictive modelling of Airport Maintenance Activities: M/M/2 Model, Markov Chain Process, and Monte Carlo Simulation Approach

Project report submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Computer Science and Engineering

by

Aditya Khadekar - 20UCS009
Naman Jain - 20UCS122
Divyanshi Gautam - 20UCS066
Yash Mittal - 20UCS234

Under Guidance of
Dr. Sudheer Sharma

Department of Computer Science and Engineering
The LNM Institute of Information Technology, Jaipur

May 2023

The LNM Institute of Information Technology
Jaipur, India

CERTIFICATE

This is to certify that the project ”Predictive modelling of Airport Maintenance Activities: M/M/2 Model, Markov Chain Process, and Monte Carlo Simulation Approach” submitted by Aditya Khadekar -(20UCS009) ,Naman Jain -(20UCS122), Divyanshi Gautam - (20UCS066), Yash Mittal - (20UCS234) in partial fulfilment of the requirement for a degree in Bachelor of Technology (B. Tech) is a genuine record of the work done by them at the Department of Computer Science and Engineering, The LNM Institute of Information Technology, Jaipur, (Rajasthan) India, during This report, in my/our opinion, is a regular requirement for the issuance of the Bachelor of Technology (B. Tech) degree.

Date

Adviser: Dr. Sudheer Sharma

Acknowledgments

We would like to express our gratitude to The LNM Institute of Information Technology and to our project mentor Dr.Sudheer Sharma in particular, who provided us with the wonderful opportunity to work on this project and whose inspiring suggestions and encouragement assisted us in finishing it. Without the collaboration of many people, this would not have been possible, and we would want to show our appreciation to them. To constantly succeed, you need one seasoned somebody to guide you from beginning to end for us, that person was our coach Asst. Prof. Dr. Sudheer Sharma.

Aditya Khadekar -(20UCS009)
Naman Jain -(20UCS122)
Divyanshi Gautam - (20UCS066)
Yash Mittal - (20UCS234)

Abstract

This report paper discusses how Markov models can be used to optimize airways landing and takeoff queues to reduce air traffic congestion. The M/M/2 system, triangular model, and Markov chain are used in combination to develop a comprehensive model that takes into account the characteristics of both the aircraft and the airport.

The M/M/2 system measures airport performance in terms of capacity, demand, and delays, providing a standardized way of measuring airport performance that can be used to compare different airports. The triangular model is used to model the arrival and service rates in a queue, which are critical parameters for understanding the behavior of air traffic queues. It assumes that the arrival rate and service rate are random variables that follow a triangular distribution.

Using Monte Carlo simulation, a large number of random samples are generated from the triangular distribution for each activity's cost. These samples encompass the range between the minimum and maximum values, with a higher likelihood of falling near the most likely value. By repeating this process numerous times, a distribution of possible total costs for the project is obtained. The mean total cost and standard deviation are calculated to estimate the expected cost and quantify the variability of the cost distribution.

The Markov chain is used to model the behavior of aircraft in landing and takeoff queues. The state of the system is defined by the number of aircraft waiting in the queue, and the transition probabilities between states are determined by the arrival and service rates modeled using the triangular distribution. By simulating the Markov chain, the behavior of the system over time can be predicted, and areas for improvement can be identified.

In conclusion, the combination of the M/M/2 system, triangular model, and Markov chain can significantly improve airport efficiency by optimizing airways landing and takeoff queues. This approach provides a comprehensive model that takes into account the characteristics of both the aircraft and the airport, allowing for a more accurate prediction of system behavior and identification of areas for improvement.

Contents

List of Figures

List of Tables

Chapter 1

Milestone

Sr. No.	Milestone and Description	Start Date	Finish Date	Status
1	Initial Research on Which topic is more efficient to replicate?	30/01/2023	15/02/2023	Completed
2	Studied existing Model's use cases in various applications to gain an understanding of how Monte Carlo can be applied in Real Life Systems	15/02/2023	28/02/2023	Completed
3	Searching the technical aspects of the project, including the M/M/2 Model, Markov Chain Process, and Monte Carlo Simulation Approach Discussed the programming languages and tools used, as well as any challenges faced during the development process.	28/02/2023	18/03/2023	Completed
4	implementation of the tools and how they were used to analyze M/M/2 Model, Markov Chain Process, and Monte Carlo Simulation Approach and providing examples of the tools in action and discuss any interesting findings or insights gained through the analysis.	18/03/2023	04/05/2023	Completed
5	Report Writing and analyzing the results through matlab and changing the parameters.	04/05/2023	13/05/2023	Completed

TABLE 1.1: Project's Milestones

Chapter 2

Introduction

Air traffic congestion continues to be a significant challenge for airports worldwide, with delays, reduced efficiency, and increased fuel consumption being common problems. Efficient management of aircraft landing and takeoff queues is critical in addressing this challenge. The use of Markov models has shown great promise in analyzing and predicting complex systems and can be applied to model the behavior of aircraft in landing and takeoff queues. This research paper explores the application of Markov models to optimize airways landing and takeoff queues using the M/M/2 system, triangular model, and Markov chain.

The M/M/2 system measures airport performance in terms of capacity, demand, and delays. It provides a standardized way of measuring airport performance that can be used to compare different airports and identify areas for improvement. By combining the M/M/2 system with the Markov chain and the triangular model, we can build a comprehensive model of airways landing and takeoff queues that takes into account the characteristics of both the aircraft and the airport.

The triangular model is a useful tool for modeling the arrival and service rates in a queue, which are critical parameters for understanding the behavior of air traffic queues. It assumes that the arrival rate and service rate are both random variables that follow a triangular distribution, which is defined by three parameters: the minimum, most likely, and maximum values. This model is useful because it can capture the variability in the arrival and service rates that are often observed in real-world systems. By employing Monte Carlo simulation, we can generate a large number of random samples from the triangular distribution for each activity's cost. These samples will encompass the range between the minimum and maximum values, with a higher likelihood of falling near the most likely value. By repeating this process numerous times, we can obtain a distribution of possible total costs for the project.Using the resulting distribution of total costs, we can calculate the mean total cost, which provides an estimate of the average expenditure required for the airport maintenance system. This measure gives us an indication of the expected cost of the project.

Furthermore, we can compute the standard deviation, which quantifies the variability or dispersion of the cost distribution. A higher standard deviation implies a wider range of potential https://www.overleaf.com/project/64509d0ffb9f075afe87bfaccosts, suggesting a greater degree of uncertainty in the estimation.

The Markov chain is a mathematical model that describes a sequence of events where the probability of each event depends only on the state attained in the previous event. In the context of air traffic management, we can use a Markov chain to model the behavior of aircraft in landing and takeoff queues. The state of the system is defined by the number of aircraft waiting in the queue, and the transition probabilities between states are determined by the arrival and service rates modeled using the triangular distribution. By simulating the Markov chain, we can predict the behavior of the system over time and identify areas for improvement.

In this paper, we present a case study of a busy airport to demonstrate the effectiveness of the proposed approach. We show how the Markov chain can be used to model the behavior of the landing and takeoff queues, and how the MM2 system can be used to measure airport performance. We analyze the simulation results to identify bottlenecks and propose strategies to reduce congestion and improve efficiency. Our findings demonstrate the value of using a comprehensive approach that combines the MM2 system, triangular model, and Markov chain to optimize airways landing and takeoff queues.

Chapter 3

Proposed Work

3.1 M/M/2 Model for Airport Simulation.

3.1.1 M/M/2 Airport Simulation Model

The input parameters for the simulation include the mean arrival rate, mean service rate, and number of planes arriving and departing.

The program first initializes variables and generates random arrival time distributions for both landing and takeoff planes using an exponential distribution. It then generates random service time distributions when only one server is utilized and when both servers are used.

The simulation begins by setting up the first landing and takeoff planes. It then sets up the second landing and takeoff planes while considering whether one or both servers are occupied. Next, it calculates the service time for the remaining landing and takeoff planes based on whether one or both servers are free.

The output of the simulation includes the service start time, service time, and service end time for each landing and takeoff plane. The simulation can be used to analyze the queuing system and identify potential bottlenecks or areas for improvement.

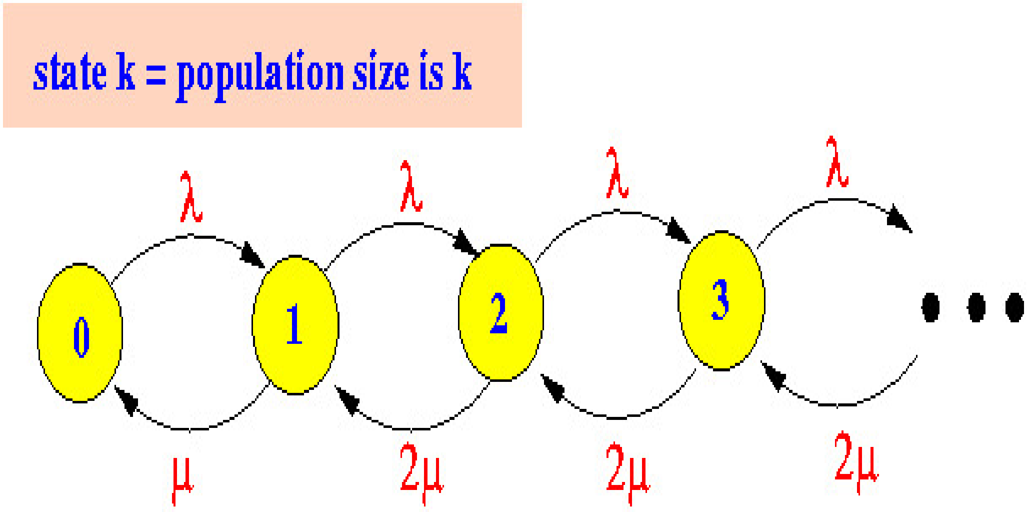

FIGURE 3.1: M/M/2 Model

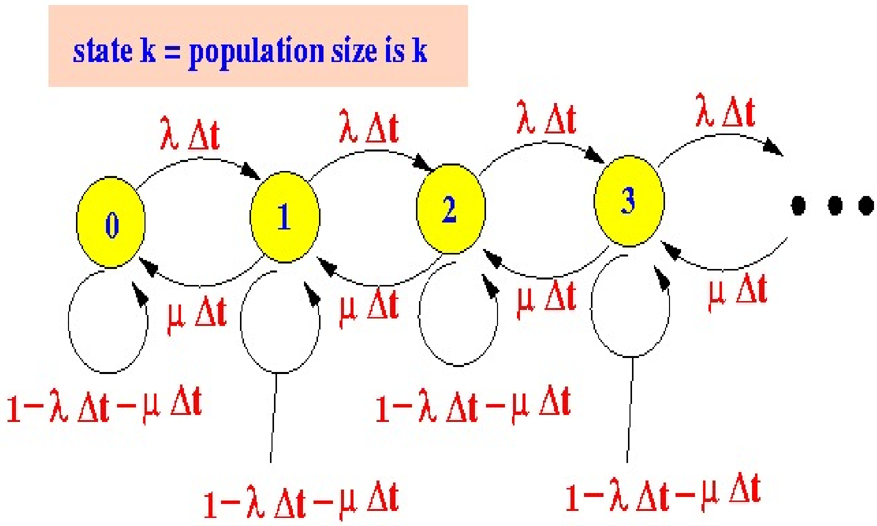

FIGURE 3.2: M/M/2 Model

3.1.2 Introduction to M/M/2 queuing System

The M/M/2 queuing model is a mathematical model used to describe a queuing system in which customers arrive according to a Poisson process and are served by two identical servers who work simultaneously and independently of each other.

The letters M/M/2 stand for:

M: Markovian arrival process, meaning that the inter-arrival times between customers are exponentially distributed.

M: Markovian service process, meaning that the service times are also exponentially distributed.

2: There are two servers in the system.

The assumptions of the M/M/2 model are:

Arrivals are independent and follow a Poisson distribution with rate λ.

Service times are exponentially distributed with rate μ.

The system has a capacity of two servers, which work simultaneously and independently of each other. Customers are served on a first-come, first-served basis. The system is in steady-state. The model is often used to analyze the performance of communication networks, computer systems, and manufacturing processes, among others. By using this model, it is possible to calculate various performance metrics of the system, such as the average number of customers in the system, the average waiting time, and the probability of a customer having to wait in a queue before being served.

Assume that steady state exists and let P_n be the limit of the probability of having n items in the system, denoted as $\Pr\{N(t) = n\}$, where $n = 0, 1, \ldots$. $N(t)$ represents the number of items in the system, including those in the service channel and in the queue (if any), at time t.

p_n represents the proportion of time the process is in state n ($n \geq 0$). In the steady state, the rate up from a particular state n to the next step $(n + 1)$ is equal to the rate down from state $(n + 1)$ to the original state n. Therefore, for the steady state probabilities P_1, we can set $P_1 = 2P_0\rho$.

For $n = 1, 2, \ldots$, we have the relation $P_{n+1} = 2\rho P_n$.

By the normalizing condition $\sum P_n = 1$, where the sum is taken from $n = 0$ to infinity, we get the equation:

$$\sum P_n = P_0 + \sum 2\rho P_0 = 1,$$

and rearranging further:

$$P_0 + 2\rho \sum P_{n-1} = 1.$$

Simplifying, we have:

$$P_0 + \frac{2\rho P_0}{1-\rho} = 1,$$

$$\frac{1-\rho}{1+\rho} = P_0,$$

where $\rho = \frac{\lambda}{\mu}$.

The steady state probabilities can be calculated using the equation:

$$P_n = \frac{2\rho^n(1+\rho)}{1-\rho}.$$

Please note that equation (3) represents the expression for calculating the steady state probabilities P_n.

3.1.3 System Components in our simulation

- **Entity**: Planes coming to land or ready to take-off.
- **Attribute**: Checking time of Arrival and time of Departure.
- **Activity**: Landing of a plane and Take-off of a plane.
- **State**: Airport Traffic.
- **Event**: Plane Arrival, and Plane Departure on and from the runway respectively.
- **State Variables**: Number of Planes in the Queue waiting to Land (In the air) and Number of Planes in the Queue waiting to take-off.

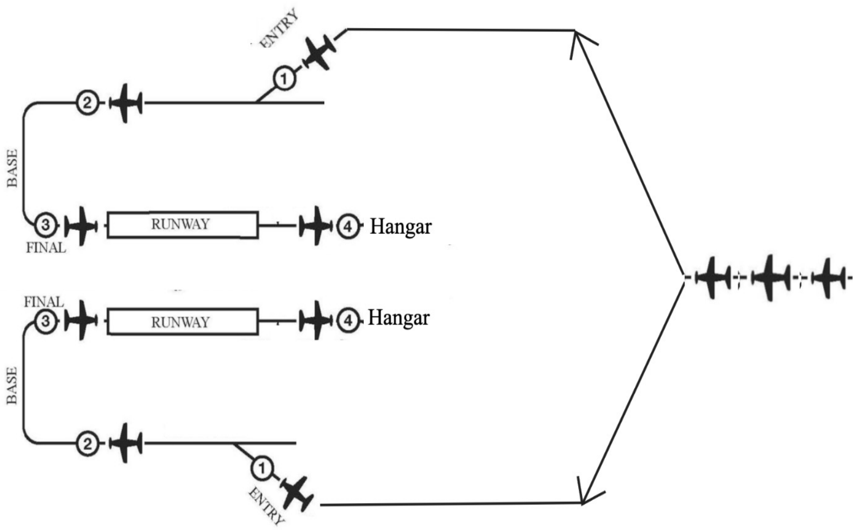

FIGURE 3.3: Snippet 1 of Markov Chain Model

```
%% Clear the environment and the command line
clear;
clc;
close all;

%% Define the input parmeters
% ----------------------------------------------------------
num_servers = 2;        % cannot be changed in this code
% ----------------------------------------------------------

mean_arr_rate = 2;         % Mean arrival rate (cust/min)
mean_ser_rate = 1.5;       % Mean service rate (cust/min)
num_planes_landing = 500; % Number of landing planes
num_planes_takeoff = 500; % Number of takeoff planes
```

FIGURE 3.4: Snippet 1 of M/M/2 Model

3.1.3.1 Simulation Description

Simulation of the Landing of Planes and Take-off of Planes is happening on two different runways, so we have taken an M/M/2 system here. Two servers since we have two runways. We initialize the mean arrival rate (λ), mean service rate (μ), and also the number of planes.

```
%% Clear the environment and the command line
clear;
clc;
close all;

%% Define the input parmeters
% -------------------------------------------------------
num_servers = 2;        % cannot be changed in this code
% -------------------------------------------------------

mean_arr_rate = 2;         % Mean arrival rate (cust/min)
mean_ser_rate = 1.5;       % Mean service rate (cust/min)
num_planes_landing = 500;  % Number of landing planes
num_planes_takeoff = 500;  % Number of takeoff planes
```

FIGURE 3.5: Snippet 1 of M/M/2 Model

We are storing the start time as well as the end time of service of the individual plane (landing or take-off) in a separate array (i.e., `service_start_time` and `service_end_time`) and initializing them with zeroes.

```
% Variable initializations
%--landing--
service_start_time_landing = zeros(1, num_planes_landing);
service_time_landing = zeros(1, num_planes_landing);
service_end_time_landing = zeros(1, num_planes_landing);
%--takeoff--
service_start_time_takeoff = zeros(1, num_planes_takeoff);
service_time_takeoff= zeros(1, num_planes_takeoff);
service_end_time_takeoff= zeros(1, num_planes_takeoff);
```

FIGURE 3.6: Snippet 2 of M/M/2 Model

Distribution follows exponential, so we create an arrival time array denoting the time at which planes arrive (`pd_arr`) using the `makedist` object in the MATLAB library generated randomly.

```
% Define random arrival time distribution

%--landing--
pd_arr = makedist('exponential', 'mu', 1/mean_arr_rate); %Exponential Distribution Object
rand_arr_time_landing = random(pd_arr, 1, num_planes_landing);
cum_arr_time_landing = cumsum(rand_arr_time_landing);   % Arrival time in real scale

%--takeoff--
rand_arr_time_takeoff= random(pd_arr, 1,num_planes_takeoff);
cum_arr_time_takeoff= cumsum(rand_arr_time_takeoff);   % Arrival time in real scale
```

FIGURE 3.7: Snippet 3 of M/M/2 Model

Similarly, for service time, we have used two choices. First, when only one server is occupied. This means if one runway is only used for landing or takeoff, then the mean service rate is μ. And when two servers are occupied, the mean service rate is $\mathbf{2}\mu$.

```
% Define random service time distribution when both servers are utilized
pd_ser = makedist('exponential', 'mu', 1/(num_servers * mean_ser_rate));
rand_ser_time_landing = random(pd_ser, 1, num_planes_landing);
rand_ser_time_takeoff= random(pd_ser, 1, num_planes_takeoff);
```

FIGURE 3.8: Snippet 4 of M/M/2 Model

POST SIMULATION COMPUTATIONS

System time for landing and take-off, waiting time in queue, length of the queue in time until it becomes 0 again, average waiting time for both landing and take-off, and standard deviations for queue length and wait time.

```
%% Post Simulation Computations

system_time_landing = service_end_time_landing - cum_arr_time_landing;
q_wait_time_landing = service_start_time_landing - cum_arr_time_landing;

system_time_takeoff = service_end_time_takeoff- cum_arr_time_takeoff;
q_wait_time_takeoff = service_start_time_takeoff- cum_arr_time_takeoff;

cum_q_wait_time_landing = cumsum(q_wait_time_landing);
run_mean_q_wait_time_landing = cum_q_wait_time_landing ./ (1:num_planes_landing);

cum_q_wait_time_takeoff = cumsum(q_wait_time_takeoff);
run_mean_q_wait_time_takeoff = cum_q_wait_time_takeoff ./ (1:num_planes_takeoff);
```

FIGURE 3.9: Snippet 5 of M/M/2 Model

3.2 Simulating a Markov Chain using Monte Carlo

The Markov process is used to model a stochastic process with random state variables $X(t_n)$ at time t_n. The state variables $X(t\_n)$ at time t_n are only related to the previous states $X(t\_i)$ $(i \ll n)$ of a finite number of time steps and are independent of states $X(t\_n-1), X(t\_n-2), \ldots, X(t\_n-i)$ $(i \ll n)$. Hence, as a "no memory process", the previous state of the Markov process does not affect the current one. The Markov process has been widely used to model stochastic processes, and it can quantitatively analyze the numerical, non-numerical, continuous, and discrete states of complex systems. Based on this, the system reliability can be analyzed. The essence here for reliability study is that the Markov process can be used to derive the probability value of each working state, the failure state, and the time required for the system to reach a steady state.

To apply the Markov processes for system reliability analysis, the following assumptions are often required.

- The system should be repairable.
- The life span and maintenance time of the system units obey an exponential distribution.
- Unit states are independent of each other and have no influence between them.

The common steps for system reliability analysis are as follows:

- Build the system state transition diagram: define the system states and draw the system transition diagram based on the failure and repair process.
- Formulate the system state transition equation: define the Markov chain and formulate its state transition equations.
- Compute the system reliability index: based on the state transition equation and system initial states, use Laplace and its inverse transformation to solve the state transition equations to obtain the reliability indices.

3.2.1 Three-State Markov Chain Process

As the number of system components increases, the complexity of the overall system states also increases, and the simple two-state repairable system model can no longer meet the requirements for reliability analysis. Therefore, a three-state Markov chain model which can represent different system areas is used to evaluate the reliability of a complex system, and a new time-based system average availability index is introduced to analyze and evaluate the overall system reliability.

The new three state Markov chain of system state transition diagram is built shown in below figure State '0' represents the **Ready** State of **Runway**, and states '1' represents the **Running** State of **Runway** and '2' respectively represent **Maintenance** State of Runway.

Here, We are presented with a system that has the following states: **Ready (R)**, **Operating (O)**, and **Maintenance (M)**.For,Example in the table below some examples of transition probabilitis are given:

State	Transition Probabilities
Ready (R)	$R \xrightarrow{0.3} O$
Operating (O)	$O \xrightarrow{0.1} M$
Maintenance (M)	$M \xrightarrow{0.5} R$

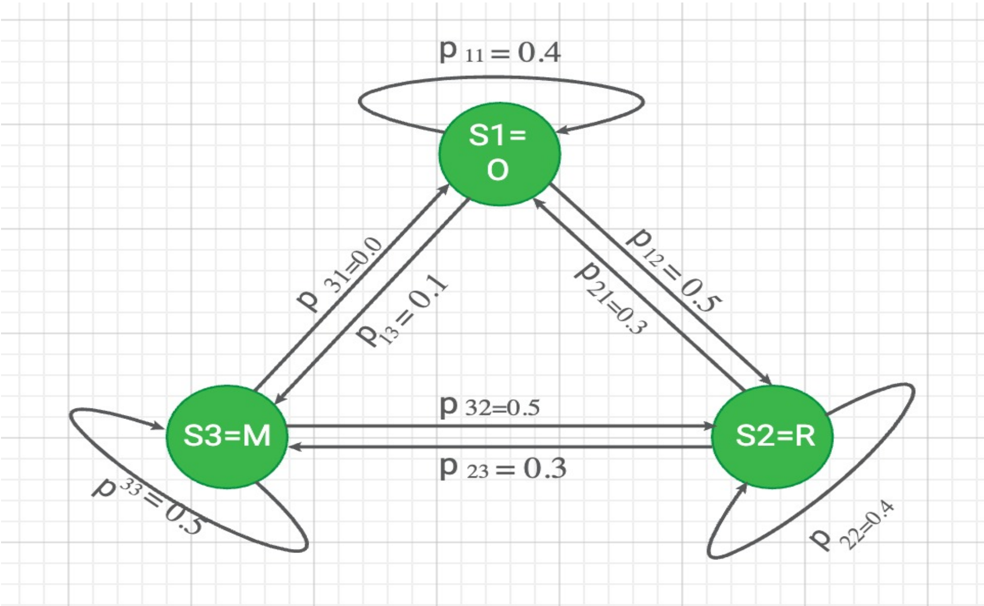

FIGURE 3.10: Three state system transition diagram

Based on Figure above, the system state transition equation in time [t, t + dt] can be derived. While P0 is the system normal work state, and P1 and P2 are the probabilities of two different system abnormal work states. The instantaneous availability of the system is given below.

$$\begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} 0.4 & 0.5 & 0.1 \\ 0.3 & 0.4 & 0.3 \\ 0.0 & 0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 0.4x + 0.5y + 0.1z & 0.3x + 0.3y + 0.4z & 0x + 0.5y + 0.5z \end{bmatrix}$$

Here,

$$\begin{bmatrix} x & y & z \end{bmatrix} = \pi_0$$

$$\begin{bmatrix} 0.4 & 0.5 & 0.1 \\ 0.3 & 0.4 & 0.3 \\ 0.0 & 0.5 & 0.5 \end{bmatrix} = A(TransitionMatrix)$$

$$\begin{bmatrix} 0.4x + 0.5y + 0.1z & 0.3x + 0.3y + 0.4z & 0x + 0.5y + 0.5z \end{bmatrix} = \pi_1$$

The probability of being in state $t = 2$ can be calculated by multiplying π_1 by the transition matrix:

$$\text{Probability}(\text{state}_{t=2}) = \pi_1 \times \text{TransitionMatrix}$$

Similarly, We can calculate the the probability of runway in each state by similar method.

3.2.2 Methodology

- Define the states and transition matrix: The three states (O, R, M) and their corresponding transition probabilities are specified in the code. The transition matrix represents the probabilities of transitioning between states.

```
states = {'O', 'R', 'M'};              % List the unique states here
num_states = size(states, 2);

% Define the state transition matrix
transition_mat = [  0.4     0.5     0.1;
                    0.3     0.4     0.3;
                    0.0     0.5     0.5];
```

FIGURE 3.11: Snippet 1 of Markov Chain Model

- Monte Carlo simulation: The Markov Chain is simulated using a Monte Carlo approach. A random draw is performed at each transition, and the next state is selected based on the computed thresholds from the transition matrix.

```
% Map state names to indices
state_map_ind = containers.Map(states, 1:num_states);

% Create a cell array for holding the generated state sequence
state_seq = cell(1, num_transitions + 1);
state_seq{1} = init_state;

% Construct the state matrix where each row represents the state vector at time step = row_index - 1
state_mat = zeros(num_transitions + 1, num_states);
state_mat(1, state_map_ind(init_state)) = 1;

trans_mat_cumsum = cumsum(transition_mat, 2);   % Compute thresholds
assert(all(trans_mat_cumsum(:, end) == 1), "Sum of probabilities != 1")
rand_draw = rand(1, num_transitions);           % Random numbers sampled

% Next State is selected based on random draw and computed thresholds
for t_index = 1:num_transitions
    state_cumsum = trans_mat_cumsum(state_map_ind(state_seq{t_index}), :);
    for s_index = 1:num_states
        if rand_draw(t_index) <= state_cumsum(s_index)
            state_mat(t_index + 1, s_index) = 1;
            state_seq(t_index + 1) = states(s_index);
            break;
        end
    end
end
```

FIGURE 3.12: Snippet 2 of Markov Chain Model

- Cumulative Running Mean (CRM) computation: The CRM of each state is calculated by cumulatively summing the state matrix and dividing it by the number of transitions. This provides a measure of the average occurrence of each state over time.

```
% Compute the CRM
crm_state_mat = cumsum(state_mat, 1);
divisor
```

FIGURE 3.13: Snippet 3 of Markov Chain Model

3.3 Cost-estimation using the triangular model for Airport Simulation.

3.3.1 Introduction to Triangular Model

The triangular distribution is a probability distribution that is often used in cost estimation to represent the uncertainty of cost estimates. It is characterized by three parameters: a minimum cost value, a maximum cost value, and a most likely cost value.

During the development of an automated cost estimating system, several factors led to the selection of the triangular probability-density function to model construction costs. The triangular-density function is customarily used when function parameters are directly estimated by experts. A typical example is for estimating activity durations by identifying a minimum value, a most likely value, and a maximum value. These values are then used to construct triangular-density functions to represent uncertain activity durations. For this work, however, it was necessary to estimate parameters of the triangular-density function using cost When using a range-estimating methodology, the cost items are assumed to be random variables rather than known parameters.
As a result, a probable distribution of total cost is obtained as compared to a single-point estimate that would be generated in a deterministic cost approach. The resulting probability distribution enables the cost estimator to define cost estimate values that can be associated with prescribed levels of confidence, thus permitting quantification of the exposure to financial risk. These features are very helpful to the cost engineer, both in preparing useful estimates and in managing resources during project execution.

Most range cost estimates are prepared using a probabilistic technique such as Monte Carlo simulation. In this technique, the individual cost elements, each defined as random variables, are sampled according to their distribution functions, and an associated total cost is calculated by summing the values.

```
          | 0                            if x < a
f(x) =    | 2(x-a)/((b-a)*(c-a))         if a <= x < b
          | 2(c-x)/((c-a)*(c-b))         if b <= x < c
          | 0                            if x >= c
```

FIGURE 3.14: The PDF for the triangular distribution

This sampling process is repeated a large number of times to produce a sample, or probable distribution of total costs Simply stated, the Monte Carlo technique uses a random number generator to simulate the construction of the project a multiple number of times (say, 1,000 or more), storing and recording the results after each iteration or pass through the process. The estimator can then represent the project costs as a probability distribution and make appropriate statistical inferences to guide financial decision making. A range-estimating methodology enables cost estimators to incorporate the inherent variability in component costs within the total cost estimate and to assign a level of confidence to each selected value.

Activity	Nodes	Duration Est (days)			Labor Cost ($/h)		
		Min	Max	Likely	Min	Max	Likely
A	1, 2	4	8	6	$100	$150	$120
B	1, 3	2	8	4	$100	$150	$120
C	2, 4	1	7	3	$100	$150	$120
D	3, 4	6	12	9	$100	$150	$120
F	3, 5	5	15	10	$100	$150	$120
I	3, 6	10	25	15	$100	$150	$120
J	4, 7	5	12	9	$90	$110	$100
K	5, 7	1	3	2	$90	$110	$100
L	6, 8	2	6	3	$90	$110	$100
M	7, 9	10	20	15	$90	$110	$100
N	8, 9	6	11	9	$90	$110	$100

FIGURE 3.15: Problem Statement for Cost Estimation Of Airport Maintenance

3.3.2 Methodology

Consider a Project that is characterized by the following activity edge table in fig 3.15 . The table indicates: 1) the activity names; 2) the associated event node names (numbers); 3) the duration times (in days) for each activity; and 4) the hourly labor costs for each activity. Assume two people are required to complete each task and that each works 8 hours a day.
For easy understanding, Activities are represented by Alphabet.

To estimate costs using the triangular model, you can follow these steps:

1. Clear the environment and command line using ‘clear‘, ‘clc‘, and ‘close all‘.

2. Define the simulation parameters, including the number of samples (‘num_samples‘) and the number of histogram bins (‘num_hist_bins‘).

```
%% Define the simulation parmeters
num_samples = 2000;
num_hist_bins = 51;

activity =  {'A' 'B' 'C' 'D' 'F' 'I' 'J' 'K' 'L' 'M' 'N'};
num_activities = size(activity, 2);

min_hours = [64 32 16 96 80 160 80 16 32 160 96];
max_hours = [128 128 112 192 240 400 192 48 96 320 176];
likely_hours = [96 64 48 144 160 240 144 32 48 240 144];

min_cost = [100 100 100 100 100 100 90 90 90 90 90];
max_cost = [150 150 150 150 150 150 110 110 110 110 110];
likely_cost = [120 120 120 120 120 120 100 100 100 100 100];
```

FIGURE 3.16: Snippet 1 of Cost Estimation Model

3. Define the inputs for each activity, including the minimum hours (‘min_hours‘), maximum hours (‘max_hours‘), and likely hours (‘likely_hours‘) for labor, and the minimum cost (‘min_cost‘), maximum cost (‘max_cost‘), and likely cost (‘likely_cost‘) for each activity.

4. Perform some dimension checks to ensure the input values are correct.

5. Define probability distribution samplers for labor hours and cost using the ‘makedist‘ function with the ’triangular’ distribution and the input parameters for each activity.

6. Run the Monte-Carlo simulation by generating random samples of labor hours and cost for each activity using the ‘random‘ function and the previously defined probability distribution samplers.

7. Calculate the total cost for each sample by multiplying the labor hours and cost for each activity.

```
% Some dimension checks to make sure values were inputted correctly
assert(size(min_hours, 2) == num_activities);
assert(size(max_hours, 2) == num_activities);
assert(size(likely_hours, 2) == num_activities);

assert(size(min_cost, 2) == num_activities);
assert(size(max_cost, 2) == num_activities);
assert(size(likely_cost, 2) == num_activities);
```

FIGURE 3.17: Snippet 2 of Cost Estimation Model

```
%% Define the probability distributions to sample data from

% Create probability distribution samplers for labor hours and cost
prob_dist_hours = cell(size(likely_hours));
prob_dist_cost = cell(size(likely_cost));

for ind = 1:num_activities
    prob_dist_hours{ind} = makedist('triangular', ...
        'a', min_hours(ind), 'b', likely_hours(ind), 'c', max_hours(ind));
    prob_dist_cost{ind} = makedist('triangular', ...
        'a', min_cost(ind), 'b', likely_cost(ind), 'c', max_cost(ind));
end
```

FIGURE 3.18: Snippet 3 of Cost Estimation Model

8. Calculate the cumulative running mean (CRM) of the project cost using the ‘cumsum‘ function and plot it against the number of samples.

9. Calculate the mean, standard deviation, and standard error of the project cost.

10. Calculate the 10%ile and 90%ile of the collected dataset using the ‘prctile‘ function.

11. Calculate the 10%ile and 90%ile assuming the data is distributed as a normal distribution using the ‘norminv‘ function.

12. Plot the histogram of the project cost distribution with the final mean, actual and

```
%% Run the Monte-Carlo Simulation

% rand('seed', 5);                    % Seed to control randomization
tic;

rand_hours = zeros(num_activities, num_samples);
rand_cost = zeros(num_activities, num_samples);

for ind = 1:num_activities
    rand_hours(ind, :) = random(prob_dist_hours{ind}, 1, num_samples);
    rand_cost(ind, :) = random(prob_dist_cost{ind}, 1, num_samples);
end

rand_task_cost = rand_hours .* rand_cost;
```

FIGURE 3.19: Snippet 4 of Cost Estimation Model

normalized 10%ile to 90%ile boundaries.

13. Print the computation results for the project cost mean, standard deviation, standard error, and 10%ile to 90%ile boundaries.

The objective of this Module is to perform a Monte Carlo simulation to estimate the distribution of project cost based on probability distributions for labor hours and costs for different activities. The code defines the simulation parameters, creates probability distribution samplers for labor hours and cost using triangular distribution, runs the Monte-Carlo simulation, calculates the mean, standard deviation, and standard error of the project cost, and plots the results in two figures: a histogram of the project cost distribution and a cumulative running mean of the project cost. Finally, the code prints the computation results, including the mean, standard deviation, and 10 percent to 90 percent boundaries of the project cost distribution.

```
rand_proj_cost = sum(rand_task_cost, 1);
sample_range = 1:num_samples;
cum_run_mean_proj_cost = cumsum(rand_proj_cost) ./ sample_range;
mean_proj_cost = cum_run_mean_proj_cost(num_samples);

toc;

std_dev_proj_cost = std(rand_proj_cost);
std_err_proj_cost = std_dev_proj_cost / sqrt(num_samples);

% Calculate 10%ile and 90%ile of the collected dataset
percent_10 = prctile(rand_proj_cost, 10);
percent_90 = prctile(rand_proj_cost, 90);

% 10%ile & 90%ile assuming data is distributed as normal distribution.
norm_percent_10 = norminv(0.1, mean_proj_cost, std_dev_proj_cost);
norm_percent_90 = norminv(0.9, mean_proj_cost, std_dev_proj_cost);
```

FIGURE 3.20: Snippet 5 of Cost Estimation Model

```
%% Plot the results

figure(1)
histogram(rand_proj_cost, num_hist_bins);
hold on
xline(mean_proj_cost, '--r');
xline(percent_10, '-.g');
xline(norm_percent_10, '-.c');
xline(percent_90, '-.g');
xline(norm_percent_90, '-.c');
hold off
legend('Histogram', 'Final Mean', ...
    '10%->90% Actual Boundary', '10%->90% NormInv Boundary');
title('Histogram: Project Cost Distribution');
xlabel('Project Cost in $');
ylabel('Count');
```

FIGURE 3.21: Snippet 6 of Cost Estimation Model

```
figure(2)
plot(sample_range, cum_run_mean_proj_cost);
hold on
yline(mean_proj_cost, '--r');
hold off
legend('CRM', 'Final Mean');
title('Cumulative Running Mean (CRM) of Project Cost');
xlabel('Num Samples');
ylabel('CRM of Project Cost in $');
```

FIGURE 3.22: Snippet 7 of Cost Estimation Model

```
%% Print the computation results

fprintf('Project Cost - Mean: %.2f\n', mean_proj_cost);
fprintf('Project Cost - Standard Deviation: %.2f\n', std_dev_proj_cost)
fprintf('Project Cost - Standard Error: %.2f\n', std_err_proj_cost)
fprintf('Project Cost - 10%% -> 90%% Boundary: (%.2f, %.2f)\n',...
    percent_10, percent_90);
fprintf('Project Cost - 10%% -> 90%% Norm Inv Boundary: (%.2f, %.2f)\n',...
    norm_percent_10, norm_percent_90);
```

FIGURE 3.23: Snippet 8 of Cost Estimation Model

Chapter 4

Simulation and Results

4.1 Airport Landing/Take-off Simulation Results

4.1.1 Graphical Outputs of M/M/2 Model

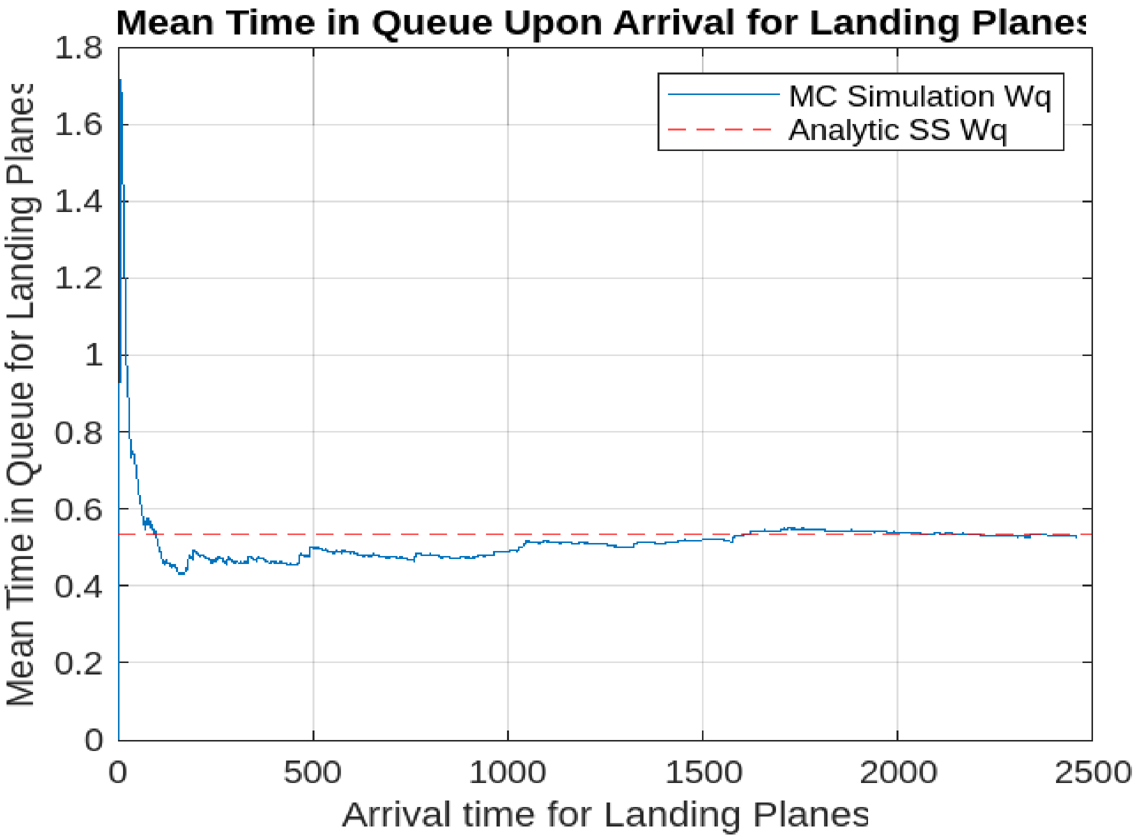

FIGURE 4.1: Mean time In Queue (landing)

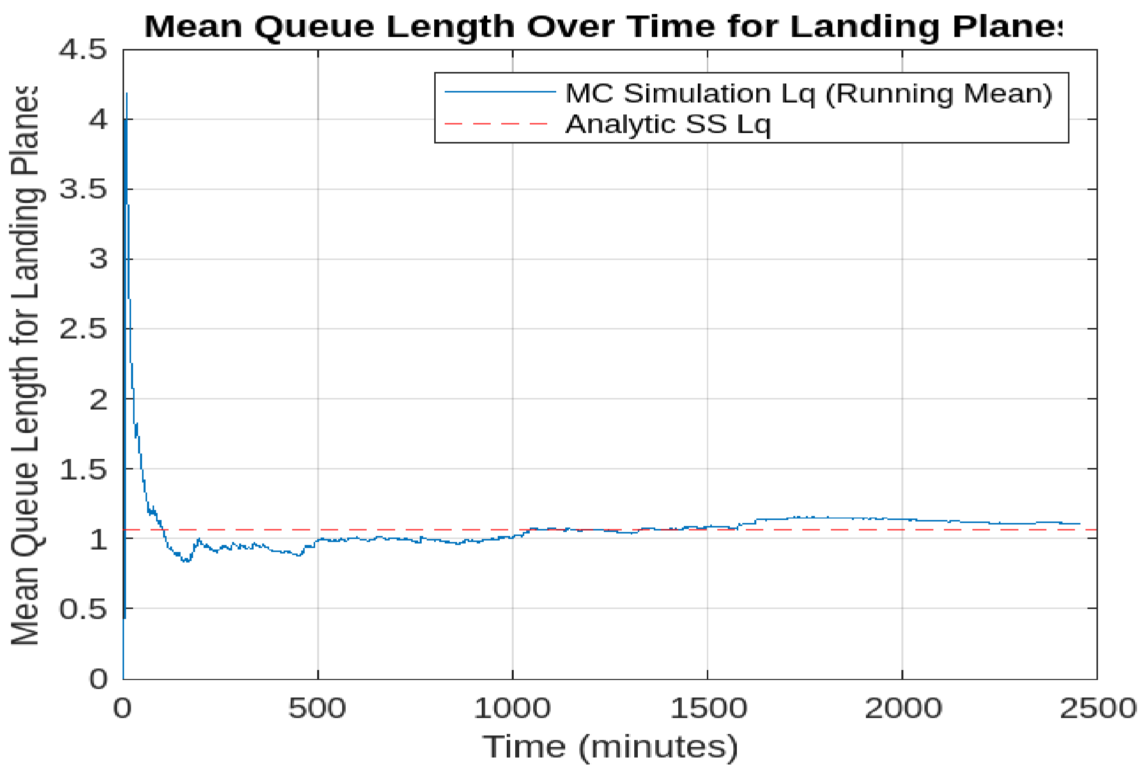

FIGURE 4.2: Mean Queue Length (landing)

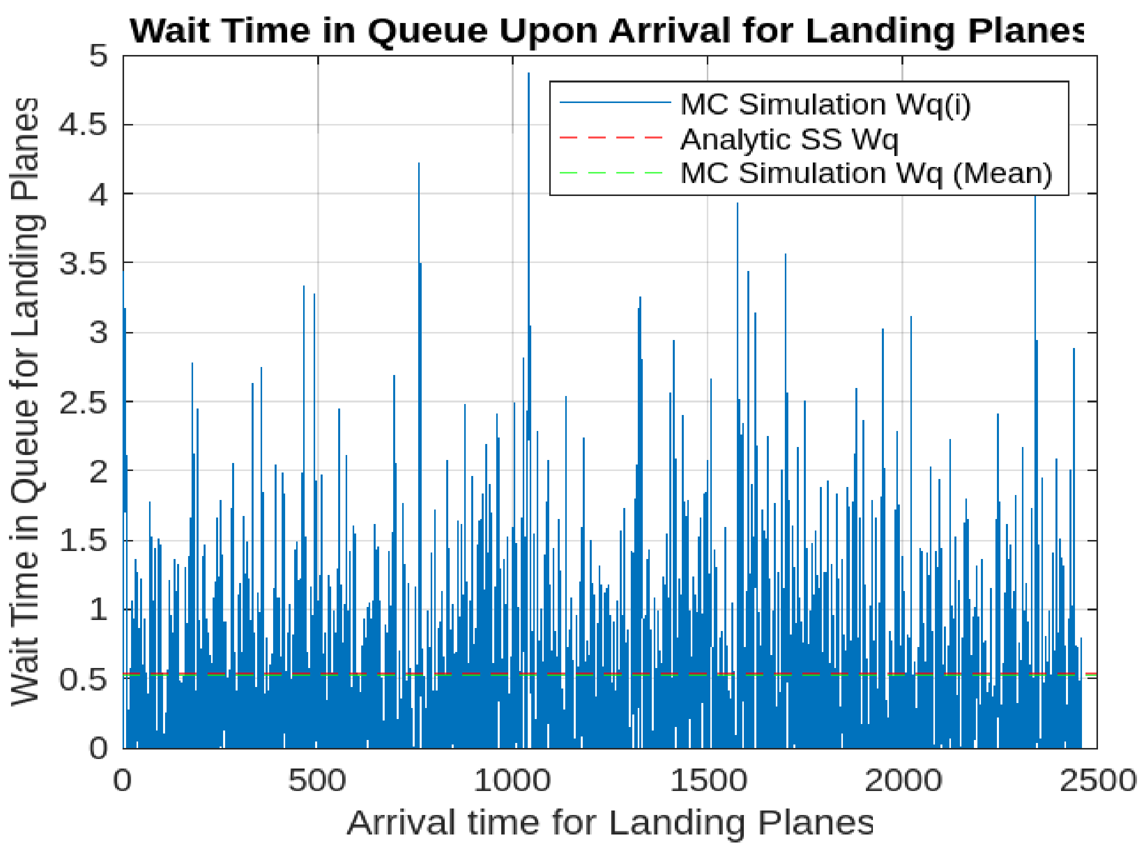

FIGURE 4.3: Wait Time In Queue (landing)

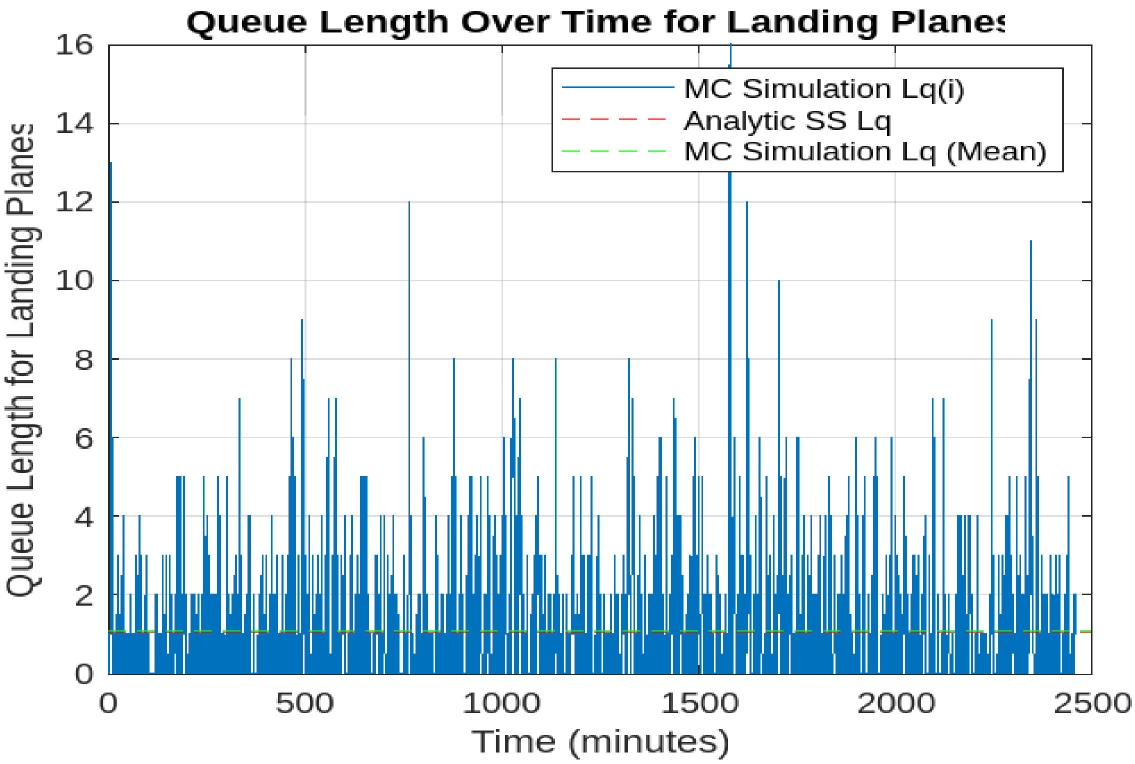

FIGURE 4.4: Queue Length Over Time (landing)

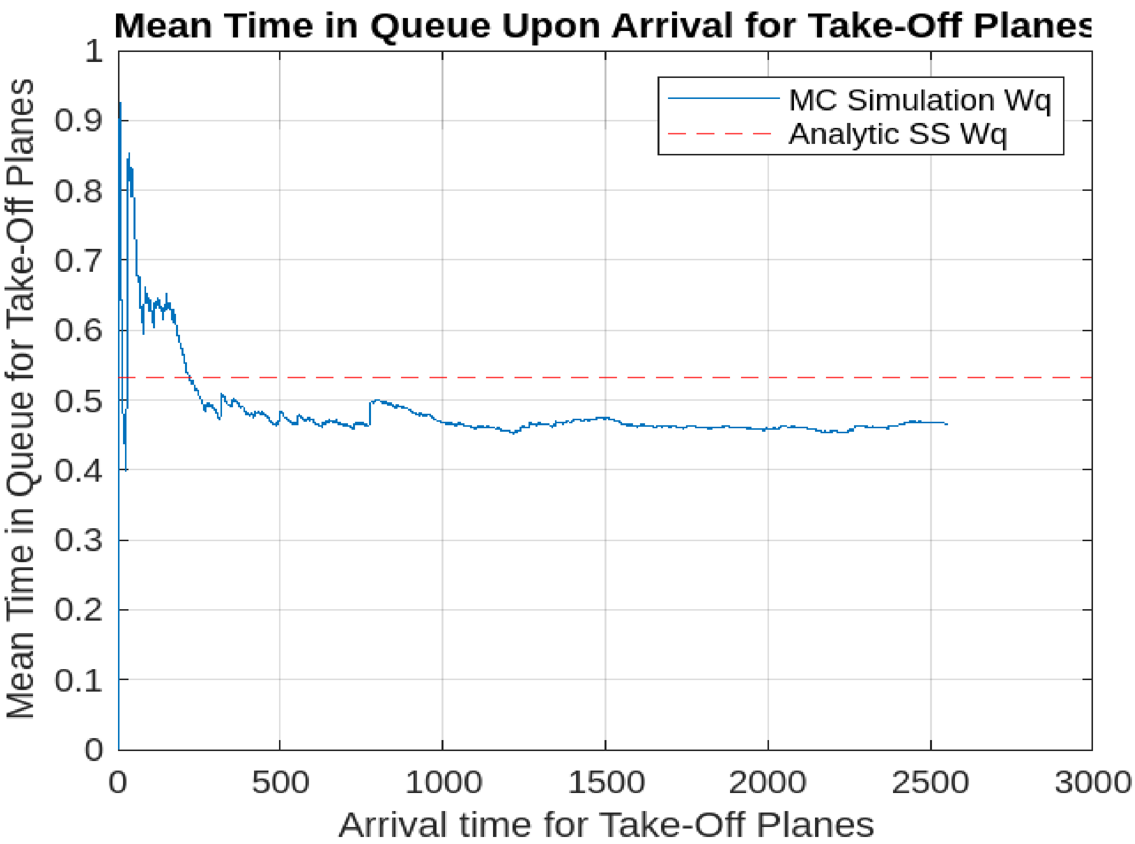

FIGURE 4.5: Mean Time In Queue (take-off)

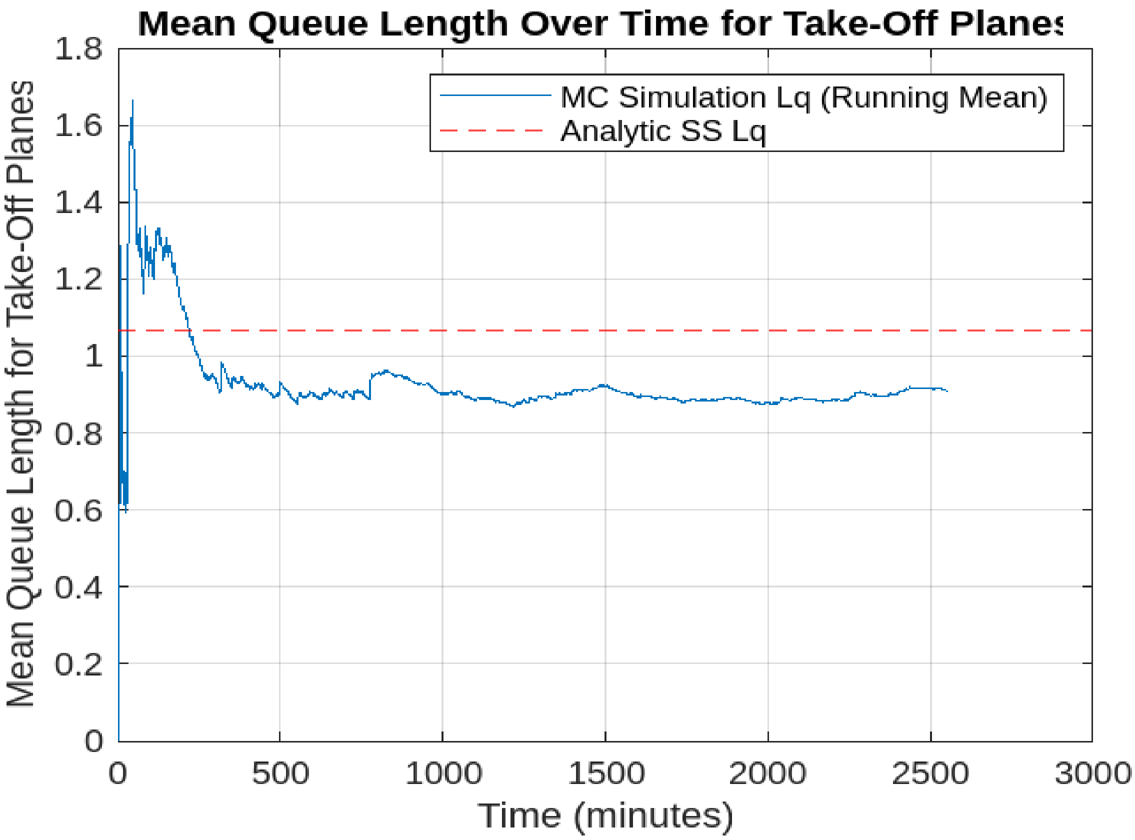

FIGURE 4.6: Mean Queue Length (take-off)

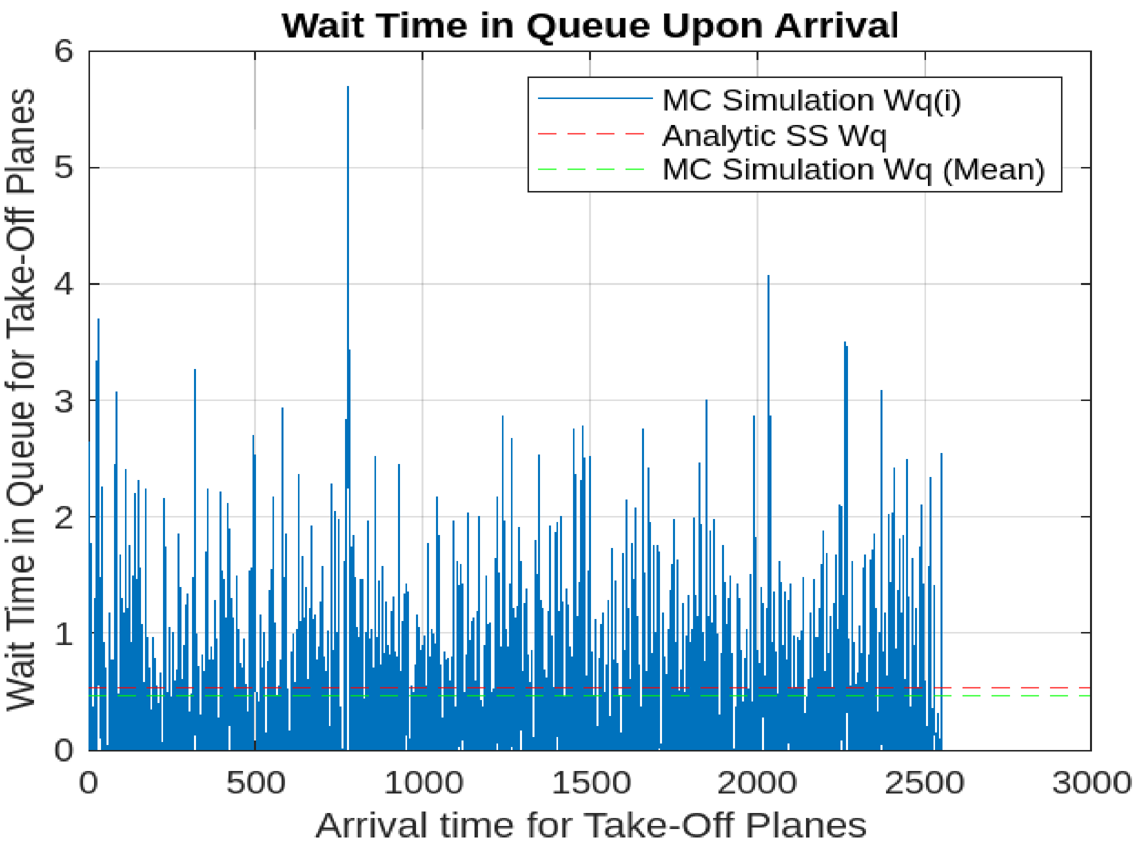

FIGURE 4.7: Wait Time in Queue (take-off)

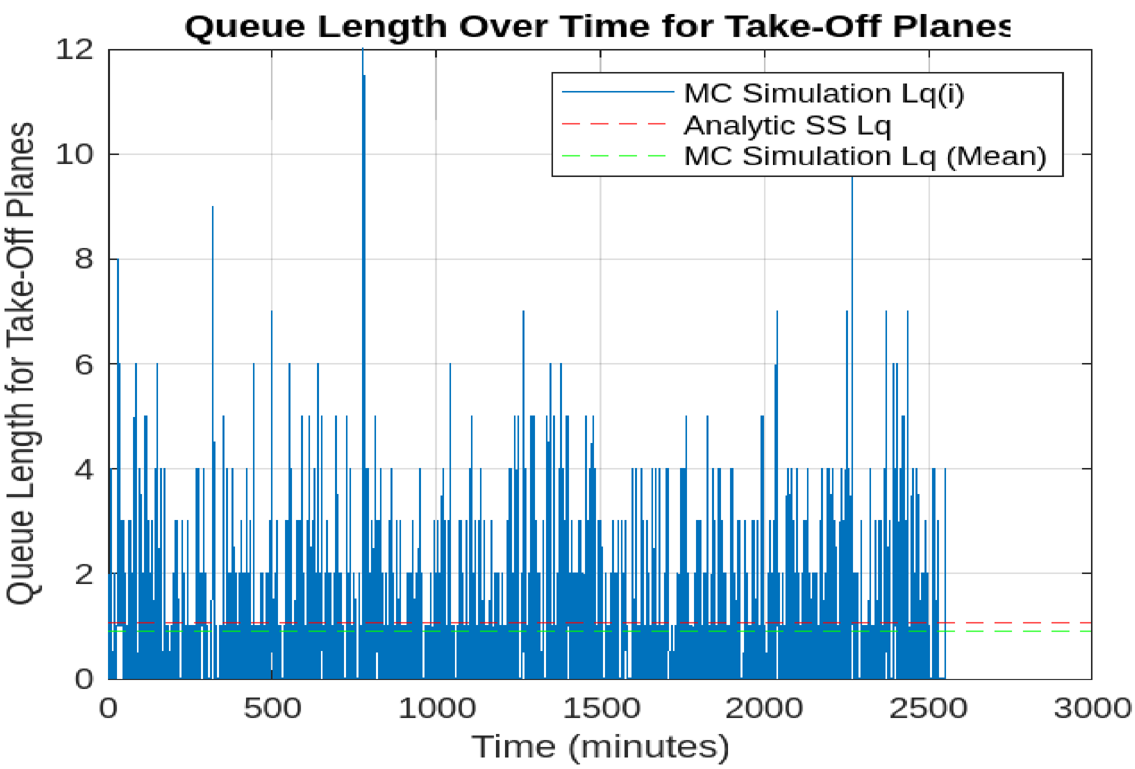

FIGURE 4.8: Queue Length Over Time (take-off)

4.1.2 Console Output:

Mean Queue Waiting Time(Wqmc) for Landing Planes: 0.53 minutes

Mean Queue Length (Lqmc) for Landing Planes: 1.10 planes

Mean System Time (Wmc) for Landing Planes: 0.95 minutes

STD Queue Waiting Time (Wqmc)for Landing Planes: 0.66 minutes

STD Queue Length (Lqmc) for Landing Planes: 1.71 planes

STD System Time (Wmc) for Landing Planes: 0.75 minutes

STE Queue Waiting Time (Wqmc) for Landing Planes: 0.01 minutes

STE Queue Length (Lqmc) for Landing Planes: 0.03 planes

STE System Time (Wmc) for Landing Planes: 0.01 minutes

FIGURE 4.9: Console

```
Mean Queue Waiting Time (Wqmc) for Take-Off Planes: 0.47 minutes

Mean Queue Length (Lqmc) for Take-Off Planes: 0.91 planes

Mean System Time (Wmc) for Take-Off Planes: 0.88 minutes


STD Queue Waiting Time (Wqmc) for Take-Off Planes: 0.61 minutes

STD Queue Length (Lqmc) for Take-Off Planes: 1.42 planes

STD System Time (Wmc) for Take-Off Planes: 0.71 minutes


STE Queue Waiting Time (Wqmc) for Take-Off Planes: 0.01 minutes

STE Queue Length (Lqmc) for Take-Off Planes: 0.03 landing planes

STE System Time (Wmc) for Take-Off Planes: 0.01 minutes
```

FIGURE 4.10: Console

4.2 Markov Chain Results

4.2.1 Graphical Outputs of Markov Chain Model

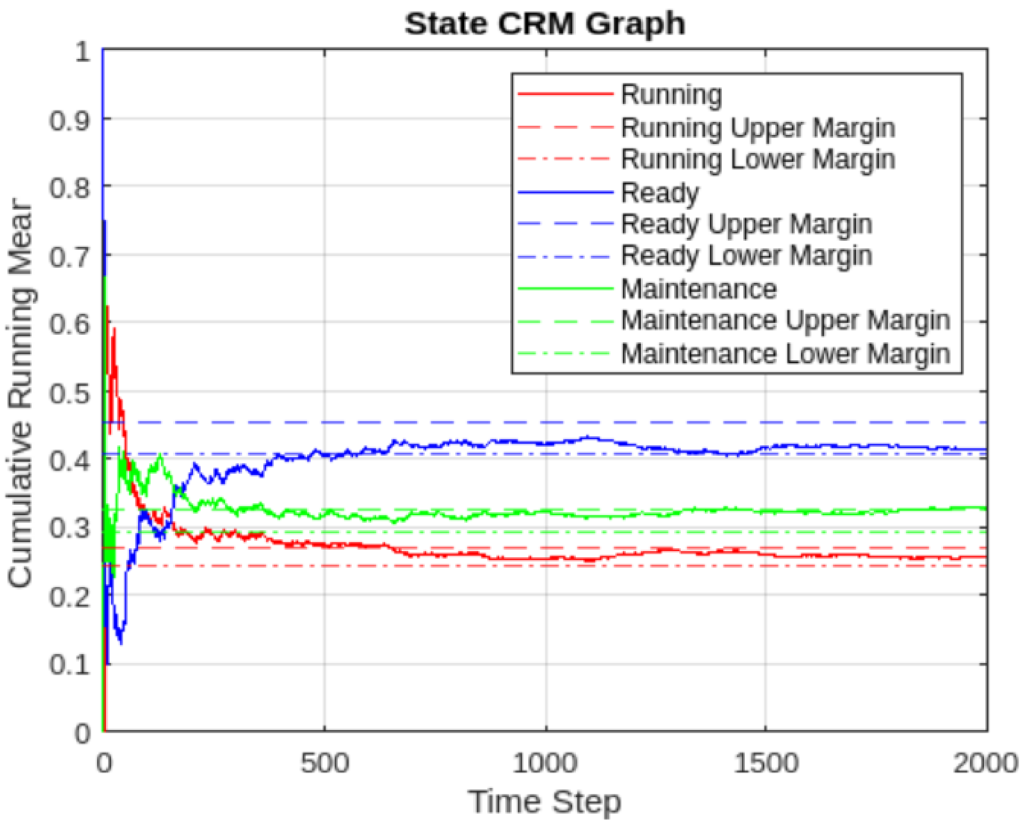

FIGURE 4.11: Markov Chain Simulation graph

4.2.2 Console Outputs of Markov Chain Model

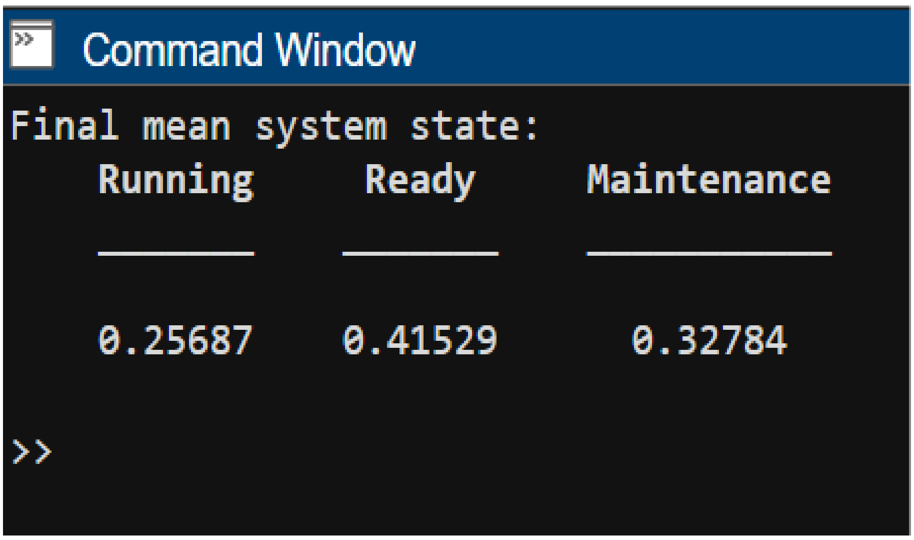

FIGURE 4.12: Simulation output

4.3 Triangular Model Results

4.3.1 Graphical Outputs of Cost Estimation

```
Elapsed time is 0.023609 seconds.
Project Cost - Mean: 159368.88
Project Cost - Standard Deviation: 10736.78
Project Cost - Standard Error: 240.08
Project Cost - 10% -> 90% Boundary: (145645.84, 173456.57)
Project Cost - 10% -> 90% Norm Inv Boundary: (145609.14, 173128.62)
>>
```

FIGURE 4.13: Cost estimation Result

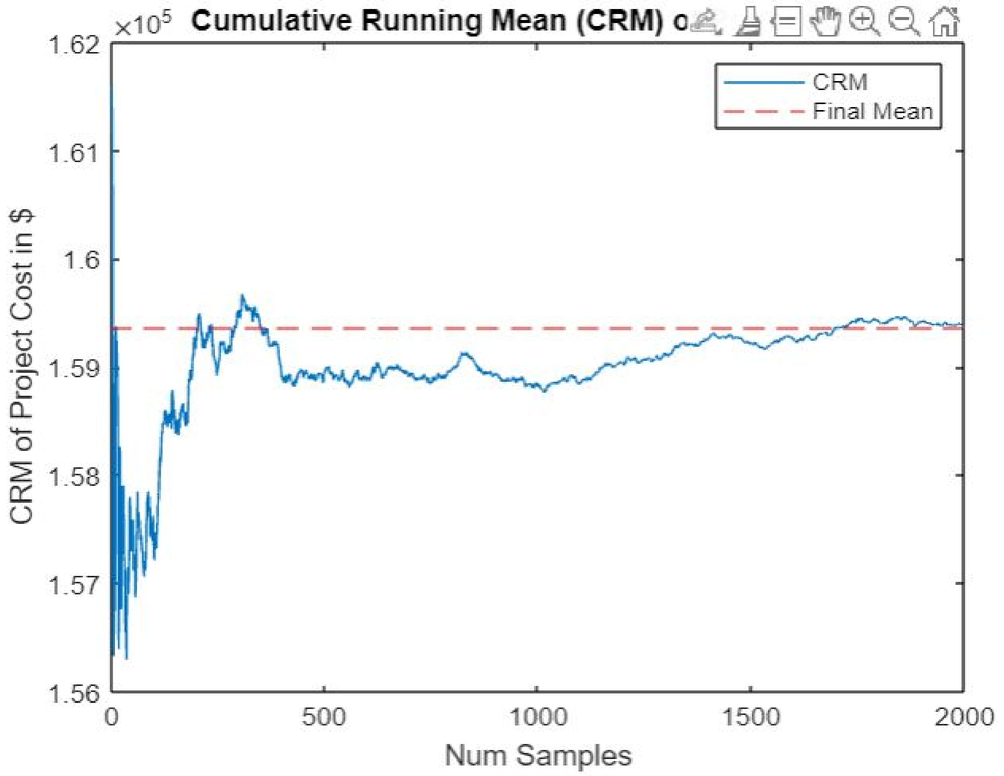

FIGURE 4.14: Mean cost after several simulations

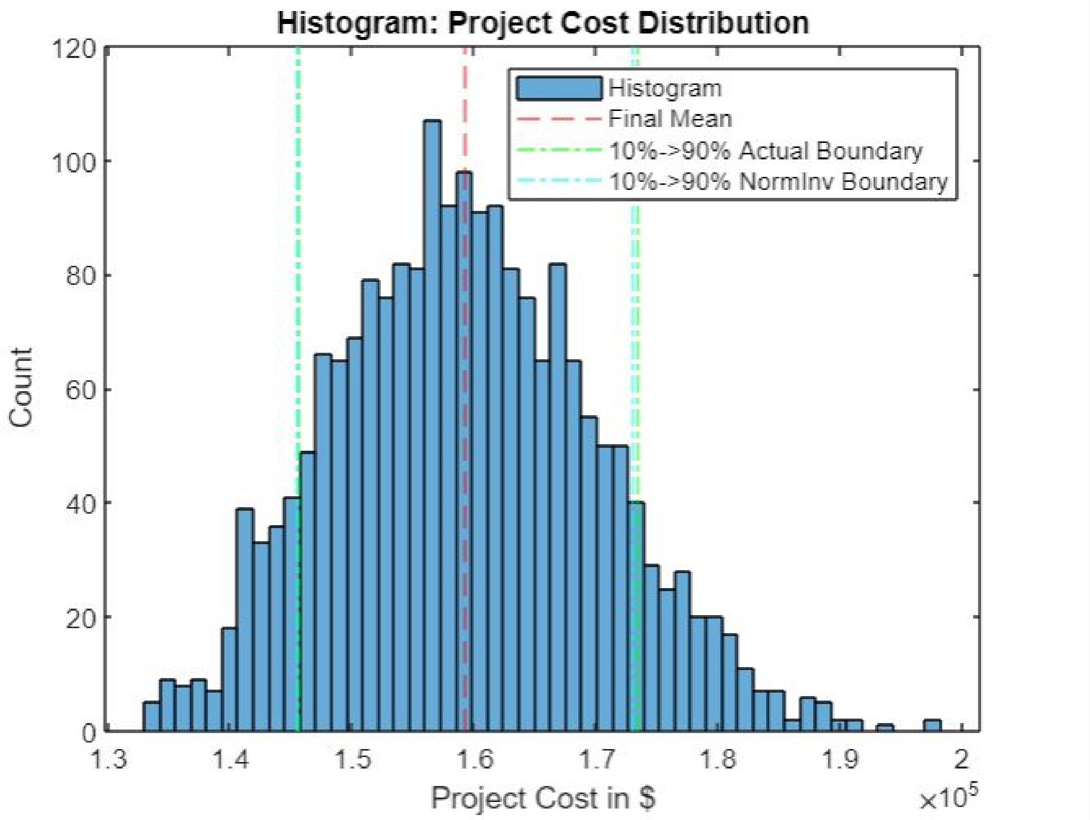

FIGURE 4.15: Cost Simulation Result

Chapter 5

Conclusions and Future Work

Airport Queueing System For Landing/Take-off of planes We calculated the following performance measures using M/M/2 system Mean Queue Waiting Time (Wqmc),Mean Queue Length (Lqmc), Mean System Time (Wmc), These values signify the overall plane experience and can be helpful for evaluating the system performance.The results were compared with the analytic steady-state values (which are obtained through several simulations in real-time) and finally, the plots show how accurate the simulation is in alignment with the steady-state values.

Monte Carlo simulation to generate a Markov Chain representing a system with three states: 'R', 'O', and 'M'.The Monte Carlo simulation generates a Markov chain by randomly selecting the next state based on the transition probabilities.The cumulative running mean (CRM) is computed to analyze the performance of the system. The CRM represents the average state of the system at each time step, indicating how long the system spends in each state.

Monte Carlo simulation to analyze the distribution of project costs. The simulation generates random samples of labor hours and costs using triangular probability distributions defined by the input parameters. It then calculates the cost for each task by multiplying the randomly sampled labor hours and costs. The project cost is obtained by summing up the costs of all tasks. So this as theory can be used to have an idea about the labor costs in our queueing system.

In terms of future work, by combining these systems, one can get a comprehensive understanding of system performance and project cost distribution. For example, the CRM simulation and analysis from the Airport Queueing system to evaluate the performance of the airport system over time and identify patterns or trends in planes waiting times. We can then integrate the project cost analysis from the MC simulation to assess the financial implications of different system performance scenarios. This can help in optimizing system parameters, such as resource allocation at the airport, to achieve a balance between operational efficiency, customer satisfaction, and cost-effectiveness.

Chapter 6

Bibliography

[1] J. A. Vrugt, ”Markov chain Monte Carlo simulation using the DREAM software package: Theory, concepts, and MATLAB implementation,” in IEEE Transactions on Geoscience and Remote Sensing, vol. , pp. 2020.

[2] G. Yan and H. Fang, ”A simple simulation method for queue system,” in ieee transaction,2019.

[3] Y. Huang and S. Kuang, ”M/M/n/m Queuing Model under Non-preemptive Limited Priority,” in IEEE Transactions, vol. 10, no., pp. [10-15],2015.

[4] F. Mahfuz, ”Markov Chains and Their Applications,” Master’s thesis, University of Texas at Tyler, Scholar Works at UT Tyler, Math Theses, Spring, 4-28-2021.

[5] S. Jannat and A. G. Greenwood, ”Estimating Parameters of the Triangular Distribution Using Nonstandard Information,” in Proceedings of the 2012 Winter Simulation Conference, C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, Eds.,2020.

[6] Computer Network Systems: Queueing Theory and Performance Evaluation, T. G. Robertazzi,Springer, 3rd Edition, 2000, ISBN-10: 0387950370.

[7] Discrete Events System Simulation, J Banks, J. S. Carson II, B. L. Nelson, D. M. Nicol, Prentice Hall,5th Edition, 2009, ISBN-10: 0136062121.