MODEL-BASED APPROACH TO THE UTILIZATION OF HETEROGENEOUS NON-OVERLAPPING DATA IN THE OPTIMIZATION OF COMPLEX AIRPORT SYSTEMS

A Dissertation
Presented to
The Academic Faculty

By

Harold Nemo Adodo Nikoué

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Engineering
Department of Aerospace Engineering

Georgia Institute of Technology

December 2021

MODEL-BASED APPROACH TO THE UTILIZATION OF HETEROGENEOUS NON-OVERLAPPING DATA IN THE OPTIMIZATION OF COMPLEX AIRPORT SYSTEMS

Thesis committee:

Professor John-Paul Clarke
School of Aerospace Engineering and School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor David Goldsman
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Eric Feron
School of Aerospace Engineering
Georgia Institute of Technology

Professor Brian German
School of Aerospace Engineering
Georgia Institute of Technology

Professor Kemal Dinçer Dingeç
Department of Industrial Engineering
Gebze Technical University

Date approved: Friday 3rd December, 2021

Créer, c'est aussi donner une forme à son destin.

Albert Camus

Pour maman et tata Doris,

Vous me manquerez toujours.

ACKNOWLEDGMENTS

First and foremost, I would like to thank the people without whom none of it would have been possible. My mother, Evelyne Nikoue, and my aunt, Dr. Doris Nikoue, who will never get to read my dissertation. I would then like to thank my close friends and the rest of my family, especially my sister who kept me above waters while pursuing my career as a professional student.

I would like to give thanks to my research advisors, Professor John-Paul Clarke and Professor David Goldsman. Professor Clarke's ability to get to the essence of a problem and create new interpretations and avenues for research has challenged me and inspired me. Professor Goldsman's continued support and explanations have made it possible to improve the quality of my simulations and my understanding.

I would also like to thank the other members of my committee. Professor Eric Feron provided me ample feedback on the first section of the thesis and more importantly much encouragement to finish my thesis. Professor Brian German gave me guidance at an early phase of my graduate studies. Professor Kemal Dinçer Dingeç was instrumental in the development of the more theoretical material in the second part of the thesis.

In the many years I have spent at Georgia Tech as a graduate student, many graduate students have shaped my work and education. I would like to thank Yassine Ridouane, Jean-Guillaume Durand, Lauren Brads, Fanruiqui Zeng, Andrew Kendall, Jose Magalhaes, Joseph Bakthiar, James Gloyd and Gustavo Lee. Our multiple conversations have made made life as a graduate student more enjoyable and valuable. I owe special thanks to Aude Marzuoli who challenged how I was thinking of my initial work and made me a better researcher. Although I have never met him in person, I am deeply indebted to Terran Melconian for raising my software design skills. His repository for the M.I.T. Extensible Air Network simulation has exposed me to clean code principles, good code commenting and showed that code collaboration can be done across decades.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ACO Ant Colony Optimization

AR Autoregressive

ARIMA Autoregressive Intergrated Moving Average

ARMA Autoregressive Moving Average

ARTA AutoRegressive to Anything

cdf cumulative distribution function

CDR Correct Detection Rate

CE Cross-Entropy Method

CPD Change-point Detection

DEDS Discrete-Event Discrete-Time Simulation

DIMIA Department of Immigration and Multicultural and Indigenous Affairs

EDA Estimation of Distribution Algorithm

far false alarm rate

FIDS Flight Information Display System

KL Kullback-Leibler

m.g.f. moment generating function

MFCQ Mangasarian Fromovitz Constraint Qualification

MLE Maximum Likelihood Estimation

MRAS Model Reference Adaptive Search

NOBM Non-Overlaping Batch Mean

ROC Receiver Operating Characteristic

SFO San Francisco International Airport

SYD Sydney Kingsford Smith International Airport

SUMMARY

Simulation and optimization have been widely used in air transportation, particularly when it comes to determining how flight operations might evolve. However, with regards to passengers and the services provided to them, this is not the case in large part because the data required for such analysis is harder to collect, requiring the timely use of surveys and significant human labor. The ubiquity of always–connected smart devices and the rise of inexpensive smart devices has made it possible to continuously collect passenger information for passenger-centric solutions such as the automatic mitigation of passenger traffic. Using these devices, it is possible to capture dwell times, transit times, and delays directly from the customers. The data; however, is often sparse and heterogeneous, both spatially and temporally. For instance, the observations come at different times and have different levels of accuracy depending on the location, making it challenging to develop a precise network model of airport operations. The objective of this research is to provide online methods to sequentially correct the estimates of the dynamics of a system of queues despite noisy, quickly changing, and incomplete information. First, a sequential change point detection scheme based on a generalized likelihood ratio test is developed to detect a change in the dynamics of a single queue by using a combination of waiting times, time spent in queue, and queue-length measurements. A trade-off is made between the accuracy of the tests, the speed of the tests, the costs of the tests, and the value of utilizing the observations jointly or separately. The contribution is a robust detection methodology that quickly detects a change in queue dynamics from correlated measurements. In the second part of the work, a model-based estimation tool is developed to update the service rate distribution for a single queue from sparse and noisy airport operations data. Model Reference Adaptive Sampling is used in-the-loop to update a generalized gamma distribution for the service rates within a simulation of the queue at an airport's immigration center. The contribution is a model predictive tool to optimize the service rates based on waiting time information. The two

frameworks allow for the analysis of heterogeneous passenger data sources to enable the tactical mitigation of airport passenger traffic delays.

CHAPTER 1
MOTIVATION AND PRIOR WORK

1.1 On the need of simulations for Queuing Systems

Queues occur whenever a single or a group of entities have to wait their turn for service. The accumulation of entities at one resource leads to delays, the loss or cancellation of some jobs and in some extreme cases to the complete breakdown of a system such as a theme park, cash registers at a supermarket, a group of cloud servers handling disparate jobs or a call center. Ultimately, the user is the one who gets penalized by failures of these systems. It is desired to understand the dynamics of queues in order to mitigate the adverse effects of delays and provide adequate levels of service to a user. Several techniques exist to model, analyze, design and control all types of queuing system. At the source of the tools to understand queues is Queuing Theory. Queuing Theory [1] is a branch of stochastic processes [2],[3], [4] dedicated to the analysis of stochastic flow systems encompassing discrete and continuous Markov Chains, Phase-type distribution[5], ladder processes and queuing networks such as the closed Jackson network. It is used to explain the long-time and time averaged behaviors of queues. Most queuing systems; however, experience sudden spikes in traffic or drops in demand that cannot be explained by steady-state or asymptotic analysis. To account for these transient changes, one needs a faithful simulation of the system under study.

Stochastic simulations of queues makes it possible to know what happens to the system on average but also for rare long-tailed events by looking at the distributions of the output metrics rather than a single number. Although they can be time intensive to create and calibrate, they are crucial in the analysis, design and control of queuing systems. By nature, air transportation is very prone to disruptions that can spawn from seasonal events, inclement

weather, labor union strikes, breakdowns of the reservation centers or even in rare instances of the planes themselves. Stochastic simulations are sometimes the only tools for tactical decision making.

1.2 Queues and Air Transportation

The Bureau of Transportation Statistics [6] under the U.S. Department of Transportation collects and makes available flight performance statistics such as flight delays, flight cancellations and the number of passengers carried by flight. Not only is that data made publicly available as monthly reports, it is also used to penalize airlines in case of excessive delays. According to the FAA, passengers are entitled a compensation for excessive delays such as large arrival delays (arrival delays exceeding 45 minutes) and tarmac delays. Tarmac delays are not allowed to exceed 3 hours for domestic flights and 4 hours for international flights. All of this monitoring and penalties are based off the air transportation system from a flight perspective and not from a passenger perspective. Yet passengers are the ones disproportionally affected by flight delays [7]. Before 2010, there hasn't been many studies that have focused on mitigating air transportation delays from a passenger perspective.

Passenger-centric research is recent. In its seminal book on airport planning, Odoni [8] has looked at passenger terminals from an airport designers. He has looked at sizing the terminal areas for the anticipated traffic and encouraged the use of queuing theory to model airport flows. It is not until the Total Delay Impact study of 2010 [7] led by Barnhart et al that a complete accounting of passenger experienced delays has been performed. The study of passenger information has historically been difficult due to the cost of directly capturing passenger information. Collecting information about transit times throughout an airport or waiting times at specific choke points historically has required getting people on the ground to perform costly punctual surveys. Even when available, passenger data can be held in silos. Retailers hold passenger transactions for their personal tracking. Immigration services often collect throughput information to monitor their services. Airlines only have

access to their passenger information. The ubiquity of smart phones and smart devices in the last twenty years has made it possible to directly obtain passenger information through location sharing and voluntary information sharing. New information is collected thanks to biometric sensors at airports, information sharing from third-party beacons, location data from the bluetooth and wi-fi beaconss as well as passenger satisfaction through social websites such as Twitter. This diverse and heterogeneous data is feature-rich and provide direct information about the passengers. This rise has ushered the development of data-based studies on passenger movement throughout the airspace. Marzuoli [9], [10] looked at the aftermath of the Asiana Flight 214 crash at SFO and how passengers could have been re-accommodated at nearby airports or on other modes of transportation to limit the impact of the crash on the National Airspace System (NAS) and reduce the time it took to restore regular operations at SFO and nearby airports. Social media was used to complement the information officially released by airport authorities in order to find which airports were active and what was their updated capacity. In 2019, the same author leveraged cellular location data and social media data to show how cellular data can be used to monitor the health of the national airspace [11]. In a recent paper, Monmousseau compared the times it takes to clear immigration across multiple airports based off CBP data [12] and developed regression models to predict wait times based on key characteristics such as the day of the year, the day of the week or the number of passengers in the previous hour.

Now that the data is captured from the passenger, it is possible to make decisions that affect the passenger directly. Typical landside air traffic decisions are flight based such as departure metering and gate assignment. Departure metering is the process of limiting takeoff and arrivals to limit congestion in the airspace. Gate assignment is the process of assigning flights to different gates to make it easier for crew and passengers to make their connections and thereby abasing potential time conflicts. Another way to limit disruptions to passengers is to directly use the passenger information to match what is happening in the air. Queues can be staffed to match big arrivals or big departures and mitigate passenger

delays. Passengers can also be routed individually to limit the risks of missed connections. This requires the use of complete airport passenger simulations. Passenger-centric information is far from perfect, often having missing and incomplete data. Passenger location sensors such as Blue-tooth or Wi-Fi beacons may leave wide gaps in their coverage of airport terminals making it difficult to pinpoint the location of every passengers. The quality and quantity of the data can also vary in time by time of day or time of the year. Moreover, even a well-calibrated model of airport operations may drift over time due to the changing nature of passenger traffic. This can be the case at to the beginning or end of a busy congestion period. Therefore, one must be able to accommodate for missing and incomplete data as well as detect abrupt changes in the observations. The thesis addresses multiple questions related to the processing of missing or incomplete data for the simulation of passenger traffic and for the detection of change points in the data itself. Some of the questions are:

- How can we use the available observations of the queue to compensate for missing data?
- How can we use the simulation to correct for the inherent noise in the observations?
- Are some observations more valuable to detect drifts in the simulation model?
- Can we quickly detect the start or stop of a congestion period for highly correlated queue observations?
- Can multiple observations be combined to better detect changes in the dynamics of a queue?
- How many observations or combinations of observations are needed before a change can be declared?

The contributions of this dissertation are

1. A framework for the in-the-loop optimization of service rate distribution based on cycle time observations

2. A robust method to detect change points in a time series of correlated M/M/1 observations using Non-Overlapping Batch Means (NOBMs) and a generalized likelihood ratio test

3. A parallel test to detect change points in a queue traffic intensity based on any combination of waiting times, time spent in queue and queue length observations

1.3 Sydney International Airport Simulation

The first model to be built is a simple model of queueing at immigration service desks at Sydney International Airport. We focus on the segment of passengers' walk from their arrival at the gates to their departure from the immigration zone. It is assumed that the immigration services act as a single server queue with a time-varying service rate.

1.3.1 Airport Description

Sydney Kingsford Smith International Airport (SYD) is Australia largest airport, and the 38th busiest airport in the world in 2015 with 36 million passengers per year [13]. SYD Figure 1.1 has two piers, Pier B and Pier C which each hosts an immigration zone where passport control occurs. SYD is physically isolated from Sydney domestic airport, located 2.3 miles away. All the flights scheduled are the airports are international flights, and last more than 2 hours. Because of the long flight times, arrival time predictions can be made with great certainty two or more hours in advance, as the knowledge of departure times greatly reduces the arrival times uncertainty. Assuming arrival time is known two or more hours in advance, the distributions of staff at the airport can be adapted tactically in anticipation of future demand. This unique situation makes it possible to change staffing levels in advance of a anticipated demand with great efficacy.

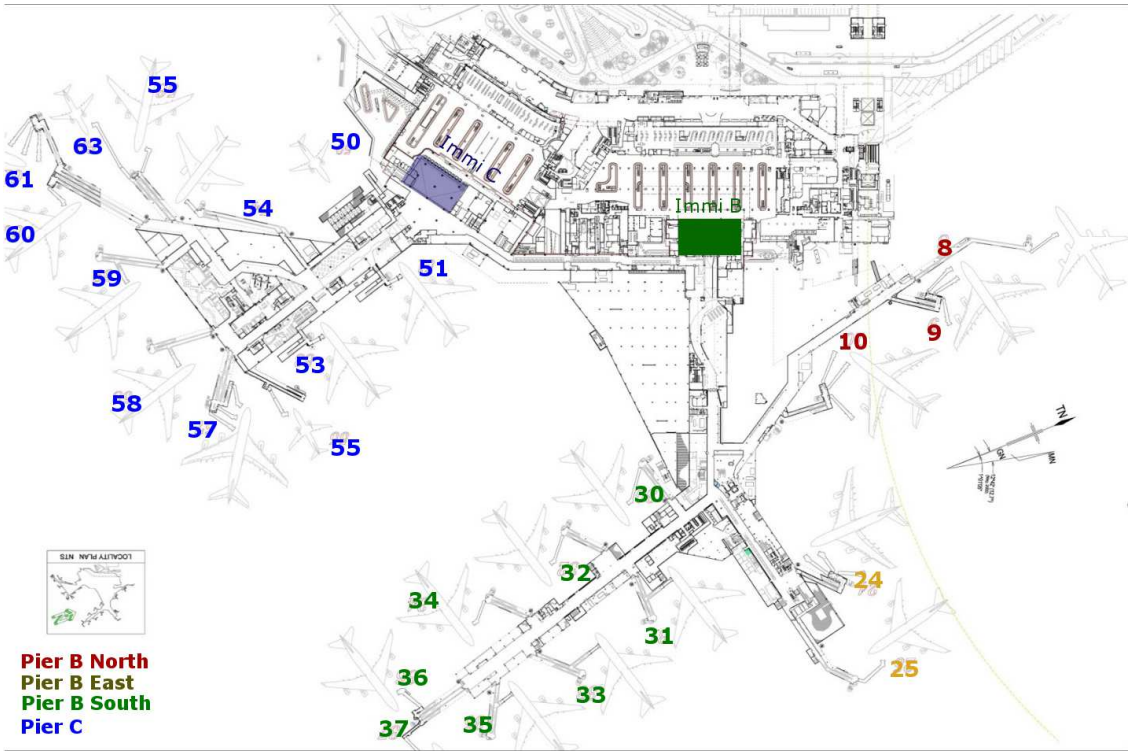


Figure 1.1: Sydney International Airport - Arrivals

1.3.2 Data Sources

This section describes the three main sources of data in this study. The section also addresses the inconsistencies in some of the databases, potential applications, and the calibration of the input to the simulation.

Data Description

In 2012, 12,369,193 revenue passengers flew from or to Sydney International Airport [14]. Data concerning passengers and flights was collected at Sydney International airport by the Sociéte Internationale de Télècommunications Aéronautiques (SITA) and the airport authorities from January 2012 to May 2013. The data included flight information in the form

of Flight Information Display System (FIDS), Cell phone geo-locations from MAC device time stamps, and immigration records from the Australian Department of Immigration and Indigenous Affairs DIMIA. Table Table 1.1 describes the different databases, their content and their size.

Table 1.1: Data sources available on Sydney Airport.

Label	Description	Date Range	Size
DIMIA	Passenger time stamps at immigration for each border crossing (DIMIA)	Jan. 2012 - May 2013	15,461,430 passengers (6,756,997 arrivals and 8,704,433 departures)
FIDS	Arriving and departing flight information including block, scheduled, estimated times and flight number	Jan. 6th - Dec. 1st 2012	578,104 (287,447 arrivals and 290,656 departures)
DWELL	Wi-Fi enabled devices tracking data for each location (x,y) triangulated zone time stamp	Jul. 1st - Dec. 12th	2,047,235 unique device IDs (827,474 arrivals and 1,236,372 departures overlapping)

FIDS database

Flight Information Display System (FIDS) dataset contains information on gate, block, as well as estimated and scheduled times for departing and arriving flights at Sydney International Airport. That data provides the simulation with the initial arrival time of incoming passengers in the airport. The records do not specify at what time the Estimated Time of Arrival (ETA) was recorded, nor how many times it was modified. Neither does it include the number of passengers on the flight. From the DWELL dataset, seasonality of flight delays, sources of airside flight delays, as well as taxi times distributions can be studied. The DIMIA dataset was used to complement the dataset, and provides an estimated passenger count for each flight.

DIMIA database

Passenger time stamps at immigration were recorded by the Australian Department of Immigration and Multicultural and Indigeneous Affairs (DIMIA). The DIMIA dataset consists of border crossing activities for 2012 at SYD. During these activities, passenger's nationality, time stamps at immigration services, origin or destination airports of passengers, processing desk identifier number, and the flight number were recorded manually. DIMIA served two purposes in the simulation. It provided a reliable estimate of service rate, and it was also used to complement the flight information. By matching a single passenger record in DIMIA to a flight, it was possible to compute the average number of passengers per flight, and fill-in the missing load factor from FIDS. It allowed us to generate a distribution of passenger occupancy per flight ID.

The dataset is also used to determine the service rate at each immigration desk at any hour during any day of the week. Note that each day of the week has a specific service rate distribution. Since every DIMIA record contains the processing desk ID along with the time at which a passenger was processed, the number of unique open desks could be retrieved for any given time period. The service rate per desk is then obtained by dividing the number of passengers processed by the number of open desks for a 15 minutes time period.

The limitation of the dataset is the inconsistency of the manual recordings. Some entries are missing. In some cases, tail numbers are recorded in place of flight numbers, if that information is present. Some individual days in October and December are also missing from the data as shown on Figure Figure 1.2, which shows the number of datapoints in the DIMIA datasets in 2012.

Device ID connection times in the airport (DWELL) database

The DWELL database consisted in the time stamp at which a Wi-Fi enabled device contacted a Wi-Fi access point inside the airport. Any Wi-Fi enabled device continuously probes its surrounding at discrete nonuniform intervals to find available signals, and eval-

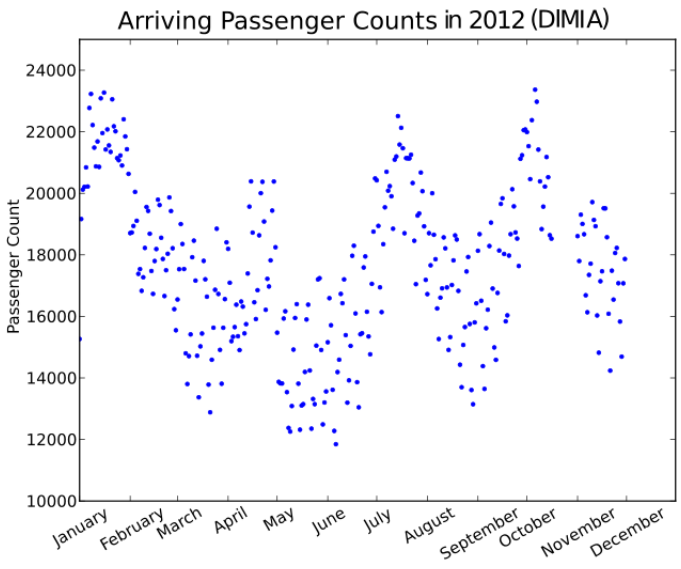


Figure 1.2: DIMIA counts of arriving passengers stamps at Immigration in 2012

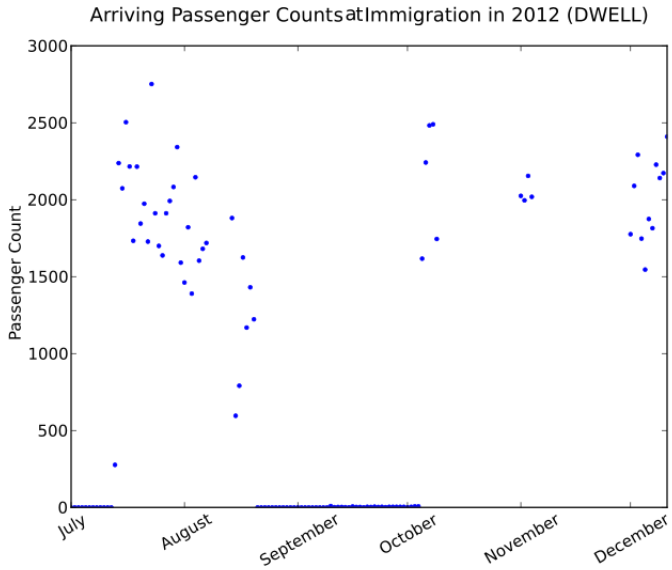


Figure 1.3: DWELL counts of passengers recorded in Immigration zones in 2012

uate their strength. SITA created a network of more than 400 Wi-Fi access points, 130 people-counters and 50 Bluetooth sensors spread throughout Sydney International Airport's terminals [**SITA:2012**]. Using iflow tool, each device was given a unique device ID, and each time stamp was assigned to a group of airport locations. A device was typically assigned to more than four zones of the airport. Furthermore, because devices do not connect to internet periodically, only few of the MAC addresses were observed more than once.

The devices which were observed more than once could have as much as an hour delay between observations. A MAC address is a unique device ID that is being communicated to an internet service transmitter over Wi-Fi every time the device proves its surrounding. Wi-Fi positioning provided by SITA to locate passengers inside the airport. Wi-Fi positioning can achieve up to 15 meters in accuracy for conventional systems. It has the advantage of only using the resources already set in place at most airports to provide internet services. Position is computed by triangulation every time the cellular device probes its environment to determine the availability and strengths of surrounding signals. The sampling rate is infrequent varying from 2 seconds to several minutes depending on the particular device, and battery life. From the provided DWELL tables, only devices with more than one observation, and observation time intervals smaller than 15 minutes were considered.

As with the DIMIA data, the quantity of available DWELL data was inconsistent throughout the year, and between airport zones. It can be observed in Figure Figure 1.3, that the DWELL data lacked information for most days in August, September and October. Although all the days of those months are present in the database, only few data points remained when only the paths between immigration zones and all the gates are kept. The number of records varies widely between zone, and from one gate to the next. Gate 8 for example has more than 180,000 single hits, while gate 50 has only 405 hits. These numbers just take into account the zones for the entire time period, without focusing on paths or even on the repetition of the same device ID. If only full paths are considered, the number of records are drastically reduced. In this context, a path is a sequence of device identifiers associated to at least one time stamp at a gate and one at the closest immigration zone.

1.3.3 Airport Model

Subsection subsubsection 1.3.3 outlines the general structure of the simulation model. Subsection **??** reviews the principles behind the discrete-event simulation used in the model.

Model Structure

We only model passenger walk from gates to immigration. Because of the unique characteristics of SYD, we assume that the schedule of flight arrivals at gate is known exactly.

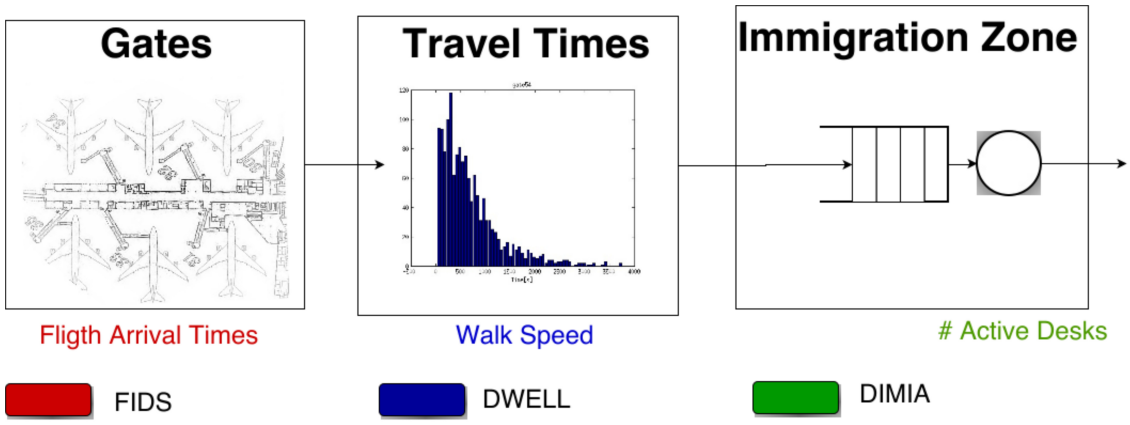


Figure 1.4: Structure of the arrival process model

Figure Figure 1.4 depicts the arrival process for passengers at the airport, with the data sources below each subsystem. The arrival process starts with the arrival of flights at their gates. Passengers then immediately leave the gates, and walk to the nearest immigration zone, where they enter a queueing process. They leave the system after being served at an immigration desk. The immigration zone is treated as a single queue with multiple servers with independently identically distributed service rates. To calibrate the model, we then need to define the distribution of walk times from gates to the immigration zones. The service rate per server is estimated based on a historic database.

Simulation process

Arrivals and departures of passengers in the system occur at discrete time steps. For this reason, a discrete event-based simulation was chosen to simulate the dynamics [**Leemis2004Discrete**]. In a Discrete-Event Discrete-Time Simulation (DEDS), time progresses directly from one scheduled event to the next scheduled event. An event for the simulation can be the arrival of a flight, an arrival of a passenger at immigration, or a departure from the immigration zone. The simulation starts with the schedule of flights. Each scheduled events are recorded in a time-ordered Future Event List (FEL)[15]. The FEL contains the start time and duration of each event in the simulation. After each event, the state variables are updated. Since the schedule of flight is known, the randomness in inter-arrival times at the service node come from the walk-time distribution. The service node includes a queue and a time-varying number of servers. The service discipline of servers is First-Come First-Served (FCFS). When a new desk becomes active, the next customer in front of the queue is served. A desk can only close after finishing serving a customer. Hence, the activation or deactivation of a desk does not affect the order of the immigration queue.

1.4 Modeling Input Distribution

The simulation uses the data presented in section subsection 1.3.2 as input to the simulation model. To describe the simulation, we explain how the different datasets were processed to generate input for our model, and we analyze the output of the data for the validation set.

1.4.1 Input Analysis

The two input distributions to the simulation are arrival times and service rates at queuing services. Walking speeds were computed from the input database by approximating passenger walk times from DWELL data. Service rate estimates were first generated from throughput then updated using the process highlighted in the next chapter. Each input

was analyzed from a subset of one or multiple of our data-sets, and validated against the remaining data.

Arrival Rates

Walk times are used to compute the inter-arrival times between two passengers at Pier B and Pier C immigration zones. By computing the inter-arrival times, we can estimate the arrival rates at the two immigration zones by time of day. As mentioned above, the DWELL database corresponds to discrete time-stamps of device unique ids inside the airport together with a list of the most likely location the passenger would have been located at the time of the recording. Walk times are formed by isolating device ids that had time stamps at one of the immigration zones (Pier B or Pier C) and another time stamp at one of the gate, then computing the time difference. When analyzing the available DWELL data, no data was observed at the immigration zone located in Pier-B. This implies that no walk-times could be directly estimated from our data for gates in Pier-B North, East or South. In other words, out of the 24 gates at the airport, only 8 gates could be used for walk time estimation. The approach we decided to use, is to use the walk time distributions for the 8 available dates, and use a universal walk speed distribution for gates with no information. To perform this estimation, we fitted the walk times individually for each gate in Pier C for the months of July, August, September and October.

Fitting walk times After collecting the walk times, we converted these times to walk speed depending on the shortest path distance to immigration C. The walk speed distribution for all gates during the month of July is illustrated in figure Figure 1.5. After plotting these speeds, it became obvious that some of the speeds were unrealistic. They correspond to measurements at gates situated near the immigration zone, and the error is due to the original location uncertainty in our data-set. To filter for these high speeds, we discriminate against unrealistic walk speeds. We used the widely accepted model of un-impended walk

speed reported by Fruin [16] and Young [17] to discriminate against walk speeds that were greater than two standard deviations from the mean. That model assumes un-impended walk speed follows a Gaussian distribution with a mean of 265 ft/min and a standard deviation of 50 ft/min. This average was first observed at the Port Authority of New-York by Fruin, and later confirmed by Young at San Francisco International Airport (SFO).

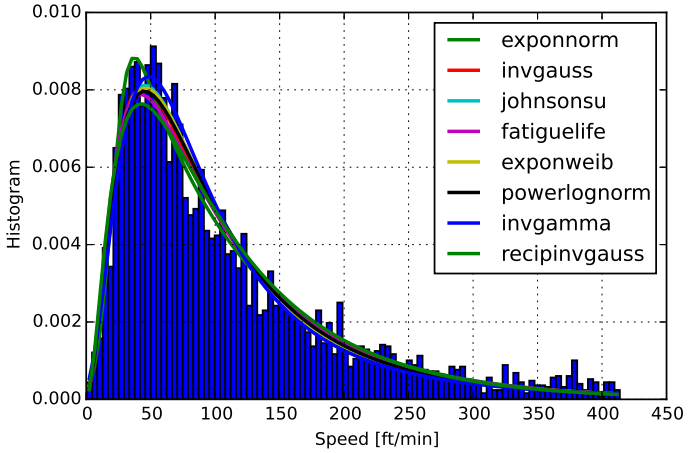


Figure 1.5: Parametric Fits of the Walk Speeds from Gates to Immigration for the Training Set (6,638 records)

Table 1.2: table of number of data points per gate for the walk times to immigration service C

Gates	50	51	53	54	55	56	57	58	59	60	61	63
number of elements	9	467	585	1145	2131	1112	30	973	1337	1602	2211	864

After filtering for reasonable walk speeds, we analyze the walk times by gate. We first look at a box plot of all walk times in Pier C and then at the walk time distribution by gate. We look at the walk times versus distance for the different gates in figure Figure 1.6 and the overall distribution of walk times for each gate Figure 1.7. The walk times collected

in this fashion represents a multitude of passengers. It represents individual passengers who go directly from their gates to the immigration zone un-impended, as well as group of passengers who tend to travel slower, and passenger who may stop by the restrooms along their journey to the immigration area. From the plot of walk times vs distance (figure Figure 1.6), the distance effect seems to be pronounced only for the lowest walk times for each gate. It implies that the distance effect is a major factor only for un-impended walk times. To verify the importance of walk speed, we would need to perform a linear regression on distance, type of traveler and whether the traveler is transiting through Sydney Kingsford Smith International Airport (SYD) or not.

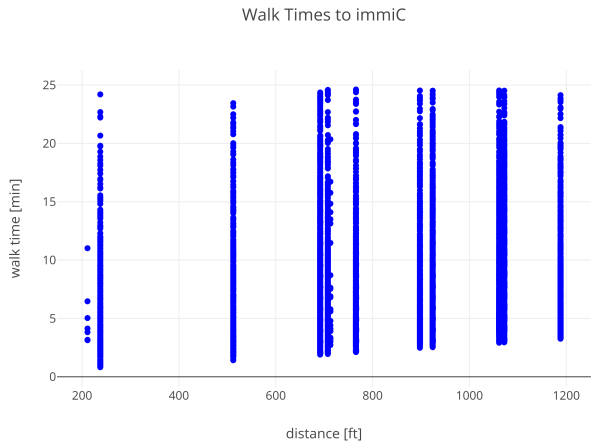


Figure 1.6: Walk times to immigration zone by distance for pier C

After obtaining walk times, we fitted them to all 80 distributions in Python Scipy to minimize the mean square error between the fit and the normalized frequency. As expected the lognormal distribution and the unbounded Johnson distribution figured among the best five fits for each gate. We evaluate the goodness of the fit by performing a Kolmogorov-Smirnov on each fit. The results are summarized in table Table 1.3. Although those were the best distributions for our data-set, the walk time distribution fits for all but 5 gates (56, 57, 58, 59 and 61) are rejected by the Kolmogorov-Smirnov test ($p<0.05$). This is likely due to the walk time distribution representing different types of passengers and hence being multimodal. We have decided to use the empirical CDF to generate the walk times in our

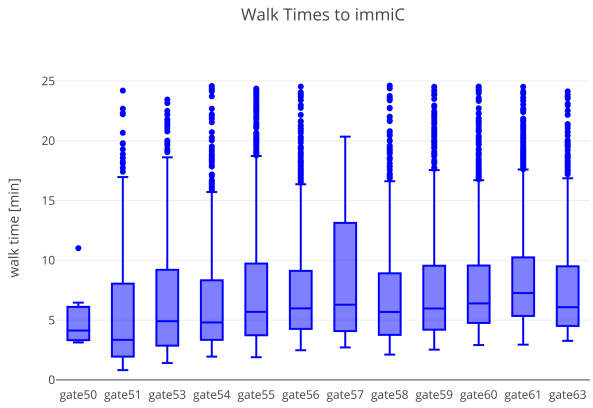


Figure 1.7: Box plots for the walk times for all gates in pier C

simulation for the gates in Pier-C. We tested the empirical distribution against a validation set formed of the months of November and December. It can be seen in table Table 1.3, that the p-values are quite high, indicating that the empirical distribution of walk times represents well the walk times from gates to immigration. To make the more general, a mixture model of Gaussian or lognormal distributions could be envisioned. For the gates with no data, we fit the walk speeds derived from the walk times for pier C in the next section.

Table 1.3: table of Kolmogorov-Smirnov Goodness-of-fit test p-values

Distribution	gate 51	gate 56	gate 57	gate 58	gate 59	gate 60	gate 61	gate 63
Lognormal	0.0457	0.2199	0.868	0.1113	0.0908	0.0134	0.0505	0.0365
JohnsonSU	0.0443	0.2138	0.8657	0.1115	0.0911	0.0134	0.0561	0.0358
Empirical(validation)	0.767	1	1	0.9875	1	0.9992	1	1

Fitting walk speeds For gates with no walk time information, we study the free flow walk speeds of passengers. Having a distribution of walk speed makes the arrival rate more general, and extensible to all the regions of the airports. One drawback of this approach, is that it does not account for stops made along a passenger path, or even all the possible paths of a passenger. This can be studied in a future model by extending the model. The model could be transformed into a more complex Markov process, where each location in

the airport is a node. Each node along the path would have a dwell time probability, and a probability of transition to the next node. The walk speeds are obtained from the available walk times by dividing shortest path distance by the walk times for each passenger. This resulted in an empirical walk speed distribution The parameters of the walk speed distribution were then estimated for all the gates at once.

The month of July was selected as a training set for the walk speed. Statistics reported in this section are only related to that month. Out of the 578,104 devices observed for the whole year, only 6,638 paths were conserved for the month of July. We used Python and the Fitter package[**fitterPy**], to find the best parametric distribution for the walk speed. Fitter works by ranking the maximum likelihood best fit of all the 80 distributions in Scipy, and returning the ones with the least mean square error. According to this metric, the best distributions for the walk speeds for all the gates are the Johnson distribution with unbounded support (Johnson's S_{U^-}), Wald, the inverse Gaussian, the power lognormal, the fatique life, the log normal, the exponential Weibull and the inverse Gamma distributions. The best fits for these 8 distributions are illustrated on Figure Figure 1.5. A t-test was performed to develop a confidence that the distribution mean represent the population mean without prior knowledge of the distribution variance. The 6 distributions provide similar levels of accuracy for the training set. Table Table 1.4 reports the p values of a t-test that was performed on the means of the parametric distributions.

Table 1.4: table of p-values for student t-test analysis of means of the training set

Distribution	Gate 8	Gate 24	Gate 53	Gate 54	Gate 58	Gate 59	Gate 60
johnsonsu	0.49511	0.94909	0.97936	0.97434	0.94443	0.98822	0.83714
wald	0.53310	0.89841	0.77131	0.91784	0.98284	0.94774	0.93039
invgauss	0.46680	0.45479	0.77131	0.98292	0.92882	0.97637	0.81265
powerlognorm	0.49511	0.94909	0.79007	0.97433	0.93963	0.98476	0.82915
fatiguelife	0.45459	0.95389	0.77660	0.98223	0.92708	0.97594	0.80778
exponnorm	0.45567	0.95420	0.77726	0.98248	0.92701	0.97577	0.80796

From the results, it can be seen that the scores are overly higher for gates 50 through 60. They represent a higher proportion of the data points used for the fit, which explains the better results. Because there are relatively few data points for the other gates, the fit may not be as accurate, and cause some underfitting.

The Johnson distribution was retained as the parametric distribution that represents our data. After fitting for the month of July, the fit is tested against the walk speeds between gates and immigration observed during the months of October, November and December Figure 1.8.

Validation of Arrival Speeds To verify that the parametric fit is representative of the other months, the mean walk speeds for the walk durations from every gate to immigration is compared to the mean of the distribution with another t-test. The p-values for this test are summarized in table Table 1.5. It shows that overall, the p-values for the gates 50-60 are lower, while the p-values for the other gates are higher. It indicates that there was a slight over-fit in favor of the gates between 50 and 60. The parameters of the Johnson's S_{U^-} (a=-2.2, b=0.834,loc=0.1605,scale=0.134) were deemed to provide acceptable results, and used for the simulation.

Table 1.5: table of p-values for student t-test analysis of means of the test set

Distribution	Gate 8	Gate 24	Gate 53	Gate 54	Gate 58	Gate 59	Gate 60
johnsonsu	0.39183	0.48369	0.85477	0.97936	0.96660	0.92203	0.84182
wald	0.44522	0.52350	0.82275	0.92185	0.96506	0.99916	0.93420
invgauss	0.35947	0.95534	0.85290	0.98838	0.95289	0.90451	0.81780
powerlognorm	0.39182	0.48369	0.85477	0.97936	0.96247	0.91655	0.83397
fatiguelife	0.34650	0.44243	0.84867	0.98784	0.95180	0.90210	0.81296
exponnorm	0.34761	0.44352	0.84919	0.98809	0.95169	0.90209	0.81313

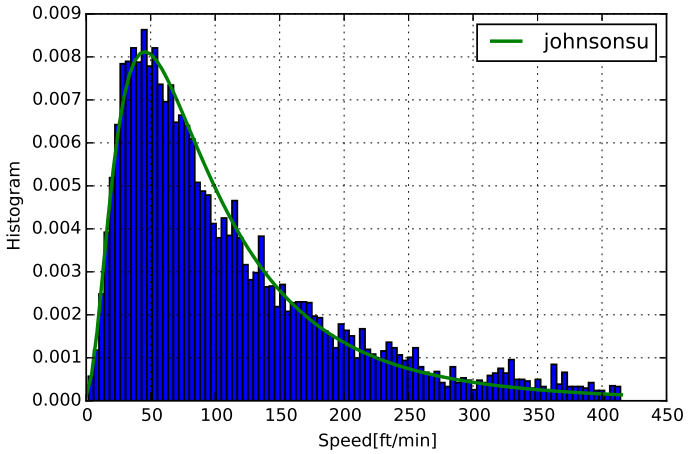


Figure 1.8: Test set(14,554 records) against Johnson's S_U^- fit

It can be seen that no distribution provides a high confidence, over 90th percentile, for all the gates. If we picked a 90 percentile f-value, we will fail to reject the hypothesis of different means for 3 gates in the dataset for both the Johnson's S_{U^-} and the Wald distributions. This can attributed to the variability of good data per gate. The Johnson's SU and the Wald distribution overall provides the best performance guarantee.

Estimation of throughput

Throughput is the number of passengers processed by a server or server system at a given time period. As mentioned above, it has been recorded by the Department of Immigration and Multicultural and Indigenous Affairs (DIMIA). It gives an idea of how busy the servers were on a particular day. A throughput does not give any information on the utilization of a server, the fraction of the time period for which a server is busy; nor does it gives information on the length of the queue. In our simulation, we are interested in the capacity of a server k to process passengers at a time interval j. This capability is the service rate of the server, or the mean number of customers that can be served at 100 % utilization by each individual server per unit time [**Shmula**]. It is denoted $\mu_k(t_j)$. Throughput cannot directly be used as an input to our simulation, but can be indirectly used to estimate the service rate

for the server system. For this purpose, we consider 100 time periods of 15 minutes with the longest average time spent in the immigration zone, and use them to estimate the initial parameters of the service rate distribution.

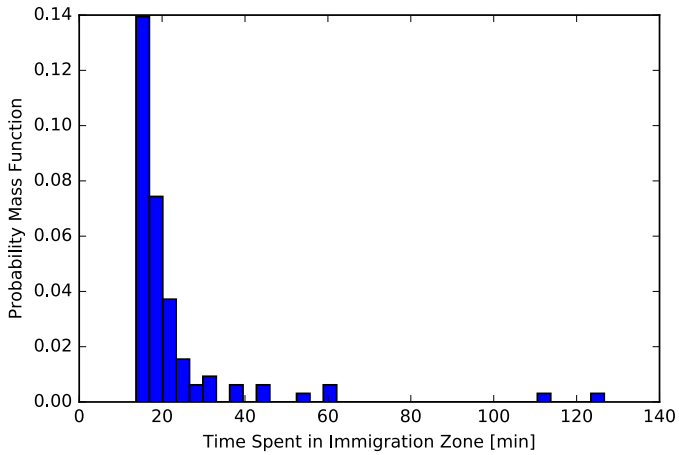


Figure 1.9: Empirical Distribution of time spent at immigration

This procedure also produced two extreme values for which the time spent at immigration exceeded an hour, as can be seen on Figure Figure 1.9. Those large values were removed from the analysis. For each time period, we computed the corresponding throughput at immigration, and the number of active desks. If we were simply to accept throughput as service rates, we would obtain the empirical distribution of Figure Figure 1.10. After filtering, the distribution of throughput for an individual desk in 15 minutes has a large support ($[3, 1210]$) and even greater variance 8778.536. It likely does not represent well the effectiveness of a single immigration worker working at a desk. To obtain more realistic service rates, we wish to improve the service rate distribution using available information on wait times, and arrival rates. This is a partially Bayesian approach, as the distribution selection is fully guided by the parametric fit of the throughput. This is in contrast to a Fully Bayesian Approach, for which the distribution selection and the parameter estimation are performed based on likelihoods, as is performed for Bayesian Model Average [**Chick2001**].

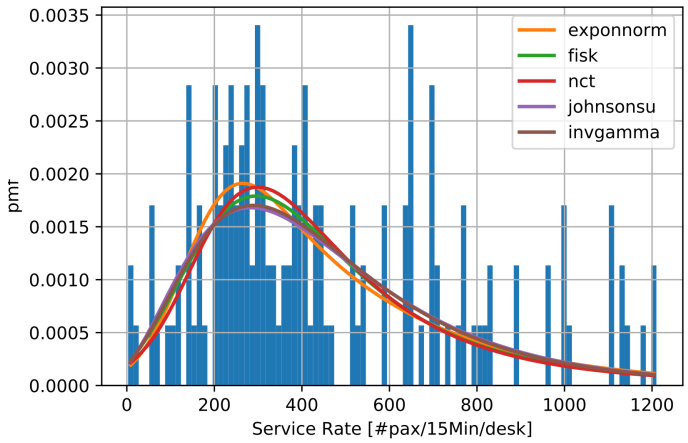


Figure 1.10: Initial Empirical Service rates per desk Per 15 minutes distribution from throughput

The empirical throughput were fitted using Python's Fitter library. This library fit the empirical data to all distributions in Python Scipy package, and rank them by sum of square errors. The Exponential Normal distribution, the fisk distribution, the non-central student t distribution and the Johnson distribution with unbounded support ($Johnson_{S_u}$) were found to perform the best. The three parameters $Gamma(\alpha_0, \beta_0, \zeta_0)$ distribution was retained to generate the service rate distribution required for the queueing process. This distribution has shape $\alpha_0 = 2.715983003$, inverse scale $\beta_0 = 183.43$, and location $zeta_0 = -37.5609$.

Table 1.6: Parameters for Service Rates per 15 minutes per desk

Distribution	Sumsquare error	Shape1	Shape2	Location	Scale
Exponnorm	0.000062	3.387588411	-	146.76293633	92.65
$Johnson_{S_u}$	0.000065	-7.45162592	2.1135730122	-164.83525407	33.002
Gamma	0.000066	2.715983003	-	-37.560950309	183.43

Service Rates

The service rate is the maximum number of passengers an individual server can process during a 15 minutes time interval. Realized throughput, on the other hand, is the actual number of passengers processed during the same period. It is a function of the service rate and the demand during that period. Because the queue spends most of its time starved when there is no or little demand, the throughput can vary considerably from the service rate in practice. Service rate, on the other hand, does not depend on the demand, but rather on the number of active desks. Throughput therefore underestimates the service rate. Yet, it is a typical surrogate for service rates in practice, where throughput during periods with high demand are taken as representatives of the service rates. In an effort to provide the most suitable input to our simulation, we have decided to use a revised model of service rates. This is motivated, in part, to the unreliability of the data-set available to us. The Wi-Fi information which should be used to estimate queue length lacks spatial accuracy in the immigration area, which makes it difficult to estimate which passengers are actually in a queue. Similarly, the immigration records only accounts for a fraction of passengers. For example on July 26th, one of the busiest day in the data-set, there was 183,718 passengers in the airport according to flights data. Of these 183,718 passengers, only 15,647 were recorded at immigration in our DIMIA database. This strongly suggest that some servers were not accounted for, and that the recording is incomplete.

1.5 Estimation Framework

The service rate distribution is a key factor in determining the capacity of the immigration services area. Because they are difficult to observe directly, service rates are often approximated from throughput information at capacity. In this section, we present a sequential sampling method to estimate the service rates from simulation and observed queue statistics.

1.5.1 Simulation of Passenger Traffic from Gates to Immigration

The goal of the simulation is to accurately emulate what happened inside the airport between the different international gates and their respective immigration zone. It is desired to obtain an account of the performance of the system by estimating average waiting times (W), average queue throughput (N_Q) and average time spent in the immigration zone or cycle time W in a 15 minutes time period. The number of passengers in the queue L_Q and in the immigration zone L are recorded of a time period. The DEDS previously described in subsubsection 1.3.3 is used to obtain granular queue statistics.

1.5.2 Updating Input Distribution

The typical output statistics of a discrete-event simulation are: the wait time ($W_Q(t_i)$), the average time spent in the queue during a time interval t_i, the cycle time ($W_c(t_i)$),the average time spent by passenger in the queue-server system during a time period t_i, queue length $L_Q(t)$, the system length ($L(t)$), the total number of passengers waiting in the system and the number of passengers being served $N(t)$ [15], [18]. Of all these variables, we use the cycle times as the control variable to update the service rates of the immigration queue. This choice is motivated by the robustness of wait times and cycle times to data inconsistency and unavailability. Indeed two passengers who stand in the same queue observe the same wait time on average. Only a few number of observed wait times are necessary to optimize the service rate selection.

1.5.3 Model-Based Update of Service Rates

By estimating the service rates in the context of the simulation, we can leverage the simulation model to train the samples of the service rates, creating a self-reinforcing feedback system between simulation and optimization, see figure Figure 1.11. The objective of this section is to present a method to update the marginal service rates distribution by minimizing the expected risk between observed and actual wait times. To optimize the stochas-

tic at each operation, and ensure convergence we use Model Reference Adaptive Search (MRAS). In the following section, we describe the MRAS algorithm, we define our parametric simulation, the objective function and present an optimization approach chosen to minimize the cost function.

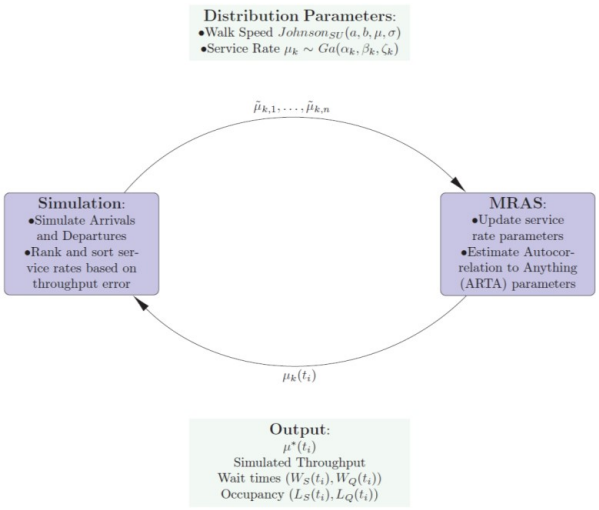


Figure 1.11: Diagram of the Model-Based Simulation

Procedure

For each new service at the immigration queue, we draw a specific service rate and convert it to a service time. All the service rates drawn during one run of the simulation form a sample path. After all the simulation runs results are obtained, we compare, sort and rank all the sample paths, retaining the best ρ-th quantile ($0 < \rho < 1$). Those samples are used to build the expected loss function to optimization over for the parameter estimation problem Equation 1.1. This process is illustrated in figure Figure 1.12. Once a desired confidence on the quality of our service rates is obtained, the update is stopped and a service rate distribution is returned.

$$\min_{\mu(t)} R(\mu(T)) = \sum_{t_i \in T} \mathbb{E}[\left\| W_c^{sim}(t_i, \mu(t_i)) - W_c^{act}(t_i, \mu(t_i)) \right\|^2] \tag{1.1}$$

This approach to updating the input distribution based on a simulation fits in the broader

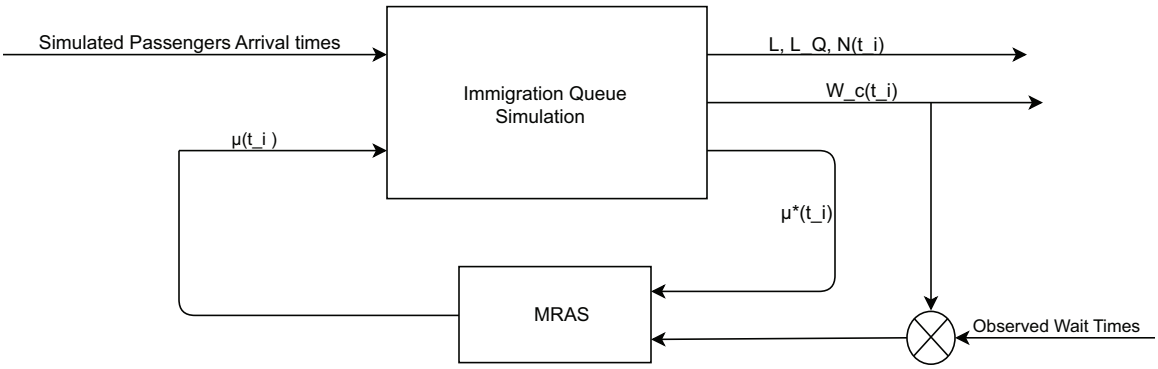


Figure 1.12: Diagram of the Sequential Update Methodology

category of model-based stochastic search methods. Model-based stochastic search methods [19] build a model of the world by continuously updating the sampling distribution until the sequence of samples converges to the globally optimal and potentially degenerate distribution. If the environment is non-stationary, new solutions are generated by updating the probabilistic model from previous iterations. The process is repeated until the distribution converges to the distribution representing the solutions to global optimization problems [20]. It is done in two phases: a sample generation phase that uses the current model to serve as input to the simulation and generate new samples, and an update phase during which the parameters of the probabilistic model are estimated from the best samples and updated to create a new family of distribution. The simulation model has been described in section **??**. The estimation process and update methods consist of finding the best parameters for the distribution based on the current samples as well as the best dependency structure for the best sample paths. This process will be described below as part of the update method.

We motivate the use of a model-based algorithm, by our desire to construct a good stochastic simulation of airport operations together with more reliable input distributions. Some examples of model-based algorithms are sequential Bayesian methods, Importance Sampling, Kalman Filters, the Cross-Entropy Method (CE) [**DeBoer2005**], Ant Colony Optimization (ACO) and Estimation of Distribution Algorithms (EDAs). Cross-Entropy method is an adaptive algorithm for rare-event simulation, that was later adapted to solve

combinatorial and multi-extremal optimization problems. The method successfully estimate the parameters that minimize the Kullback-Leibler (KL) divergence between the current best distribution family and the optimal distribution. Model Reference Adaptive Search (MRAS) [20] [21] is a generalization of the Cross-Entropy method with better convergence guarantees. The MRAS algorithm is described in the next section.

1.5.4 Model Reference Adaptive Search (MRAS)

In CE, we successively estimate the best parameters to minimize the KL divergence between the current parametric distribution and a target distribution. MRAS, on the other hand, operates in steps. At each iteration, the candidate solutions minimizes the KL divergence with respect to the current reference model. Because each iteration is based on the best candidate solutions from the previous iteration, the objective is monotonic. The MRAS method works with a family of distribution $\{f(\cdot, \theta_k), \theta_k \in \Theta\}$ on the solution space $\mathcal{H}$, where Θ is the parameter space and a sequence of reference distributions g_k. $f(\cdot, \theta_k)$ can be viewed as the sample distribution from which we generate candidate solutions at the kth iteration of the algorithm. As an alternative $f(\cdot, \theta_k)$ can be seen as the projection of g_k unto the current parameter space that minimizes the KL divergence between $g_k(\cdot)$ and $f(\cdot, \theta)$.

$$\mathcal{D}(g_k, f(\cdot, \theta)) := \mathbb{E}_{g_k}[ln\frac{g_k(x)}{f(x, \theta)}] \tag{1.2}$$

To construct the sequence of reference distributions, we start with an initial distribution $g_0(x) > 0 \forall x \in \mathcal{H}$. At each iteration $k \geq 1$, we bias the estimation of the parameters towards the best performing samples according to a performance function $H(X)$.

$$g_k(x) = \frac{H(x)g_{k-1}(x)}{\int_{\mathcal{H}H(x)g_{k-1}(x)\nu(dx)}} \tag{1.3}$$

In practice, the new parameter of the reference distribution is obtained by sample average over N i.i.d. samples $X_1, \ldots, X_N$ generated from $\{f(\cdot, \theta_k), \theta_k\}$.

$$\tilde{\theta}_{k+1} = argmax_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \frac{[S(H(X_i))]^k}{\tilde{f}(X_i, \tilde{\theta}_k)} I_{\{H(X_i) \geq \gamma_{k+1}\}} lnf(X_i, \theta) \tag{1.4}$$

For our model, X represents the parameters of the distribution and the performance function given by equation Equation 1.5 and a strictly increasing function $S(\dot{}) = exp(-rH(\dot{}))$

.

$$H(X) = -R(X) = -R(\mu(t)) = \exp\big(-\sum_{i=1}^{N} k[W_i - \tilde{W}(X_i)]\big) \tag{1.5}$$

$$\tilde{f}(X_i^k, \tilde{\theta}_k) = (1-\lambda) f(X_i^k, \tilde{\theta}_k) + \lambda f(X_i^k, \theta_0) \tag{1.6}$$

In both CE and MRAS, the threshold parameter γ_k depends on the desired percentile $0 \leq \rho \leq 1$ of samples to accept. In MRAS , the sample size N increases adaptively as a function of the growth parameter α. At the same time, the desired percentile ρ may decrease to avoid premature convergence to low quality solutions.

Algorithm

The algorithm consists of two phases:

1. Generate candidate solutions according to the current distribution $f(\cdot, \theta_k)$.
2. Compute a new parameter θ_{k+1} by using the samples generated in the previous step, focusing on the most promising region(s) of the decision space.

The steps of the MRAS algorithm are reviewed in Algorithm 1. Besides selecting the size of the initial sample N, the mixing coefficient λ and a strictly increasing function $S(\cdot)$, one of the major difficulty in using this method is the optimization performed to obtain the updated parameter θ_{k+1} in equation Equation 1.4.

The optimization problem is considered in the implementation section **??**. Similarly to the Maximum Likelihood Estimation (MLE) process, the optimization model can be ill-behave for a three-parameter distribution. We describe our optimization method in section subsubsection 1.5.4.

Algorithm 1 MRAS algorithm

1: **Initialization** :Specify $\rho_0 \in (0,1]$ an initial sample size $N_0 > 1$,$\epsilon \geq 0$, $\alpha > 1$, a mixing coefficient $\lambda \in (0,1]$, a strictly increasing function $S(\cdot) : \mathbb{R} \to \mathbb{R}^+$, and an initial p.d.f. $f(x,\theta_0) > 0, \forall x \in \mathcal{H}$. Set $\tilde{\theta_0} \leftarrow \theta_0$, $k \leftarrow 0$.

2: **repeat**

3: Step 1: Generate N_k i.i.d. samples $X_1^k, \cdots, X_{N_k}^k$ according to $\tilde{f}(\cdot, \tilde{\theta}_k) := (1-\lambda) f(\cdot, \tilde{\lambda_k}) + \lambda f(\cdot, \theta_0)$.

4: Step 2: Compute the sample $(1-\rho_k)$-quantile $\tilde{\gamma_{k+1}}(\rho_k, N_k) := H_{(\lceil (1-\rho_k)N_k \rceil)}$, where $\lceil a \rceil$ is the smallest integer greater than a, and $H_{(i)}$ is the ith order statistic of the sequence $\{H(X_i^k), i = 1, \cdots, N_k\}$.

5: Step 3:

6: **if** $k = 0$ or $\tilde{\gamma}_{k+1}(\rho_k, N_k) \geq \bar{\gamma_k} + \epsilon/2$ **then**

7: Set $\bar{\gamma}_{k+1} \leftarrow \tilde{\gamma}_{k+1}(\rho_k, N_k)$, $\rho_{k+1} \leftarrow \rho_k$, $N_{k+1} \leftarrow N_k$.

8: **else**

9: find the largest $\bar{\rho} \in (0, \rho_k)$ such that $\tilde{\gamma}_{k+1}(\bar{\rho}, N_k) \geq \bar{\gamma_k} + \epsilon/2$

10: **if** Such a $\bar{\rho}$ exists **then**

11: set $\bar{\gamma}_{k+1} \leftarrow \tilde{\gamma}_{k+1}(\bar{\rho}, N_k)$, $\rho_{k+1} \leftarrow \bar{\rho}$, $N_{k+1} \leftarrow N_k$.

12: **else**(if no such $\bar{\rho}$ exists)

13: Set $\bar{\gamma}_{k+1} \leftarrow \bar{\gamma_k}$, $\rho_{k+1} \leftarrow \rho_k$, $N_{k+1} \leftarrow \lceil \alpha N_k \rceil$.

14: **end if**

15: **end if**

16: Step 4:

17: Compute: $\tilde{\theta}_{k+1} = argmax_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \frac{[S(H(X_i))]^k}{\tilde{f}(X_i, \tilde{\theta}_k)} I_{\{H(X_i) \geq \bar{\gamma}_{k+1}\}} ln f(X_i, \theta)$

18: **until** a specified stopping rule is satisfied

The Generalized Gamma Distribution We select our service rates distribution family to be a generalized three parameters Gamma distribution of the form Equation 1.7. The

generalized Gamma distribution is characterized by a shape $\alpha > 0$, an inverse scale $\beta > 0$ and a location ζ parameter This distribution matches the distribution of the throughput of the immigration queue, which is used as the initial guess for the distribution in the MRAS process. .

$$f(x, \alpha, \beta, \zeta) = \frac{\beta^{\alpha}}{\Gamma(\alpha)}(x - \zeta)^{\alpha-1} e^{-\beta(x-\zeta)} \tag{1.7}$$

The exponential distribution of parameter λ can be represented as a Gamma distribution with $\alpha = 1$, $\beta = \lambda$ and $\zeta = 0$. Because of this property, the three parameters Gamma distribution is seen as a good generalization of the exponential.

Interior point optimization of the MRAS function

In solving for a good service rates for our queueing problem, we are looking for a process that possess both the desired marginal distribution obtained from MRAS, but also an adequate dependency structure given by AutoRegressive to Anything (ARTA). In this section, we describe the design approach and the implementation to obtain a correlated three parameters Gamma input process.

The optimization problem As part of the estimation problem, we need to optimize the MRAS objective function equation **??**, while guaranteeing that the resulting parameters result in a feasible distribution. Our optimization problem is:

$$\begin{aligned} min_{\alpha_k, \beta_k, \zeta_k} & -\frac{1}{N}\sum_{i=1}^{N} \frac{[S(H(X_i))]^k}{\tilde{f}(X_i, \tilde{\theta}_k)} I_{\{H(X_i) \geq \gamma_{k+1}^{-}\}} lnf(X_i, \theta) \\ & \alpha_k \geq 0 \\ & \beta_k \geq 0 \end{aligned}$$

The constrained optimization problem becomes ill-defined at the boundary of the feasible set when either $0 \leq \alpha < \epsilon$ or $0 \leq \beta < \epsilon$ for small $\epsilon > 0$. Traditional unconstrained optimization methods that rely only on the gradient of the objective function quickly lead to

ill-behaved behaviors. For this reason, a method was sought that could keep the candidate solution in the strict interior of the feasible set at all times, and return a solution close to the optimal solution. A general class of methods with these two properties are interior point methods [**NoceWrig06**], [**Bertsekas1999**] [22] such as log-barrier methods, Newton barrier methods, primal-dual methods, subgradient approximation methods for large problems and the ellipsoid method.

Let us define our problem generally as follows:

$$min f(x) \tag{1.8}$$

$$\text{subject to } c_i(x) = 0, \qquad i \in \mathcal{E} \tag{1.9}$$

$$c_i(x) \leq 0, \qquad i \in \mathcal{I} \tag{1.10}$$

Where $c_i(x)$ denotes a nonlinear constraint. $\mathcal{E}, \mathcal{I}$ are the set of equalities and inequalities respectively. Furthermore, we let $g(x) = \nabla f(x)$, the Jacobian $J(x) = \nabla c(x)$ and $J_{\mathcal{A}}$ the set of rows of the Jacobian corresponding to active constraints $c_i(x) = 0$. It is to be noted that our problem only contains inequality constraints. The sequence of solutions $\{x_k\}_k^\infty$ for barrier methods and Primal-dual methods follow a central path trajectory with the following properties:

1. $x^* = \lim_{k\to\infty} x_k$ satisfies the Karush-Kuhn Tucker conditions, see [**NoceWrig06**], e.g. there exists a nonempty set M_λ of Lagrange multipliers λ satisfying

$$M_\lambda = \{\lambda : g^* - {J^*}^T\lambda, \lambda > 0, \text{ and } c(x^*)\lambda = 0\}$$

2. The Mangasarian Fromovitz Constraint Qualification (MFCQ) hold at x^*, e.g. there exists p such that $J_{\mathcal{A}}(x^*)^T p > 0$

3. There exists $w > 0$ such that $p^T H(x^*, \lambda)p \geq w||p||^2$ for all $\lambda \in M_\lambda$ and all nonzeros p satisfying ${g^*}^T p = 0$ and $J_{\mathcal{A}}^* p \geq 0$, where $H(x^*, \lambda)$ is the Hessian of the Lagrangian.

The conditions ensure that the problem is always strictly convex, and that the solution is unique. Furthermore if we let $\mu > 0$ such that the problem achieves perturbed complementary at every iteration $c(x_\mu)\lambda_\mu = \mu$ we are guaranteed that the problem stays in the strict interior at all times, and converge to the optimum as $\mu \to 0$ [22].

Because of the nature of our problem, we found log-barrier and Newton barrier methods to quickly results in ill-conditioned gradients and Jacobian. The primal-dual algorithm we used, was found to be more stable than our log-barrier implementations.

Solving the optimization problem - finding the search direction The optimization algorithm iterate between two steps: the direction search, and the step size search.

Let $\theta = \{\alpha, \beta\}$ be our vector of decision or primal variables. The direction step's goal is to simultaneously solve for the primal and dual search directions $p_k = (\Delta\theta_k, \Delta\lambda_k)$ that will be used to update the primal and dual variables at the next iteration. To simplify the explanation we present our optimization problem Equation 1.8 in a more general equivalent form:

$$\begin{aligned} & min_{\theta_k} - f(\theta_k) \\ \text{subject to } & \alpha_k \geq 0 \\ & \beta_k \geq 0 \end{aligned}$$

We set up the problem as a general Newton root finding problem given a nonlinear equation $F(x)$, we try to solve for the step size $\Delta x_{k+1} = x_{k+1} - x_k$ such that $F(x_{k+1}) = 0$ in equation Equation 1.11.

$$\nabla F(x_k)\delta(x_{k+1}) = -F(x_k) \tag{1.11}$$

In the primal-dual method, we solve for the step sizes that satisfy the first order necessary

conditions or stationary conditions Equation 1.12 and the relaxed complementary slackness Equation 1.13 where $\mu > 0$ is a small positive number and $\mathbf{e}$ is a vector of ones.

$$g(\theta^*) = J(\theta^*)^T \lambda^* \tag{1.12}$$

$$c(x^*) \cdot \lambda = \mu \mathbf{e} \tag{1.13}$$

To find the search direction p_k, we solve the Newton search in equation Equation 1.14.

$$\begin{pmatrix} H(\theta_k, \lambda_k) & -J(\theta_k)^T \\ \Lambda J(\theta_k) & C(X) \end{pmatrix} \begin{pmatrix} \Delta\theta_k \\ \Delta\lambda_k \end{pmatrix} = - \begin{pmatrix} g(\theta_k) - J(\theta_k)^T \lambda \\ c(x)(\lambda - \Pi(\theta, \mu) \end{pmatrix} \tag{1.14}$$

Where, we have:

$$H(\theta, \lambda) = \nabla^2 g(\theta_k) - \nabla J(\theta_k)^T \lambda_k = -\nabla^2 f(\theta)$$

$$J(\theta) = \begin{pmatrix} I_2 & 0 \\ 0 & 0 \end{pmatrix}$$

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \end{pmatrix}$$

$$C(x) = \begin{pmatrix} c_1(x) & 0 \\ 0 & c_2(x) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\Pi(x, \mu) = \mu \cdot / c(x)$$

Solving the optimization problem - choosing the step size s_k We have:

$$\theta_{k+1} = \theta_k + s_k \Delta\theta_k \tag{1.15}$$

$$\lambda_{k+1} = \lambda_k + s_k \Delta\lambda_k s_k \qquad \in (0, 1]$$

There exist many update rules for finding the step sizes for primal-dual methods [**NoceWrig06**], [**Bertsekas1999**] and [22]. Choosing s_k such that θ_{k+1} is in the neighborhood (N_∞) of the central path, guarantees that all iterates stay in the interior of the Feasible set. $\mathcal{F}$.

$$N_\infty(\gamma) = \{(x, \lambda, s) \in \mathcal{F}^\circ | x_i s_i \geq \gamma\mu, i = 1, 2, \cdots, n\}$$

Another popular method is to use the Armijo rule or the Wolfe-Powell conditions commonly used in unconstrained optimization. We have decided to use Predictor-corrector step to find the next solution. We denote $(\Delta\theta_{aff}, \Delta\lambda_{aff}) = (\Delta\theta_k, \Delta\lambda_k)$ to be the solutions in equation Equation 1.14 to the linearized problem, and choose s_{aff} as the step size

$$s_{k,max} = min(1, max_{i:\Delta\theta_{aff,i}<0} - \frac{x_i}{\Delta x_i}) \tag{1.16}$$

In the correction step, we correct for any infeasible solutions generated in the affine search direction step due to nonlinearities.

$$\begin{pmatrix} H(\theta, \lambda) & -J(\theta)^t \\ \Lambda J(\theta) & C(\theta) \end{pmatrix} \begin{pmatrix} \Delta\theta_{corr} \\ \Delta\Lambda_{corr} \end{pmatrix} = \begin{pmatrix} 0 \\ -\Delta c(\theta)_{aff}\Delta\Lambda_{aff} + \mu e - \theta\lambda \end{pmatrix} \tag{1.17}$$

Termination criterion We end the iteration when strict complementary slackness is satisfied (e.g. $\lambda_k^T c(\theta_k) \simeq 0$) and the stationary conditions are satisfied (e.g. $||g(\theta_k) - J(\theta_k)^T\lambda_k|| \simeq 0||$). To speed up the optimization, we relax those conditions by stopping the algorithm as soon as no significant improvement in step size (e.g. $||\Delta x_k|| < tol$) is observed.

1.5.5 AutoRegressive to Anything (ARTA) model

In operations, the service rates between two-time periods are not completely independent. Our assumption that the service rates are i.i.d. variables is therefore misleading at best. To capture the influence of the time of the day on our data, there is a need to treat the

service rates as time series. The time-series process can be specified with a pre-specified marginal distribution obtained from MRAS and an auto-correlation through lag p as a measure of dependence. One can use Autoregressive (AR) models for stationary models and Autoregressive Moving Average (ARMA) or Autoregressive Intergrated Moving Average (ARIMA) for non-stationary models. These common models limit the users to Gaussian distributions. One can alternatively use a transformation oriented approach that transforms a base correlated process to generated an input process with the selected marginal distribution and desired correlation.

AutoRegressive to Anything (ARTA) is a transformation oriented approach to model arbitrary time series. With ARTA, we can transform independently identically distributed variables $X_1, \cdots, X_n$ from any well-formed marginal distribution into a time-series process $Y_1, \cdots, Y_n$ with an auto-correlation structure specified by lag p [23], [24]. The transformation is performed by estimating the underlying time correlation from our variables $\{X_t\}$, generating a standardized Autoregressive Process $\{Z_t\}$ of lag p and finally transforming that process into the target marginal distribution $\{Y_t\}$.

The ARTA process results in a stationary time series $\{Y_t\}$ with the two properties:

1. $Y_t \sim F_Y, t = 1, 2, \cdots$ where F_Y is an arbitrary distribution

2. $(\rho_1, \rho_2, \cdots, \rho_p) = \rho = (Corr[Y_t, Y_{t+1}], Corr[Y_t, Y_{t+2}], \cdots, Corr[Y_t, Y_{t+p}])$

In other words our final variables have the desired marginal distribution and product-moment correlation $-1 \leq \rho \leq 1$.

$$\rho_{t,t+h} = \frac{Cov(X_t, X_{t+h}}{\sigma_t \sigma_{t+h}} = \frac{\mathrm{E}[(X_i - \mu_i)(X_i - \mu_j)]}{\sigma_i \sigma_j} \tag{1.18}$$

ARTA process The ARTA process is defined by our target product-moment correlation and the corresponding auto-correlation structure or rank correlation $r = (r_1, r_2, \cdots, r_p)$ for the Autoregressive (AR)(p) process $\{Z_t\}$. The process can be summarized in the following steps.

1. Estimate the Pearson correlation vector $\rho = (\rho_1, \rho_2, \cdots, \rho_p)$ for our initial sample

2. Derive the base process, the corresponding stationary Gaussian AR(p) process

$$Z_t = \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + \cdots + \alpha_p Z_{t-p} + \epsilon_t$$

   where $\{\epsilon_t\}$ is a series of independent $N(0, \sigma^2)$ random variables and σ^2 is selected such that $\{Z_t\}$ is $N(0, 1)$.

3. Define the ARTA process

$$Y_t = F_Y^{-1}[\Phi(Z_t)], t = 1, 2, \cdots$$

   where F_Y is probability integral transform of the target distribution, and Φ is the standard normal cumulative distribution function (cdf).

The problem that is central in this estimation is the correlation matching problem $c(r_h) = \rho_X$ of finding the stationary Gaussian AR(p) process that matches the input correlation. In other words, we want to find the auto-correlation structure r that yields $Corr[Y_t, Y_{t+h}] = \rho_h$. Here c is a nonlinear transformation that depends on a double integral as seen in equations Equation 1.20 and Equation 1.19.

$$Corr[Y_t, Y_{t+h}] = c[r_Z(t, t+h)] = \frac{E[Y_t, Y_{t+h}] - E[Y_t]^2}{Var[Y_t]} \tag{1.19}$$

Since $E[Y_t]$, and $Var[Y_t]$ are fixed from the choice of the distribution. The problem reduces to finding the appropriate expectation $E[Y_t, Y_{t+h}]$, where

$$\begin{aligned} E[Y_t, Y_{t+h}] &= E\{F_Y^{-1}[\Phi(Z_t)]F_Y^{-1}[\Phi(Z_{t+h})]\} \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F_Y^{-1}[\Phi(z_t)]F_Y^{-1}[\Phi(z_{t+h})]\phi_{r_h}(z_t, z_{t+h})dz_t dz_{t+h} \end{aligned} \tag{1.20}$$

$\phi_{r_h}(z_t, z_{t+h})$ is the standard bivariate normal probability density function with correlation r_h. Because of the difficulty to analytically solve for the right ρ_h using Equation 1.20, we will turn to a numerical procedure to find r_h. There are a few properties derived by Cario and Nelson [23] that make the problem tractable. The function $c = \rho(r_h)$ is nondecreasing for $-1 \leq r \leq 1$, lies on the origin for $r = 0$, $|c(r_h)| \leq |r_h|$ and the function is continuous when $-1 \leq r \leq 1$ and $E[|Y_t, Y_{t+h}|^{1+\epsilon}]$. In spite of these properties, the efficiency of the numerical search method depends on the marginal distribution and the ρ_X values which could cause a non-smooth integrand in Equation 1.20.

generating random variables from the ARTA process

$$\Psi = \begin{bmatrix} 1 & r_1 & \cdots & r_{p-1} \\ r_1 & 1 & \cdots & r_{p-2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{p-1} & r_{p-2} & \cdots & 1 \end{bmatrix} \tag{1.21}$$

1. Obtain the time-series statistics: Average the desired statistic by time-period
2. Obtain the lag h and the correlation ρ of that time series using R
3. Generate p initial values $\{Z_{p-1}, Z_{p-2}, \cdots, Z_0\}$ from a multivariate normal distribution with mean $\mu = 0$ and covariance matrix Ψ see equation Equation 1.21
4. Set $Z_t \leftarrow \alpha_1 Z_{t-1} + \alpha_2 Z_{t-2} + \cdots + \alpha_p Z_{t-p} + \epsilon_t$ where $\epsilon_t \sim N(0, \sigma^2)$ and $\sigma^2 = 1 - \alpha_1 r_1 - \alpha_2 r_2 - \cdots \alpha_p r_p$
5. Return $Y_t \leftarrow F_Y^{-1}[\Phi(Z_t)]$
6. Set $t \leftarrow t + 1$ and go back to step item 3.

1.5.6 Accounting for time-dependence

Obtaining the time-series parameters

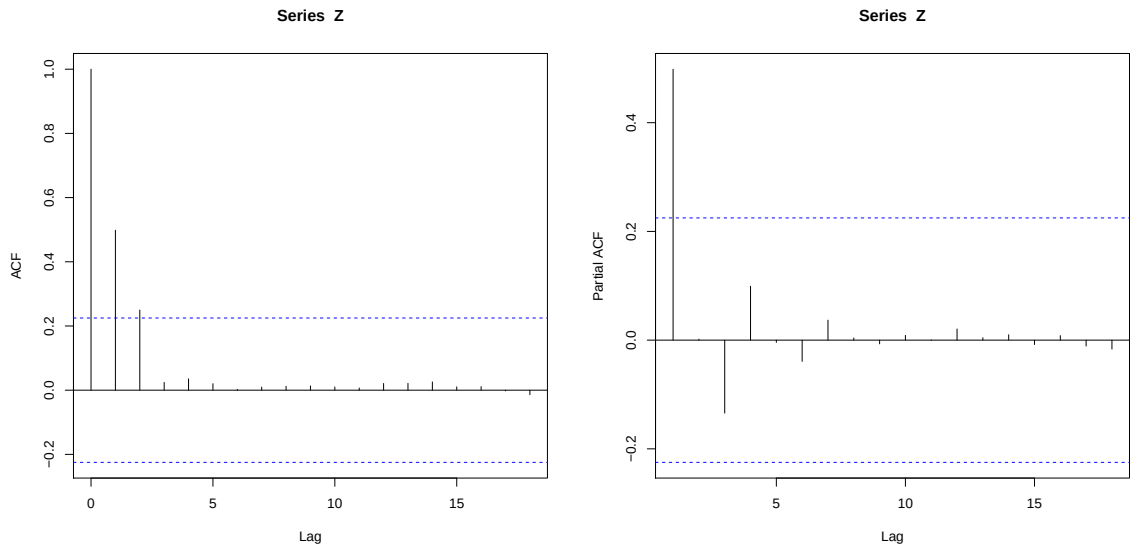
The MRAS update provides us with new parameters for the marginal generalized gamma distribution. This fit does not account for the time dependency of our time series. To increase the fidelity of our model, we study the temporal correlation of the service rates for one day. We combine the same samples used for the MRAS update, and average the service rates over a 15 minutes interval. This time-averaged service rates is the input to the ARTA process. The modeling of the ARTA process is performed in three steps:

1. Obtain the target product-moment correlation vector ρ and the lag h using R
2. For each product-moment correlation $\rho_i, i = 1, 2, \cdots, h$.
    - Guess a value of the cross correlation parameter r_h
    - Compute the corresponding cross expectation Equation 1.20
    - Iterate until the cross-correlation matching problem is solved.

To solve the cross-correlation matching problem, we currently use a simple version of the bisection algorithm. Once the cross-correlation matching problem is solved, we have a base transform F_Y that can be used to generate an ARTA process from i.i.d. standard normal variates.

Obtaining the base transform We first identify the lag or the number of terms in the AR process using auto-correlation and partial auto-correlation plots, figure Figure 1.13a and Figure 1.13b.

There is only one significant spike in the partial auto-correlation graph Figure 1.13b. This indicates that the higher order auto-correlations are effectively explained by the lag-1 auto-correlation. Using the Burg's method, we found the following auto-correlation for the



(a) auto-correlation Plot (b) Partial auto-correlation Plot

Figure 1.13: Correlograms for the best service rates

initial time series.

$$\rho = [0.64781, 0.072632, 0.014063, -0.34534, 0.1704]$$

and

$$\sigma^2 = 4.981e^{+}14$$

Two methods have been implemented to integrate equation Equation 1.20. The first method is a Monte Carlo integration with anti-variate sampling.

$$E^{MC}_{Y_t,Y_{t+h}} = \frac{1}{2N}(\sum_{i=1}^{N} F_Y^{-1}[\Phi(Z_{i,r1})]F_Y^{-1}[\Phi(Z_{i,r2})] + \sum_{i=1}^{N} F_Y^{-1}[\Phi(Z_{i,t1})]F_Y^{-1}[\Phi(Z_{i,t2})$$

$$Z_{i,r1} = Z_{i,1}, i = 1, \cdots, N$$

$$Z_{i,r2} = \rho_{trial} * Z_{i,1} + \sqrt{1 - \rho^2_{trial}} * Z_{i,2}, i = 1, \cdots, N$$

$$Z_{i,t1} = -Z_{i,r1}$$

$$Z_{i,t2} = -Z_{i,r2}$$

$$Z_{i,1}, Z_{i,2} \sim \mathcal{N}(0,1), i = 1, \cdots, N$$

Despite the use of $N = 80000$ data points for the Monte Carlo integration, the standard error obtained in the integration is still large at 8372, for a value of the integral value of $1.434e^06$. This is obviously an incorrect value.

The second method that was tried is an adaptive Newton-Cotes quadrature of degree 7. In spite of achieving an error within our tolerance (tol=0.05), the integral value (1434059.7362) is still far above the expected value for our marginal $E(X) = \zeta + \frac{\alpha}{\beta} = 693.743$ with variance $Var(X) = \frac{\alpha}{\beta^2} = 304772$. The resulting auto-correlation is not-a-number for the Newton-Cotes quadrature and 3.3 for the Monte Carlo method. Because the values of the auto-correlation are nonsensical, the current search method is incapable of solving the correlation matching problem.

1.6 Results of the Estimation

1.6.1 Simulating a day at the airport (Thursday July 26th 2012)

We simulate operations at SYD on Thursday July 26th 2012. On that day, the first flight reached the gate at 5:55 a.m. , and the last flight arrived at the gate at 11:28 p.m. The actual delays we are trying to match are shown on figure

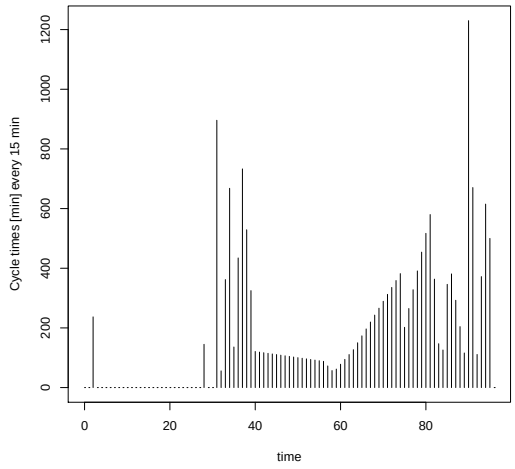


Figure 1.14: Cycle time by time of day on July 26th 2012

We let the algorithm run for 5 iterations, where each iteration consists initially of 30 runs. At each iteration, we simultaneously update all the parameters of the Gamma distribution to maximize our MRAS objective. The results of the fit for iteration 1, 2, 3, 4 and 5 are depicted on figures **??** respectively. The primal-dual method exhibit good monotonic convergence properties, but can still be unstable on some specific input data. This was the case for us after the fourth iteration, where the objective function became ill-behaved. A more detailed convergence analysis and stability analysis needs to be perform to make the algorithm more robust to non desirable data sets.

The evolution of the sample scores and the service rates are illustrated on figures Figure 1.20 and Figure 1.21. The initial trajectory of the sample clouds for the first three iterations is as expected. During the fourth iteration the top quantile stays at the same level, which forces the next iteration to use more samples and increase the uncertainty. This breaks the strict monotonicity of the score function, and also introduce an ill-behavior in our objective. The problem is currently being explored in more depth.

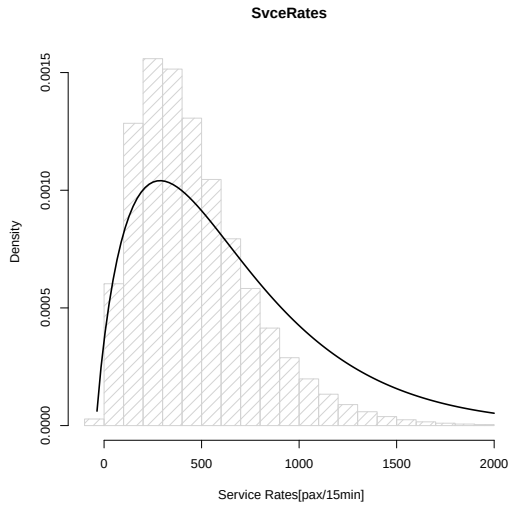


Figure 1.15: Fitted Service Rates after first iteration (July 26th 2012)

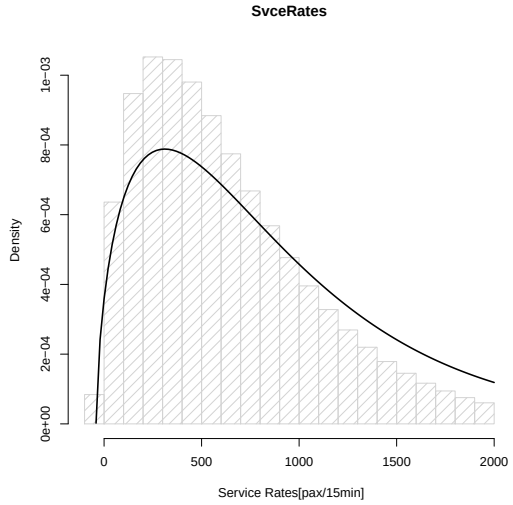


Figure 1.16: Fitted Service Rates after second iteration (July 26th 2012)

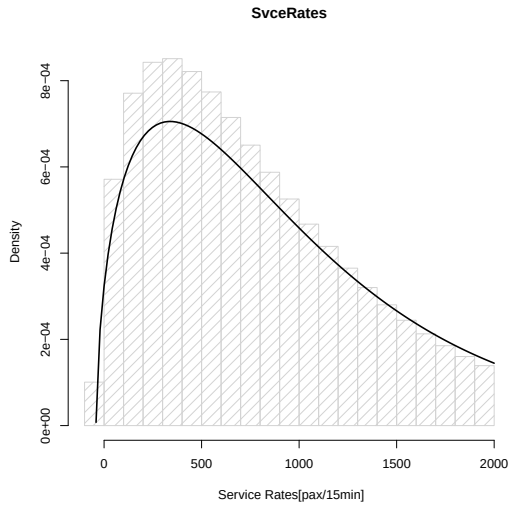


Figure 1.17: Fitted Service Rates after third iteration (July 26th 2012)

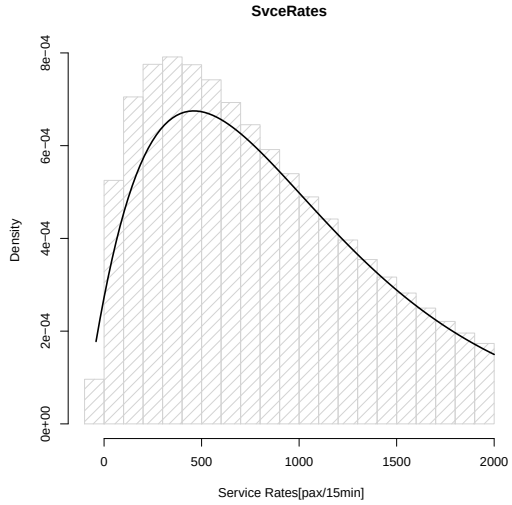


Figure 1.18: Fitted Service Rates after fourth iteration (July 26th 2012)

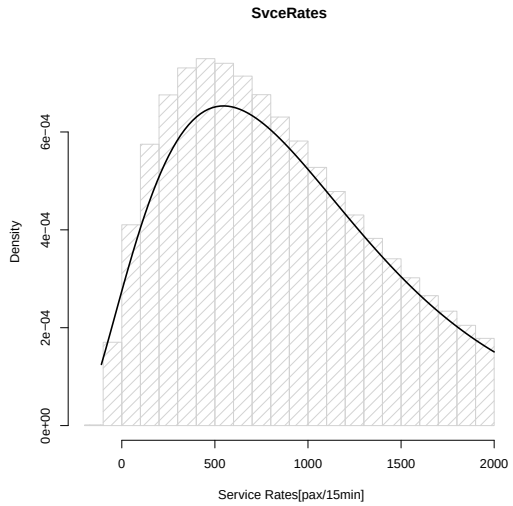


Figure 1.19: Fitted Service Rates after fifth iteration (July 26th 2012)

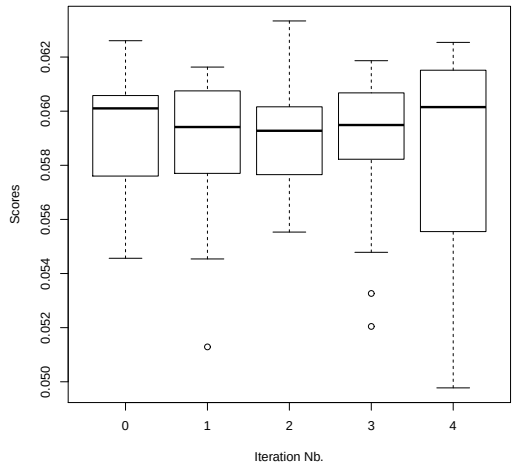


Figure 1.20: Evolution of score distribution with number of iterations (July 26th 2012)

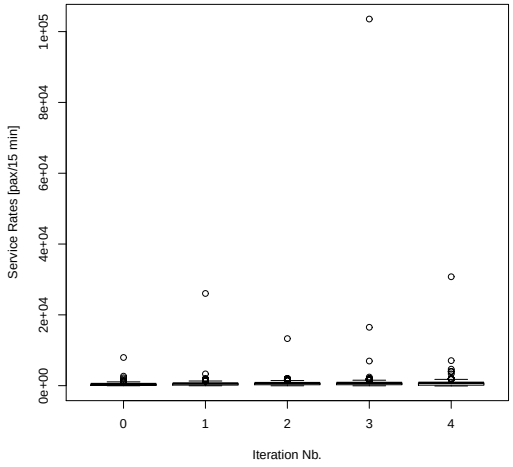


Figure 1.21: Evolution of service rate distributions with the number of iterations (July 26th 2012)

CHAPTER 2
DETECTION OF CHANGES TO THE DYNAMICS OF A M/M/1 QUEUE FROM A SINGLE PERFORMANCE MEASURE

Queuing systems are service systems where, when the server is busy or not available, an entity (e.g., a job, a customer, a packet or a phone call) may have to wait in a queue before being processed. Queuing systems evolve constantly. Such changes can be attributed to a change in the number of arrivals, a change in the number of servers, or a sudden breakdown of one of the servers. To maintain a consistent level of service, service providers need to adequately react to those sudden changes. The work done in the previous chapter focused on using simulation to associate heteregeneous sources of information that were both endogeneous and exogeneous to the queue. In many cases, it is possible to develop a well calibrated simulation of complex system because the lack of validating data. It is often not possible to collect transactional data such as start and end of service times. Similarly, exogeneous data such as the arrivals of new entities or even the number of entities to enter the system can not be ascertained. Endogeneous performance characteristics of the queue such as the number of passengers in the system or the queue or the time spent in the queue can often be approximated independently or jointly. In this chapter, we develop a strategy to detect abrupt changes in the dynamics of a M/M/1 queue based based on the partial observation of a single performance measure: number of passengers in queue, instantaneous time spent in queue and waiting time in queue. The next chapter looks at these three measures of performance jointly to make the best decision from the available information.

2.1 Heterogeneous Performance Characteristics

Any system of queues may be characterized by five parameters: the arrival rate, the service rate, the capacity of the queue, the number of parallel servers, and the queuing discipline.

When those five components are provided for a Markovian queue, the performance of a the system is fully determined. The service rate is the number of entities that a server can process at full capacity. Due to the inherent transient nature of real queuing systems, arrival and service rates are typically not directly observed in physical queues. Servers may not take the time to always record all the surges in demand. The throughput of a queuing system, defined as the number of entities that pass through the system for a given time interval, is easy to obtain. The service rate, which is defined as the maximum number of entities the servers is able to process, is only observable if all queues are operating at capacity and all queues are growing. To determine whether a change has occurred in the system, one must relies on other performance measures of the queues. Some common performance measures ares throughput, queue length, the waiting times, the systems times or the age of the customers in the process at a given time t. Those heterogeneous measures of performance make it possible to quickly detect any abrupt change in dynamics even when one of the measures fails to update.

The instantaneous queue length refers to the number of entities at a single point in time that are waiting in the queue before being served. For a physical system, obtaining the queue length requires dedicated employees or automated infrared/vision-based surveillance systems to monitor where the queue ends at any given point in time. Waiting time or delay is the time a customer wait in queue before being served. This data has historically been collected through surveys. The system time is the waiting time plus the service time. It is the total time spent in the queuing system before leaving the server. The age of the process [25] represents how long an item waiting for service has been in the system. Because it is an instantaneous property, it provides a more up-to-date representation of time spent in queue at any given with the one drawback of a high variability. In designing our change point detection strategy, we rely on the presence of observations based on waiting times, queue length or the age of a process to decide when the dynamics of a queue changed.

2.2 M/M/1 Queues

2.2.1 General Description of M/M/1 Queues

We study the simplest type of queuing system: a M/M/1 queue. Using Kendall's notation, a M/M/1 queuing system is a queuing system with the Markovian property for both the arrivals and departures for a single server with infinite capacity. A Markov chain is a random process in which the next state depends on the current state and not on the past. An arrival or a service rate is Markovian, if it has the memoryless properties. Arrivals or services are independent of each others and departures are independent of each other. In particular, they are independent from the time elapsed since the last service or arrival. In a M/M/1 queues inter-arrival times and service times follow an exponential distribution with rates λ and μ respectively where we define rates as number of customers by time unit. The fraction of arrival to service rate defines the load factor or utilization of the queue ρ. It is the average fraction of time a server is busy. A queue with $\lambda > \mu$ ($\rho > 1$) is over-saturated and fills up exponentially. In this work, we shall only be concerned with stable queues for which $\lambda < \mu$ $(rho < 1)$.

2.2.2 Steady-State Characteristics

Much work has been dedicated to the study of the steady state properties of M/M/1 queues. A M/M/1 queue can be modeled as a birth and death process, a Markov Chain where each state corresponds to the number of customers in the system [4]. It is then possible to look at the time-average properties of the system such as the average number of people in queue $L_Q(t)$ or the average number of people in the system $L_S(t)$. The expected time spent in the queue $W_Q(t)$ and in the system $W_S(t)$ are then derived from the queue length. The time spent in queue $W_Q(t)$ at time t is a property central to the analysis in the current situation and will be denoted as W_t for convenience. The expected waiting time and the variance of the waiting times are reproduced below without proofs in equations Equation 2.1 and

Equation 2.2 respectively. The queue length $L_Q(t)$ can be obtained from the waiting time using Little's Law: (i.e.: $L = \lambda W$), see equation Equation 2.3.

$$\mathbb{E}[W_t] = \frac{1}{\mu - \lambda} - \frac{1}{\mu} = \frac{\rho}{\mu - \lambda} \tag{2.1}$$

$$Var[W_t] = \frac{\rho(2 - \rho)}{(\mu - \lambda)^2} \tag{2.2}$$

$$\mathbb{E}[L_Q(t)] = \frac{\mathbb{E}[W_t]}{\lambda} = \frac{1/\mu}{\mu - \lambda} \tag{2.3}$$

These properties of a $M/M/1$ are important when analyzing a queue performance and developing a correlation between batches as shall be done further down.

2.3 Change-point Detection

Change point detection is the problem of finding abrupt changes in the distribution of a sequence of random variables $X_1, X_2, \ldots$. Two types of abrupt changes are possible: constant and transient changes. Transient changes are changes that are similar to impulses. They occur sporadically and leave the dynamics unchanged. Constant changes are sustained changes representing the new state of the system. The test is designed to capture constant changes that affects the underlying dynamics. The general framework of the Change-Point Detection (CPD) problem is a binary hypothesis test to decide whether a change in the underlying distribution of a process has happened or not, see equations Equation 2.4 and Equation 2.5.

$$H_0 = \mathbb{P}_m = \mathbb{P}_{m+1} = \ldots = \mathbb{P}_k = \ldots = \mathbb{P}_{m+n} \tag{2.4}$$

$$H_a = \mathbb{P}_m = \mathbb{P}_{m+1} = \ldots = \mathbb{P}_k \neq \mathbb{P}_{k+1} = \ldots = \mathbb{P}_{m+n} \tag{2.5}$$

The statistical process control community uses the Cumulative Sum (CuSum) test [26] in the form of control charts to detect changes. The CuSum algorithm [27], [28], [29] is a two-step algorithm consisting in a recursive Likelihood Ratio Test (LRT) to decide if a change has occurred and a maximum likelihood ratio test to ascertain when the change happened. It is used to detect a change in mean or variance in a time series consisting of independently normally distributed points. The test is widely accessible through standard control charts and many statistical packages. The method assumes that the parameters before and after the change are known. This is a major drawback. For most applications, the distibutions are unknown. Furthermore, the performance statistics of a queue are highly correlated and do not follow a Gaussian distribution. Several methods [30], [26], [31], [32] exist that achieve similar asymptotic optimality without these restrictions. One of them rely on a Generalized Likelihood Ratio Test (GLRT) [33], [34], [35] that shall be described below.

2.3.1 Likelihood Ratio Tests

In this section, we present the general Likelihood Ratio Test (LRT) framework. The LRT compares the ratio of the likelihood of the alternative hypothesis H_a to the null H_0 against a certain threshold η to make a decision on the change point location k. If the threshold is equal to one, the classifier corresponds to the Maximum Likelihood Estimator (MLE). When the threshold is a ratio of the priors, LRT is the optimal Bayes classifier [4] that can be found given the data. In other words the classifier obtained from LRT is the classifier that minimizes the risk of false detection under a given set of prior assumptions.

LRT techniques can be run sequentially when analyzing long sequences of data. This feature alongside with the optimality properties of LRT methods make them a method of choice for detection problems. Here we present the general LRT and multiple extensions to LRT that will lead us to a tractable and efficient algorithm for change point detection.

Let $\ell_m(k)$ and $\mathcal{L}_m(k)$ denote respectively the likelihood ratio and the log-likelihood

ratios where m is the start of the interval in consideration, $m+n$ is the end of the interval and k is the location of the change. Because the log function is a monotonically increasing function, the likelihood ratio can be replaced by the log likelihood ratio for the test. We have:

$$\ell_m(k) = \frac{P_{Y,H_a}}{P_{Y,H_a}} = \frac{\prod_{i=m}^{m+k} p(y_i,\theta_0)\prod_{i=m+k+1}^{n} p(y_i,\theta_1)}{\prod_{i=0}^{n} p(y_i,\theta_0)} \tag{2.6}$$

$$\mathcal{L}_m(k) = log(\ell_m(k)) \tag{2.7}$$

The likelihood ratio test to see if a change occurred is therefore:

$$\ell_m(k) \overset{p(y|H_a)\geq p(y|H_0)}{\geq} \eta \tag{2.8}$$

Or equivalently:

$$\mathcal{L}_m(k) \overset{p(y|H_a)\geq p(y|H_0)}{\geq} log(\eta) \tag{2.9}$$

We define $\hat{k} = argmax_k \mathcal{L}_m(k)$ to be the estimator of the location of the change point. If we replace k by $\hat{k}$ in equations Equation 2.8 and Equation 2.9, and the inequality is satisfied then we have found a change.

2.3.2 Generalized Likelihood Ratio Test

In many applications of interest, the type as well as the parameter of the distributions before and after the change are unknown. The first option to handle those unknown distributions is to assume the same family of distributions before and after the test. It is then possible to replace the parameters of the distributions before and after the change by their MLE equivalent resulting in the GLRT. This approach was used in [26] and [36] to define a parametric sequential test for the changes depending solely on the observations. The method is performed in the following steps:

- Compute the test statistic $G_{k,n} = -2log(\frac{p(x|H_0)}{p(x|H_a)})$ for a possible change point $k \leq n$

at each time step n

- Take the maximum of the test statistic $\max_{1\leq k\leq n} G_{k,n}$ and test if it exceeds the detection threshold h.

- Confirm the type of change by other hypothesis tests [26].

Both [26] and [36], use a finite sample correction factor for the estimator $G_{k,n}$. This correction is necessary to account for values of k close to the boundaries of the interval $[1, n]$. Although log-likelihood ratio tests converge asymptotically to the best estimators, they tend to have slow convergence properties. In the case of detection tests for a Gaussian distribution, the likelihood ratio statistic $G_{k,n}$ converge asymptotically to a χ^2 distribution. For small samples; however, any large deviation of a single observation can have a large impact on the statistic which can cause the required threshold to be artificially high. For χ^2 distributed test statistics, dividing the log likelihood ratio by its expectation under the null hypothesis gives faster convergence. This is precisely what is done with the Bartlett correction factor for Gaussian tests. The test statistic can generally be written as in equation Equation 2.11 where C is the correction factor.

$$G_{k,n} = -2\log\frac{\prod_{i=1}^{N} f(\hat{X}_{0,n})}{\prod_{i=1}^{k-1} f(\hat{X}_{0,k-1})\prod_{j=k}^{N} f(\hat{X}_{k,N})} \tag{2.10}$$

$$G_{k,n}^{C} = \frac{G_{k,n}}{C} \tag{2.11}$$

GLRT for normally distributed random variables

We will use the Generalized Likelihood Ratio Test of [36] for this test. For Gaussian random variables, the plug-in parameters are the mean $\hat{\mu}_n = \sum_{i=1}^{N} \frac{x_i}{N}$ and the variance $\hat{\sigma}^2 = \sum_{i=1}^{N} \frac{(x_i-\mu)^2}{N-1}$. Note that $\hat{\sigma}^2$ is unbiased and is therefore not the MLE variance for a

Gaussian distribution. The likelihood ratio statistic of equation Equation 2.8 becomes:

$$G_{k,n} = k \log \frac{S_{0,n}}{S_{0,k}} + (n-k) * \log \frac{S_{0,n}}{S_{k,n}} \tag{2.12}$$

In Ross' formulation $S_{r,s}$ is defined as follows:

$$S_{r,s} = \sum_{i=r+1}^{s} \frac{(x_i - \bar{x}_{r,s})^2}{s-r} \tag{2.13}$$

$$x_{r,s} = \sum_{i=r+1}^{s} \frac{\bar{x_i}}{s-r} \tag{2.14}$$

The correction factor for this case was obtained by Ross to speed up the convergence of the statistic to a χ^2 distributed random variable.

$$\begin{aligned} C = \mathbb{E}[G_{k,n}] = n * (log(2/n) + \psi((n-1)/2)) - k(\log(2/k) \\ + \psi((k-1)/2) - (n-k)(\log(2/(n-k)) + \psi((n-k-1)/2)) \end{aligned} \tag{2.15}$$

Here $\psi(z) = \frac{\Gamma'(z)}{\Gamma(z)}$ is the digamma function. The test statistic used in this work is:

$$G^C_{k,n} = \frac{G_{k,n}}{C} \tag{2.16}$$

Recursive expressions for the mean and variance in the GLRT formula

The search for $k^* = argmin_{k \in \{2, n-2\}} G^C_{k,n}$ requires us to compute the estimates of the variance before the change in $S_{0,k}$ and after the change in $S_{k,n}$. Instead of recomputing the variance to estimate $S_{0,k+1}$ for all the points $\{1, 2, \ldots, k, k+1\}$, it is possible to simply update $S_{0,k}$ with the new point x_k to obtain $S_{0,k+1}$ with a forward variance update. Similarly, it is possible to update $S_{k,n}$ to obtain $S_{k+1,n}$ with a backward variance update.

The forward variance update can be formulated as follows:

$$\bar{x}_{r,s+1} = \frac{(s-r)\bar{x}_{r,s} + x_{s+1}}{s-r+1} \tag{2.17}$$

$$S_{r,s+1} = \frac{s-r}{s-r+1}(S_{r,s} + \bar{x}^2_{r,s} - 2\bar{x}_{r,s+1}\bar{x}_{r,s} + \bar{x}^2_{r,s+1}) + \frac{(x_{s+1} - \bar{x}_{r,s+1})^2}{s-r+1} \tag{2.18}$$

The backward variance can be computed as follows:

$$\bar{x}_{r+1,s} = \frac{(s-r)\bar{x}_{r,s} - x_r}{s-r-1} \tag{2.19}$$

$$S_{r+1,s} = \frac{s-r}{s-r-1}S_{r,s} + \bar{x}^2_{r,s} - 2\bar{x}_{r+1,s}\bar{x}_{r,s} + \bar{x}^2_{r+1,s} - \frac{(x_{r+1} - \bar{x}_{r+1,s})^2}{s-r-1} \tag{2.20}$$

During simulated tests, it was observed that the forward variance estimator is more stable than the backward variance estimator for Gaussian distributed variables.

2.3.3 Change Point Models Library

We use the change point models (cpm) library by Gordon J. Ross [35] to detect change points using GLRT.

2.4 Detecting Change Point in Queues and Batch Means

Waiting times, age and queue lengths are highly correlated random processes in a M/M/1 queue, it is therefore not possible to assume their independence. It is not possible to directly apply LRT or GLRT to this change point detection problem because those two tests assume i.i.d normal variables. It is possible, on the other hand, to transform those statistics into new normaly distributed variables with low to no correlation. Once such transform, is the NOBM process, which reduces the correlation between inputs and increase the power of the detection test.

2.4.1 Non-Overlapping Batch Means

Batch Means methods allow faster convergence to identically independently distributed Normal samples. Under weakly-stationary assumptions [37], The sample mean of the batch means converges to the expected value of the distribution and the sample variance converges to the analytical variance. They are widely used in simulations as they require less samples in order for the central limit theorem to hold. NOBM is the simplest type of batch mean method. It partition the samples into batches of identical sizes.

2.4.2 The NOBM Method

Let $\{Y_i, i = 1, 2, \ldots, n\}$ be realizations from a weakly stationary process. The data is split into b batches of size m such that $b = n/m$. Then the observations $\{Y_i, i := 1, 2 \ldots, b\}$ constitute the ith overlapping batch with mean $Y_{i,m} = \sum_{k=1}^{m} \frac{Y_{(i-1)m+k}}{m}$. The sample variance for the NOBM converge to σ^2 with a rate proportional to the inverse of the square of the batch size for low autocorrelation [37] plus another term that depends on the autocorrelation of the batches Equation 2.22. Let the estimator be σ^2_{NBM} is defined as follows:

$$\sigma^2_{NBM} \equiv \frac{m}{b-1} \sum_{i=1} b(\bar{Y}_{i,m} - \bar{Y}_n)^2 \Rightarrow \frac{\sigma^2 \chi^2_{b-1}}{b-1} \tag{2.21}$$

$$\mathbb{E}[\sigma^2_{NBM}] = \sigma^2 + \frac{\gamma(b-1)}{bm} + O(1/m^2) \tag{2.22}$$

2.4.3 NOBM for Waiting Times

Waiting times for M/M/1 queues are problematic in change point detection for several reasons.

1. There is a high autocorrelation between consecutive waiting times. Even at steady-state, the autocorrelation function of the waiting times decays very slowly.

2. During simulation, the initialization bias has a high magnitude which decays slowly.

3. In steady-state operations, the marginal distribution of the waiting times has an exponential tail which breaks the normality assumption. There is always the chance of a rare event that breaks the symmetry assumption of the normal distribution.

4. The variance of the waiting times grows quickly as the service rate gets closer to the arrival rate as seen in equation Equation 2.2 for the variance of the waiting times in a M/M/1 system.

NOBMs can be used in the context of waiting times to create normaly distributed samples with low correlations for change point detection. In order to make the transformation useful, a small batch size that leads to low auto-correlation is desired. Large batch sizes cause large delays in detection and in extreme cases dilute the signal making it hard to detect change points as shall be seen in the simulated experiments below. First, we look at the auto-correlations between waiting-times and then at the auto-correlations between batch mean waiting times to

2.4.4 Auto-correlations of Waiting Times

Let $\{Y_i\}_{i=1,\ldots,n}$ be an arbitrary sequence of observations. For a batch $Y_{1,m} = \frac{1}{m}\sum_{i=1}^{m} Y_i$ of size m, we define the lag-1 auto-correlation as in equation Equation 2.23.

$$r_{BM} = \frac{Cov(Y_{1,m}, Y_{2,m})}{Var(Y_{1,m})} \tag{2.23}$$

Let $R_j = Cov(Y_1, Y_{1+j})$ and $r_j = Corr(Y_1, Y_{1+j}), R_j/R_0$. By Lemma 2 of [38], the variance of a NOBM can be written as in equation Equation 2.24.

$$Var(Y_{1,m}) = \frac{R_0}{m}\left[1 + 2\sum_{i=1}^{m-1}(1 - \frac{i}{m})r_i\right] \tag{2.24}$$

The covariance between batch means was derived by Steiger and Wilson [39] as in equation Equation 2.25

$$Cov(Y_{1,m}, Y_{2,m}) = \frac{R_0}{m} \sum_{q=-m+1}^{m+1} \left(1 - \frac{|q|}{m}\right) r_{m+q} \tag{2.25}$$

Using both equations Equation 2.24 and Equation 2.25, the lag-1 auto-correlation between batch means can be derived as:

$$r_{BM,1} = \frac{\sum_{q=-m+1}^{m-1} \left(1 - \frac{|q|}{m}\right) r_{m+q}}{1 + 2\sum_{j=1}^{m-1}(1 - \frac{j}{m})r_j} \tag{2.26}$$

To obtain the lag-1 auto-correlation between batches, only the variance of the individual samples ($Var(Y_{1,m})$) and their auto-correlations (r_j) are needed.

Auto-correlation between Waiting Times

Since NOBMs are unbiased estimators, the NOBM expected waiting time is the same as in equation Equation 2.1. The correlation between waiting times is defined as Equation 2.27 from the work of Daley [40].

$$r_j = Corr(W_1, W_{j+1}) = c \int_0^a t^j f(t) dt \tag{2.27}$$

$$c = \frac{(1-\rho)^3(1+\rho)}{2\pi\rho^3(2-\rho)} \tag{2.28}$$

$$f(t) = \frac{t^{3/2}(a-t)^{1/2}}{(1-t)^3} \tag{2.29}$$

$$a = \frac{4\rho}{(1+\rho)^2} < 1 \tag{2.30}$$

This results in the analytical batch mean waiting times auto-correlation in equation Equation 2.31.

$$r_{BM}(m, \rho) = \frac{c\int_0^a \frac{t}{m}(\frac{1-t^m}{1-t})^2 f(t)dt}{1 + 2c\int_0^a \frac{t(t^m - mt + m - 1)}{m(1-t)^2} f(t)dt} \tag{2.31}$$

The variance between batch mean waiting times can be expressed as:

$$Var(W_{1,m}) = \frac{\rho(2-\rho)}{m(\mu-\lambda)^2}\left[1 + 2\sum_{i=1}^{m-1}(1-\frac{i}{m})r_i\right] \tag{2.32}$$

The results on the batch mean variances were used in computing the auto-correlation between batches empirically in section subsection 2.4.6. The auto-correlations between batches were used as an additional check during the validation of the simulation process. As can be seen in table Table 2.1, relatively low batch sizes are required to obtain low correlation between samples.

Table 2.1: Minimum batch sizes required to satisfy $r(m,\rho) \leq \epsilon$

ϵ	$\rho = 0.25$	$\rho = 0.75$	$\rho = 0.9$
0.50	1	6	29
0.1	9	33	182
0.05	17	61	337
0.01	76	282	1561

2.4.5 Normality of NOBM Waiting Times

The NOBM also create identically distributed normal samples. To test this assumption, it suffices to compare the moment generating function (m.g.f.) of a standardized sum of waiting time samples against them.g.f. of the standard normal, equation Equation 2.33.

Let $X \equiv N(0,1)$,

$$E[e^{tX}] = e^{t^2/2} \tag{2.33}$$

The m.g.f. of steady-state waiting time is:

$$E[e^{tW_1}] = 1 - \rho + \rho\left(\frac{\gamma}{\gamma - t}\right)$$

where $\gamma = \mu - \lambda = \lambda(1-\rho)/\rho$ Let the mean of m consecutive waiting times be

$$\bar{W}_m = \frac{1}{m}\sum_{j=1}^{m} W_j$$

The standardized mean batch waiting time is:

$$X_m = \frac{\bar{W}_m - E[\bar{W}_m]}{\sqrt{Var(\bar{W}_m)}} \tag{2.34}$$

We then have that

$$logE[e^{tX_m}] = logE\left[exp\{t_m \sum_{j=1}^{m} W_j\}\right] - t_m E[W_1] \tag{2.35}$$

where

$$t_m = \frac{t}{m\sqrt{Var(\bar{W}_m)}} \tag{2.36}$$

$$t_m < \frac{\mu - \lambda}{m} \tag{2.37}$$

$$E[W_1] = \rho/(\mu - \lambda) \tag{2.38}$$

$$E[e^{t\sum_{j=1}^{m} W_j}] = (1-\rho)\sum_{j=1}^{m} c_{m,j}(t) + \rho\left(c_{m,1}(t)\left[\frac{\gamma}{\gamma - mt}\right] + sum_{j=2}^{m} c_{m,j}(t)\left[\frac{\gamma}{\gamma - (m-j+1)t + \lambda}\right]\right) \tag{2.39}$$

The derivation of equations Equation 2.35 and Equation 2.39 are in the appendix. It was found analytically that $logE[e^{tX_m}] \to t^2/2$ by computing equation Equation 2.35 for different values of m. This was confirmed experimentally by drawing multiple consecutive waiting times from simulation for different values of ρ in the next section.

2.4.6 Validating the i.i.d. NOBM assumptions for waiting times empirically

As mentioned above, the generalized likelihood ratio test (GLRT) is designed with the assumption that the samples are independently and identically Gaussian distributed with the parameters of the distribution being not known a priori. Waiting times are highly correlated with a correlation that increases as the traffic utilization ratio gets bigger. In order to use GLRT, the sampled waiting times need to be first transformed into i.i.d. normal variables. The transformation is obtained using non-overlapping batch means of various batch sizes. To evaluate the accuracy of the i.i.d. normal assumption after the transformation, the sampled NOBM waiting times are tested for normality and independence. To test the normality assumption, the moment generating function of the standardized NOBMs are comparared to the theoretical normal m.g.f. for various batch sizes. A Shapiro-Wilk test was also performed on the samples. In addition, to verify that the samples are independent, the auto-correlation between batches was computed for various batch sizes m and traffic intensity ratios ρ.

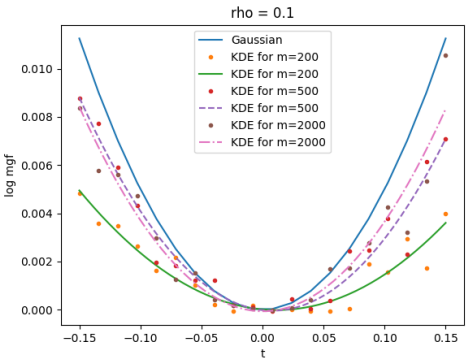
Normality Empirical Validation

To evaluate the normality of the batch means, a Shapiro-Wilk test is performed 100 times for each configuration at the 0.05 level. A positive outcome occurs when the Shapiro-Wilk test fails to reject the normality assumption. The proportion of positive outcomes then yields a normality ratio. A higher ratio gives a stronger confidence that the normality assumption is true. This fraction is reported in table Table 2.2 for the three different observations. The relationship between batch size, ρ and the normality of the batch means is nonlinear. It is difficult to generalize from this hypothesis test alone. Yet, a few conclusions can be made from the results of the test. The time spent in queue or age of the process naturally provides higher confidence in the normality assumption compared to either the waiting times or the queue lengths. The batch mean age is an average of average due to how the times spent in queue are recorded. The large number of samples that are be-

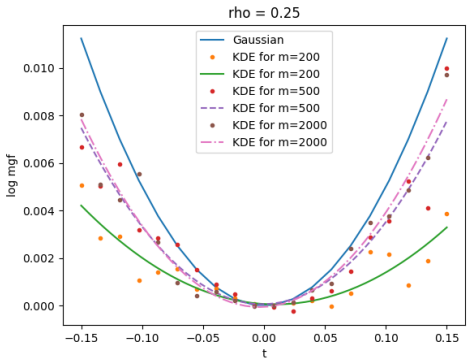
hind the batch observations for the time spent in queue are making the draws more normal thanks to the central limit theorem. In agreement with the previous observation, with higher traffic utilization there are more occupants in the queue leading to a more adequate normal assumption. On the other hand, since waiting times and queue lengths are not averages, the NOBMs appear less normal. Furthermore, the relationship between traffic utilization and normality is reversed for the latter two. For the same batch size, a higher utilization rate implies less normal samples when looking at queue lengths and waiting times. This could be attributed in part to the higher correlation at large utilization rates. To confirm those results, Q-Q plots were created for the waiting times, the queue length and the time spent in queue. The plots were added to the appendix. By visual inspection, it was found that the normal assumption became approximately valid for batches greater than 100 when looking at the age of a process. It takes batches greater than 2,000 to have a visually normal Q-Q plots for the queue lengths and the waiting-times.

In addition to the q-q plots, the moment generating function (m.g.f.) is used to validate the normality assumption following the derivations of section subsection 2.4.5. After extracting the waiting times, the log m.g.f. is computed from the samples and compared against the log m.g.f. of the standard normal in figures Figure 2.1a, Figure 2.1b, Figure 2.2a and Figure 2.2b.

The figures illustrate the change in the log normal with changes in t as in equation Equation 2.33. At low values of traffic utilization, the samples are close to normal. Figures Figure 2.1a and Figure 2.1b have many points right on the parabola $t^2/2$. The samples get closer to normal with bigger batch sizes and higher traffic utilization rates. Another observation is that the convergence to normality is slower for larger traffic utilization as can be seen in figures Figure 2.2a and Figure 2.2b where the m.g.f. are indistinguishable for batches of size 500 and 2000.

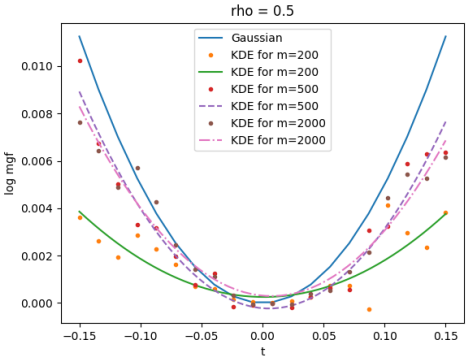


(a) $\rho = 0.1$

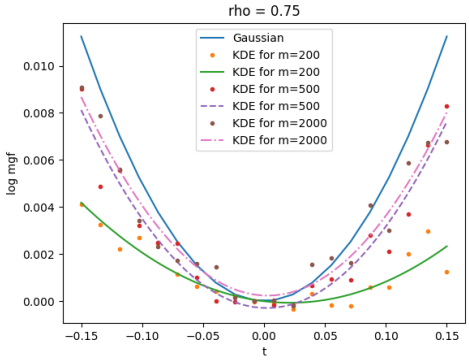


(b) $\rho = 0.25$

Figure 2.1: Waiting times Log m.g.f. at $\rho = 0.10$ and $\rho = 0.25$



(a) $\rho = 0.5$



(b) $\rho = 0.75$

Figure 2.2: Waiting times Log m.g.f. at $\rho = 0.5$ and $\rho = 0.75$

Independence Empirical Validation

Instead of running a test for independence, the auto-correlation between batches is computed. The batch means are assumed independent if there is low correlation between the batches and no evidence that the batches are normal. Let $\{Y_i\}_{i=1,\ldots,n}$ be an arbitrary sequence of observations. The auto-correlation of a the discrete NOBMs can be estimated as follows:

$$\hat{r}_{BM,t} = \frac{1}{(n-t)Var(Y_{1,n})} \sum_{j=1}^{n-t} (Y_{j,m} - \mathbb{E}[Y])(Y_{j+t,m} - \mathbb{E}[Y]) \tag{2.40}$$

The expected value of waiting times and queue lengths are well-known. The variance of the batch sizes, equation Equation 2.24, requires the knowledge of the wait times and queue length variances as well as their auto-correlation. The lag-1 auto-correlation between batch mean waiting times is presented above in equation Equation 2.31. The lag-1 auto-correlation between batch mean queue lengths can be obtained by plugging the batch mean variance Equation 2.42 into equation Equation 2.26 along with the expected queue length statistic Equation 2.3. The age-of a process does not have a closed-form solution for the auto-correlation. The simulated times spent in queue were not tested for independence or correlation. From the table Table 2.2, one can sees that waiting times are much more correlated than queue lengths especially at high traffic intensities. Bigger batch sizes considerably lower the correlation. Even for a batch of size 1,000, the auto-correlation between batches were still above 0.1 for a near congested queue ($\rho = 0.9$). Table Table 2.2 shows that we can have a high confidence in the i.i.d. assumption by picking a batch size above 200 and a ρ not too close to 1. During the change point detection experiments, the power of the tests will become clear.

m	ρ	Shapiro-Wilk			Auto-correlation	
		Wait-Times	Age	Queue Length	Wait-Times	Queue Length
100	0.25	0	0	0	0.0031	0.0
	0.50	0	0	0	0.0259	0.0033
	0.75	0	0.08	0	0.1561	0.073
	0.9	0	0.09	0	0.7194	0.3314
150	0.25	0.01	0.03	0	0.00	0.0
	0.50	0	0.03	0	0.0266	0.0011
	0.75	0	0.28	0	0.0995	0.0386
	0.9	0	0.2	0	0.6251	0.276
200	0.25	0.15	0.29	0.03	0.0059	0.0
	0.50	0	0.26	0	0.0105	0.0013
	0.75	0	0.42	0	0.0653	0.0227
	0.9	0	0.4	0	0.5494	0.2315
500	0.25	0.7	0.77	0.47	0.016	0.0
	0.50	0.22	0.78	0.15	0.0055	0.0009
	0.75	0	0.78	0	0.0149	0.0043
	0.9	0	0.85	0	0.2872	0.0952
1000	0.25	0.88	0.91	0.82	0.0113	0.0002
	0.5	0.6	0.81	0.57	0.0	0.0005
	0.75	0.1	0.9	0.05	0.040	0.0002
	0.9	0	0.95	0	0.1549	0.0318

Table 2.2: Looking at the fraction of non-rejected normality tests for 100 runs

2.5 NOBMs of Queue Length

2.5.1 Auto-correlation between Queue Lengths

Similarly to waiting times, it is possible to compute the auto-correlation between batch mean queue lengths. We use the approximation by Morse [41] for the auto-correlation between queue lengths.

$$r_j = Corr(L_{Q,1}, L_{Q,1+i}) \simeq \lambda\mu + exp\left[-(\mu-\lambda)^2\frac{i}{\lambda}\right] \tag{2.41}$$

The variance of a queue length batch mean can then be defined as:

$$Var(L_{Q,1,m}) = \frac{\lambda\mu}{(\lambda-\mu)^2 m}\left[1 + 2\sum_{i=1}^{m-1}(1-\frac{i}{m})r_i\right] \tag{2.42}$$

The latter result was used to compute the auto-correlations between batch mean queue lengths.

2.6 Simulation environment for the Change Point Detection Test

The change point detection is performed on observations made on waiting times, time spent in queue (also called age of the process) and length of the queue for a M/M/1 system. All of these statistics can be fully or partially observed in real queues. The process of developing and testing a change point method for these observations requires the development of a simulation of a server queue system. In this particular instance, a single server was method. The simulation also requires a method to generate the desired statistics from the simulation, a method to bin the observations and lastly a routine to run the change point detection on the batches. The diagram in figure Figure 2.3 illustrates the design selected for the environment in which the different change point detection methods are evaluated.

The simulation starts by setting a traffic intensity ρ. With minor loss of generality, the service rate is taken to be constant $\mu = 1$. The traffic intensity $\rho = \lambda/\mu$ is the arrival

rate λ. Together the arrival rate and service rate completely define the performance of the queue. The three direct outputs of the M/M/1 queue are the arrival times $A(t)$, start of service times $S(t)$ and departure times $D(t)$. From these three time series, the waiting times ($W_q(t)$), time spent in queue ($A(t)$) and queue length ($Q(t)$) can be derived. The latter three time series are independently fed to a batching procedure to obtain non-overlapping batches of size m. The Generalized Likelihood Ratio is then deployed on the i.i.d. normal batch means to obtain the estimated detection time indices K_t^W, K_t^Q and K_t^A for the waiting times, queue length and time in queue respectively. By comparing these estimates against the true change time $K*$ it is possible to generate the detection test metrics of interest detailed in section subsection 2.6.3

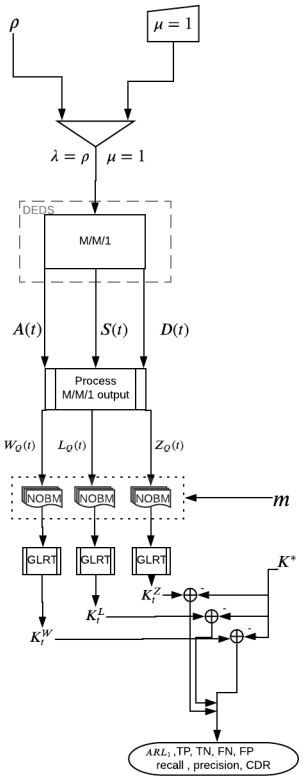


Figure 2.3: Simulation and Detection Process

2.6.1 M/M/1 simulation

A DEDS is used to emulate the dynamics of a M/M/1 queue. A DEDS is an event-driven simulation, where the simulation time moves forwards after each event. Three types of events exist: an arrival, a service an a departure. An arrival occurs when a customer arrives at the queue or at the server if the queue is empty. A service occurs when the customer leaves the queue and reaches the server. A departure represents the end of service and the immediate departure from the queuing system. During the simulation, the program collects statistics on throughput, queue length, waiting times and the age of the process upon a new arrival.

The simulation starts with the queue in a steady state for the chosen utilization factor ρ. Then each new inter-arrival time is drawn independently from an exponential distribution with rate λ. Each service time is drawn independently from an exponential distribution with rate μ. The simulation ends when the desired number of entities has been processed or when a desired stop time has been reached.

It is possible to simulate more directly the waiting times by directly drawing from a waiting time distribution or by simulating the ladder point process using Lindley equation [42]. Although these processes are faster and less data intensive, they are not amenable to the present change-point detection experiments. They do not offer the granularity to track all desired statistics or to change the distribution after a change point while keeping the implicit correlation between observations.

2.6.2 Simulating a change point in a M/M/1 queue

For a given observation, a good detection method can accurately sense a change and not signal a change-point when none is present no matter the number of observations that have been processed. In other words, although the observations do not come from a stationary process, it is desired that the change point detection method exhibits a stationary performance. To ensure the method is independent of the run length, the placement of the actual

change point is taken uniformly at random over the time domain of the simulation. To account for the possibility of no change point, the probability of having no changes during the whole detection time interval is taken to be non-zero.

To simulate the change-point, the time domain is discretized at each time unit. Each time unit is then selected as a candidate changepoint with probability p. There is therefore a non-negligible probability $(1-1/T)^T$ of having no changepoints in a time interval $[0, T]$. As T gets larger this probability converges to e^{-1}.

Another way to simulate the change-point is to draw a r.v. u uniformly at random from the interval (0,1), i.e. $u \sim U(0, 1)$. If u is larger than e^{-1}, then select a change point time uniformly at random in the time domain of the simulation, otherwise no change point will occur for the given run. This assumes that p is sufficiently small and T sufficiently large.

2.6.3 Output Statistics

There are four outcomes for the change point detection test. If a true change-point existed in the run, there is a false positive if the change is detected before the change-point, a true positive if it is detected after the change-point and a false negative if no change point is detected. On the other hand, if the run has no true change point, there can only be a false positive or a true negative. All of these outcomes are aggregated to obtain the number of true positives, true negatives, false positives and false negatives for that experiment. These statistics are summarized below.

- True positive (TP) number: The number of times a triggered change is detected.
- False positive (FP) or false alarm number: The number of time a change is detected before the actual change point or in the absence of a change point.
- True negative (TN) number: The number of times no change is detected when no change has been triggered.

- False negative (FN): The number of times no change point is detected for an entire run that had a triggered change point.

From these general statistics, it is possible to reproduce the metrics of interest for an hypothesis test such as precision (also called specificity or positive predictive value), recall (also called sensitivity), true positive rate (TPR), false positive rate (FPR), correct detection rate (CDR) and false alarm rate (FAR).

$$\text{TPR} = \frac{TP}{TP + FN} \tag{2.43}$$

$$\text{FPR} = \frac{FP}{FP + TN} \tag{2.44}$$

$$\text{recall} = \frac{TP}{TP + FP} \tag{2.45}$$

$$\text{precision} = \frac{TP}{TP + FN} \tag{2.46}$$

$$\text{CDR} = \frac{TP + TN}{1} \tag{2.47}$$

$$\text{FAR} = \frac{FP}{FP + TP} \tag{2.48}$$

ARL_1, the expected time to detection after a change-point, is also considered as a metric of interest. It is computed over all the runs by averaging the detection delays.

2.7 Setting up the single change-point detection for a M/M/1 Queue

We run the simulation for 100,000 time units with a single change point. The change consists in a relative increment in traffic intensity at a change point generated from the aforementioned procedure. The set of relative change points is: $\Delta = \{\delta\rho : \delta\rho = \frac{\rho_1 - \rho_0}{\rho_0}\}$ is $\Delta \in \{-0.5, -0.25, 0, 0.25, 0.5, 0.75, 1.0, 1.2\}$. The traffic intensity after the change ρ_1 is related to the traffic intensity before the change ρ_0 by the following relation: $\rho_1 = (1 + \delta\rho)\rho_0$. The simulation starts with an arrival rate λ_0 and a service rate $\mu = 1.0$. Only the arrival rate is effectively changed in the present scenario.

The detection algorithm operates on three measurements: the waiting time, the instan-

taneous time spent in queue (age of the process) and the instantaneous queue length. The instantaneous queue length $Q(t)$ is recorded by looking at the number of elements in the queue before each arrival or departure. The waiting time of the queue $W(t)$ is obtained by looking at the time spent in queue for an entity about to be served. It is the difference between the start of service time and the time of arrival of the entity in the queuing system. The age of the entity in the queue or the time spent in queue $A(t_k)$,, is the time an entity i has spent in queue by time t_k. Contrarily to the waiting time, this statistic is instantaneous. It can be collected at any instant during the sojourn of an entity in the queue. In the present work, the average age of the process $\bar{A}(t_k)$ is collected for all the entities in the queue at a given t_k after an arrival or a departure.

The simple way to compute the age of a process statistic is to average the time spent in queue of all entities that have arrived at the queuing system and not gone through one of the servers yet per equation Equation 2.49. An alternative method that is more computationally efficient is to continuously update the mean age of the entity for each observation, see equation Equation 2.50. After each time update, there are only three possible changes to the size of the queue. The queue increases by 1 after each arrival that is not preceded by a departure if the server is currently busy. The queue decreases by one following a departure. The size of the queue stays the same otherwise. Let us define the size of the queue at time t as $Q(t)$. $\ell_Q(t_k) = Q(t_k) - Q(t_{k-1}) \in \{-1, 0, 1\}$ is the change in queue size.

$$\bar{A}(t_k) = \sum_{i \in Q} \frac{(t_i - t)}{|Q|} \tag{2.49}$$

$$A(t_k) = \begin{cases} \left[\frac{a(t_{k-1})Q(t_{k-1}) + (t_k - t_{k-1})\ell_Q(t_k)}{Q(t_k)}\right]_+ & \text{if } Q(t_k) > 0 \\ 0 & \text{if } Q(t_k) = 0 \end{cases} \tag{2.50}$$

After the waiting times, the times spent in queue and the queue lengths are observed, they are independently binned into NOBMs before being analyzed for a change point using

Ross' implementation of GLRT for Gaussian data [35].

Outcomes of the detection experiment

For each relative deviation in traffic intensity ($\delta\rho$) and each change of batch size, a new detection test is run 1,000 times. Each run resulting in one of four outcomes: a true positive, a true negative, a false positive or a false negative. Then the correct detection rate (CDR), the detection delay (ARL_1), the true positive rate and the false positive rates are evaluated. We first compare and contrast the results of the change point detection based off time spent in queue, waiting times and queue lengths. The key take-aways from the experiments are then presented.

2.7.1 Performance of the test for different performance measures

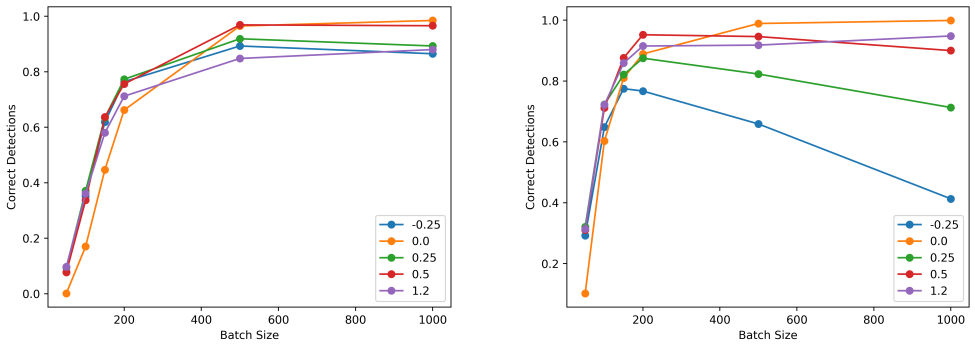
Across all experiments, we have varied the change point intensity ($\delta\rho$), the initial state of the queue (ρ_0) and the size of the NOBM. Only a few metrics characterize the quality of change point detection test: the Correct Detection Rate (CDR), the time to detection (ARL_1), the sensitivity and the specificity of the test. The relation between the different variables are highly nonlinear. Rather than trying to illustrate all the results by a single plot, we resort to pairwise comparisons of experiments with different performance measures for different $\delta\rho$, ρ and batch size.

Comparison between waiting times and mean age in queue

In a real system, there is a possibility that a job takes a very long time to be served. Whenever this occurs, the waiting time measure may become unavailable for a long time because of this particular job. By the time, that entity is served and a waiting time is observed, the dynamics of a queue may have totally changed. To handle this scenario, it is often desired to look at more instantaneous measures such as the average age of the process for the entities that are in the queue.

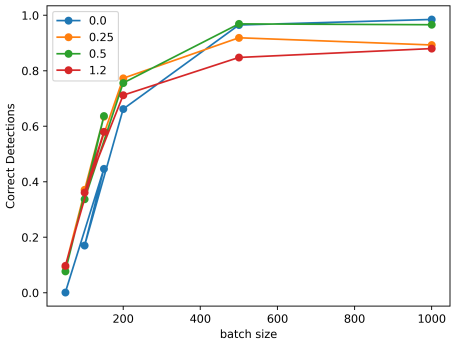
The average age of a process is the average time spent in queue by entities which are about to get served. The performance of the test is characterized by a competition between the capability of the test to quickly detect a change and the high variability in the quality of the test for large batches. It takes the least number of observations to reach normality with the age of a process as compared to the other two performance metrics, see Figure 2.5b. High correct detection rates can be achieved with low batch sizes when looking at the time spent in queue. Yet, there is a high variability in the observations as the queue gets longer. This is observed in Figure A.24. Batches that are too big actually lead to low correct detection probabilities when the signal is not strong (low $\delta\rho$).

There must therefore exists an optimum batch size when selecting a change point based off the age of a process, see figures Figure 2.4b, Figure 2.5b. Looking at the age of the process, the maximum correct detection rate is obtained for a batch of size 200 for all $\delta\rho$ except for the case of no change $\delta\rho = 0.0$. Although waiting times takes more samples to provide precise detection, they can provide higher CDR as compared to using the age in queue as a measure of performance when $\rho = 0.5$. When the arrival rate is high, there are more observations and the age in queue is quite reliable as seen in Figure 2.5b for any size of the change in traffic intensity.

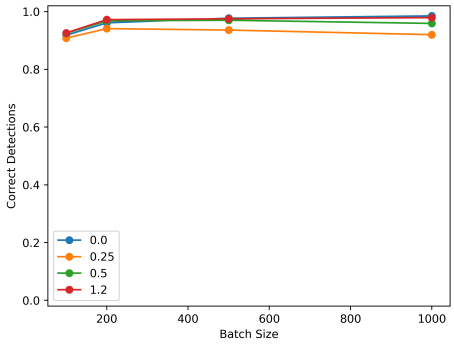


(a) Waiting Times (b) Age of Process

Figure 2.4: CDR vs. Batch Size ($\rho = 0.5$)



(a) Waiting Times



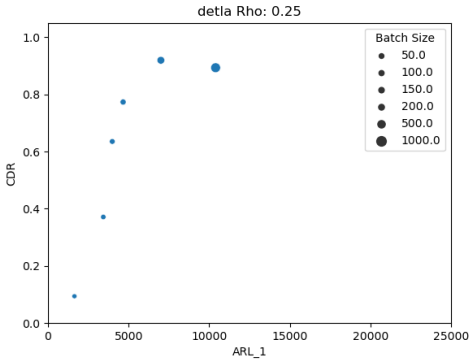
(b) Age of Process

Figure 2.5: CDR vs. Batch Size ($\rho = 0.75$)

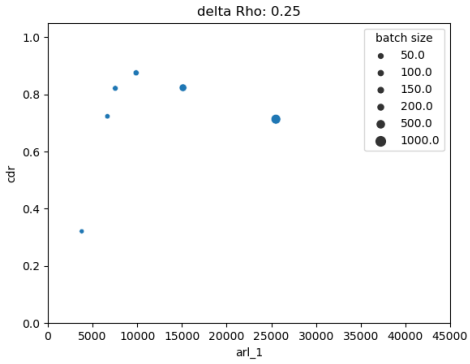
Change point detection based on the mean age in a queue provides both the shorter detection delay and the lower rate of false positive for the same initial conditions. Hence, it is expected that the detection delay is higher when waiting times are observed instead of average time spent in the queue. Small batch sizes tend to provide higher correct detection rates when looking at the age in queue. Batches of sizes greater than 500 tend to have both a higher detection rates and lower CDR for all $\delta\rho$ below 1. For a $\delta\rho$ of 1 or more, the age of a process tend to provide a higher CDR, while still paying the price of a longer detection delays.

Although the correct detection rate tends to be higher when looking at the age of a process, the detection delay is similar or higher for waiting time observations especially when considering large batch sizes.

When looking at CDR against detection delay, one can see that detecting small changes can get expensive not only in terms of delays but also in terms of error rate (figures Figure 2.6a and Figure 2.6b). All of the plots show that there is a batch size that maximizes the CDR. The plots also illustrate the trade-offs that would need to be made to reduce the detection delay.

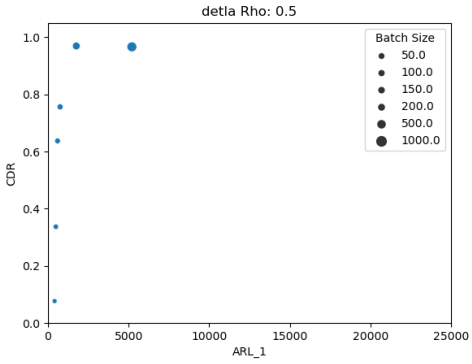


(a) Waiting Times

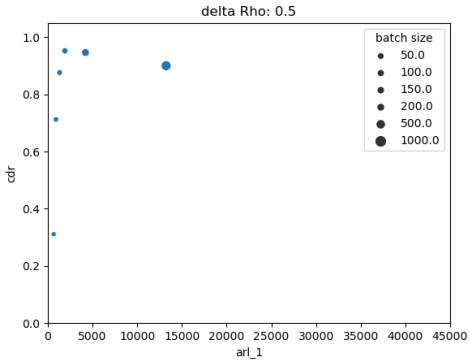


(b) Age of Process

Figure 2.6: CDR vs. ARL_1 ($\rho = 0.5$, $\delta\rho = 0.25$)

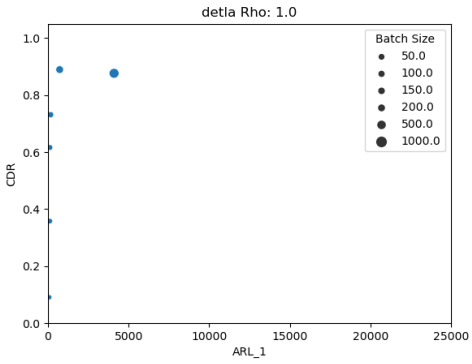


(a) Waiting Times

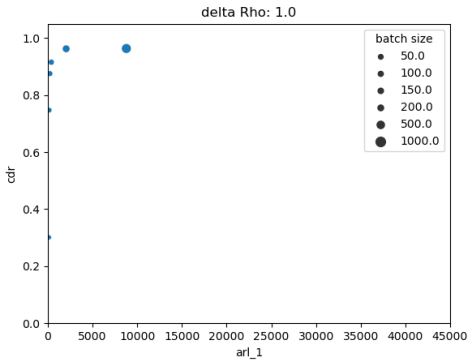


(b) Age of Process

Figure 2.7: CDR vs. ARL_1 ($\rho = 0.5$, $\delta\rho = 0.5$)



(a) Waiting Times

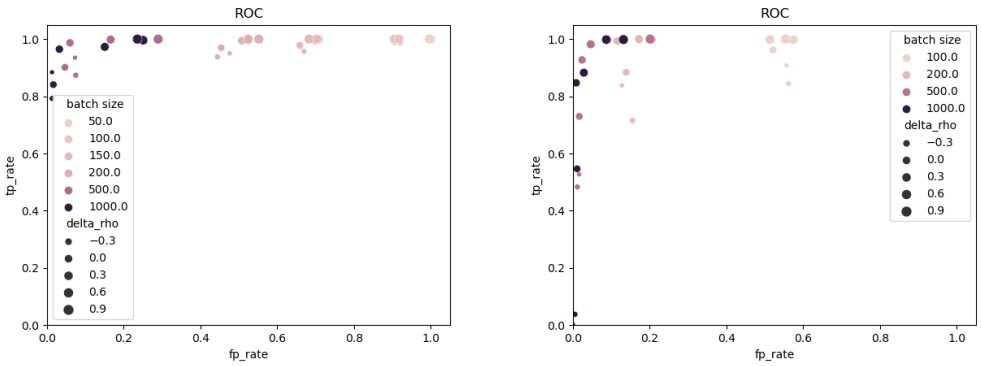


(b) Age of Process

Figure 2.8: CDR vs. ARL_1 ($\rho = 0.5$, $\delta\rho = 1.0$)

Receiver Operating Characteristic (ROC) for Waiting Times and Time Spent in Queue

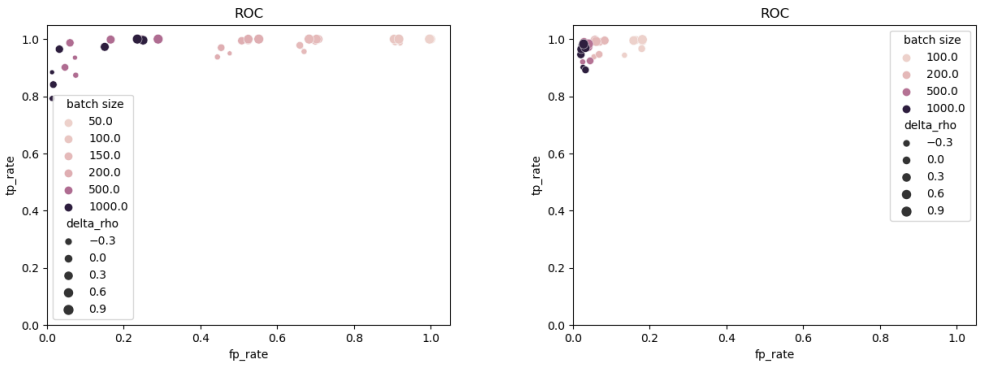
The Receiver Operating Characteristic (ROC) curve is a typical view of the trade-off between true positive and false positive rates for classification test. It illustrates how much error must be accepted in order to achieve a high rate of correct detection. As a test is tuned to deliver more correct detections (moving up on the Receiver Operating Characteristic (ROC) chart), it starts to also trigger more false detection (moving to the right on the ROC chart). A test that triggers every changes would be situated on the top right corner, rejecting and accepting every change. According to the ROC curve, the age of the process always performs better than the other two measures providing higher true positive rates with lower false positive rates for every ρ, $\delta\rho$ and batch size.



(a) Waiting Time

(b) Age of process

Figure 2.9: ROC curves for $\rho_0 = 0.5$

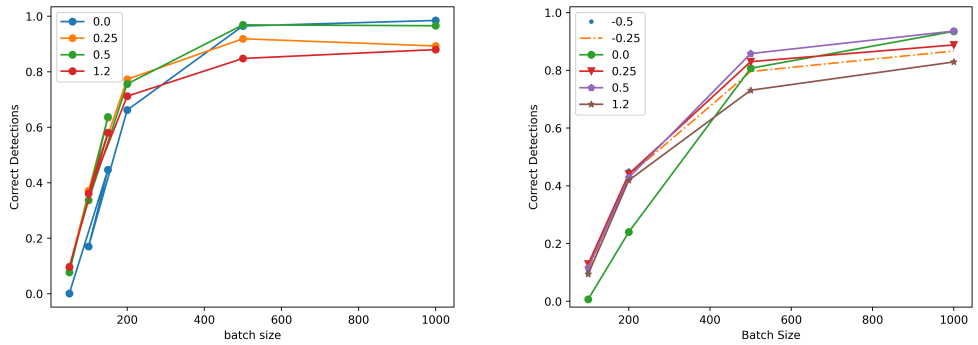


(a) Waiting Time (b) Age of process

Figure 2.10: ROC curves for $\rho_0 = 0.75$

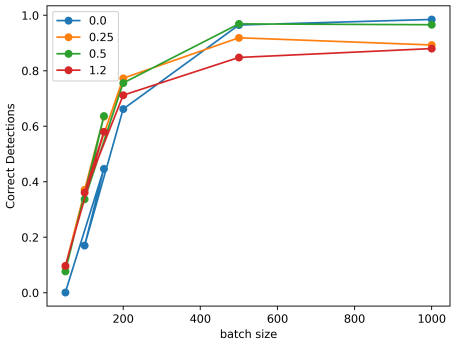
Comparison between Waiting Times and Queue Length

As previously seen, it takes larger batch sizes to achieve normality when a change point is triggered based on a queue length as opposed to a wait time or time spent in queue performance measure. It can be observed on figures Figure 2.11 and Figure 2.12, that it takes larger batch sizes to accurately detect a change in the queue dynamics. On the other hand, observing the queue length provides a clear benefit in detection delays. Although the test may not provides the same amount of accuracy, it takes less time to get an answer. This suggests the possibility to combine the different performance measures to create a faster and accurate test. This line of thinking will be explored in the next section.

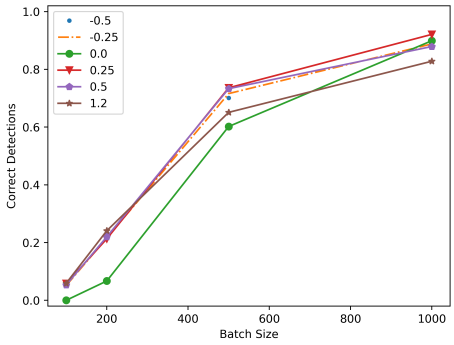


(a) Waiting Times (b) Queue Length

Figure 2.11: CDR vs. Batch Size ($\rho = 0.5$)

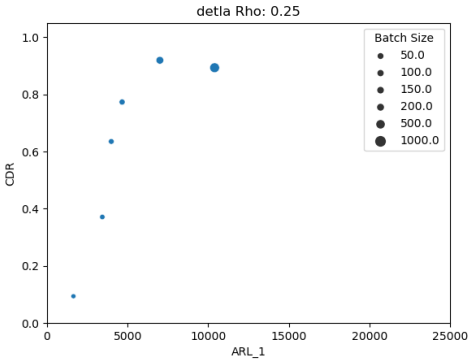


(a) Waiting Times

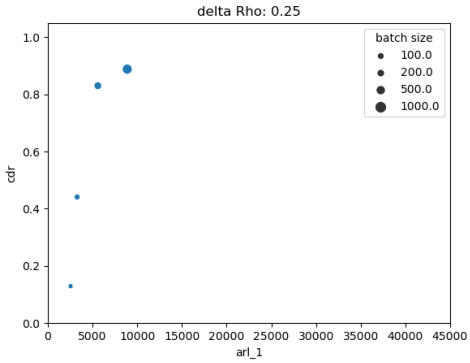


(b) Queue Length

Figure 2.12: CDR vs. Batch Size ($\rho = 0.75$)

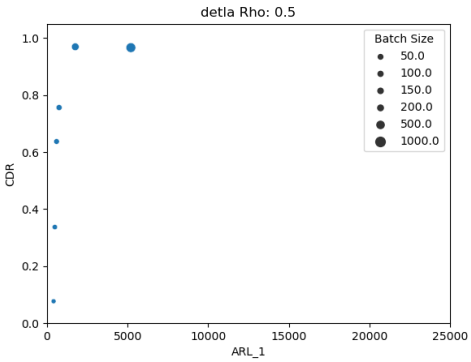


(a) Waiting Times

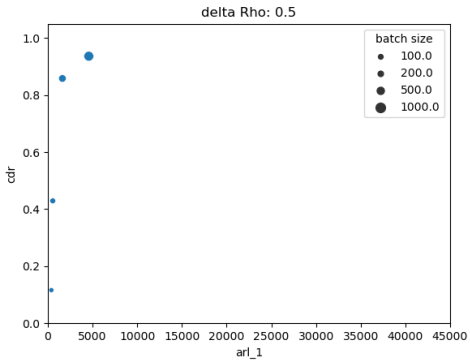


(b) Queue Length

Figure 2.13: CDR vs. ARL_1 ($\rho = 0.5$, $\delta\rho = 0.25$)

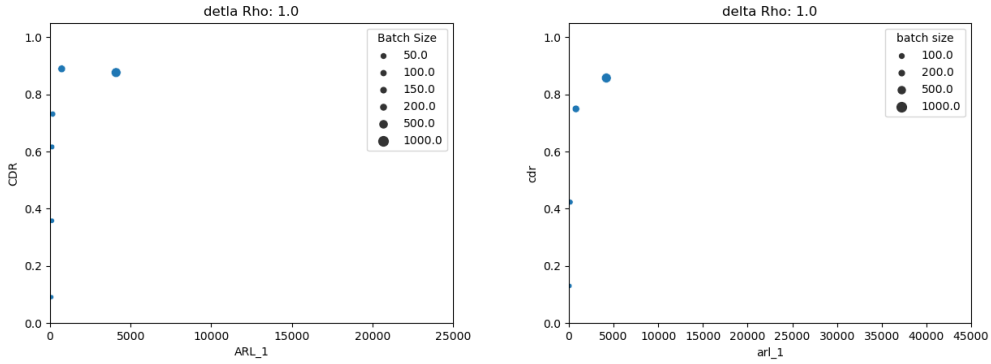


(a) Waiting Times



(b) Queue Length

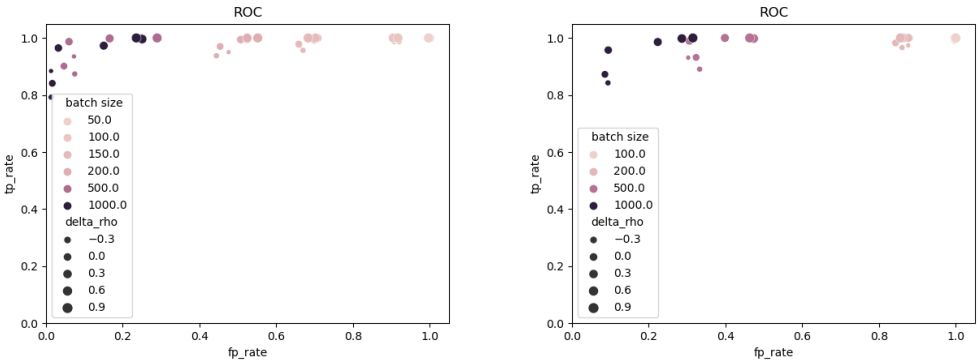
Figure 2.14: CDR vs. ARL_1 ($\rho = 0.5$, $\delta\rho = 0.5$)



(a) Waiting Times (b) Queue Length

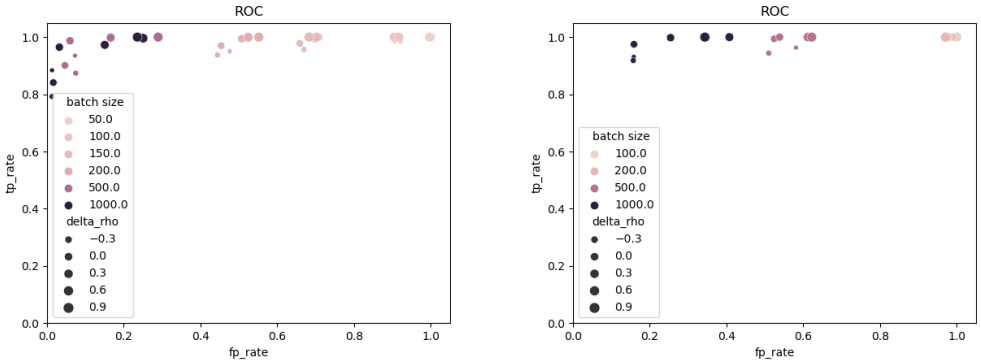
Figure 2.15: CDR vs. ARL_1 ($\rho = 0.5$, $\delta\rho = 1.0$)

Based on ROC, the detections made using the queue length are strictly dominated by the detections from waiting times. Meaning that detections based off queue length achieve both a higher false positive and a lower true positive rates for the same configuration.



(a) Waiting Time (b) Queue Length

Figure 2.16: ROC curves for $\rho_0 = 0.5$



(a) Waiting Time (b) Queue Length

Figure 2.17: ROC curves for $\rho_0 = 0.75$

2.8 Picking a batch size

The normality of samples is highly affected by the batch size. The good performance of the test therefore, requires the right selection of batch size for a specific intensity of change point depending on the performance measure.

2.8.1 Empirical Results for Detection on One Measure

The performance of the detection is quantified by looking at the probabilities of correct or incorrect detection given the absence or presence of a change-point. This approach accounts for but is not limited to Type I and Type II errors. Let $X \in \{A, Q, W\}$ be the outcome of a test for an age measure (A), queue length measure (Q) and waiting time measure (W). $X+$ denotes the detection of a test after the change-point and $X-$ denotes the absence of a detection. Then each detection outcome of a test defines an hypothesis outcome depending on whether there was a change-point or not, see equations Equation 2.52. This can easily be generalized to more than one test as will be showed in the section on combining multiple measures.

$$X + |\text{Changepoint} \iff TP$$

$$X - |\text{Changepoint} \iff FN$$

$$X + |\text{No Changepoint} \iff FP$$

$$X - |\text{No Changepoint} \iff TN \tag{2.51}$$

$$\tag{2.52}$$

Figure Figure 2.18 illustrates the decision logic to assign a change point.

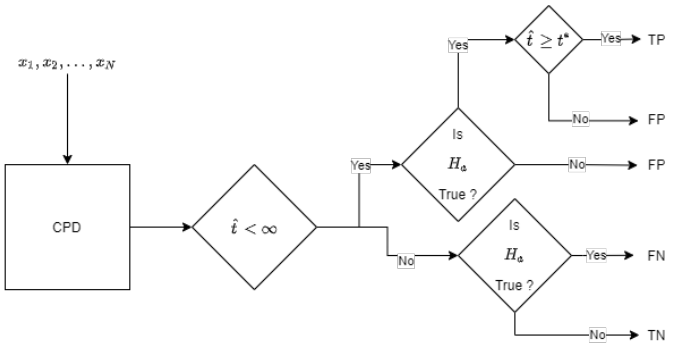


Figure 2.18: Hypothesis Test Outcome for a Change-point Detection Test

Armed with these definitions, it is possible to compare the performance of the change-point detection for different types of measure.

2.8.2 Outcomes of the Detection Test

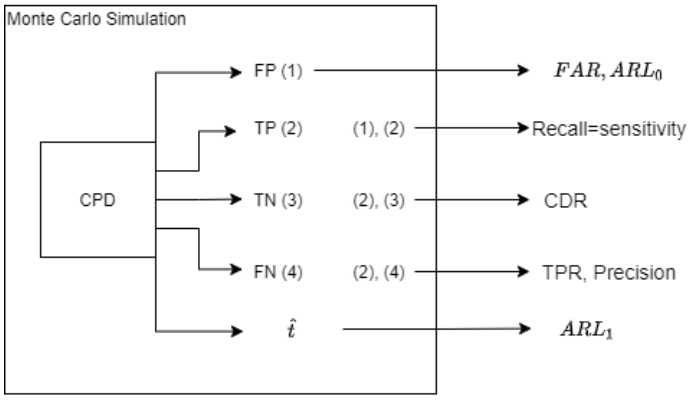


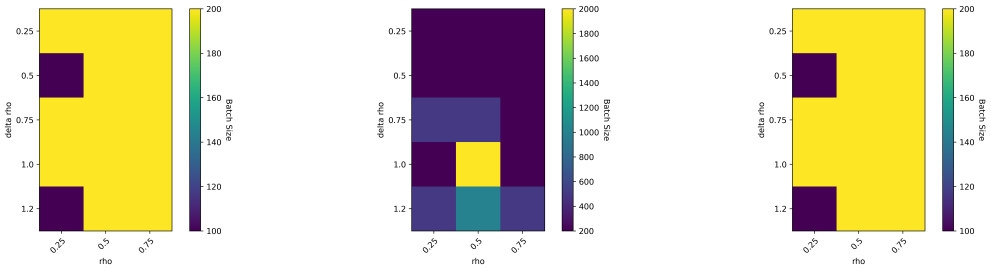
Figure 2.19: Simulation Output Statistics

The results from the simulated experiments can inform the selection of a batch size prior during live testing. The choice of batch size is a trade-off of three decisions: how low is the desired false alarm rate (FAR), how low can the false negative rate be made and how high a detection delay can be tolerated. All of these secondary performance measures can be simply inferred from the test outcomes as illustrated in figure Figure 2.19. Minimizing on only one variable can cause some undesired outcomes such as having a very quick test with a high error rate. For this reason, it is desirable to optimize on at least two objectives or to create a rejection region based on one metric and optimizes on another within the acceptance rejection such as for a Neyman-Pearson detection rule [4]. For the single detection tests, three different heuristics have been developed to tune the test.

- Picking the batch size with the lowest FAR (Metric 1)
- Fixing FAR at or below 0.05 and picking the batch size with the lowest false negative rate (Metric 2)
- Fixing FAR at or below 0.05 and picking the batch size with the shortest detection delay (Metric 3)

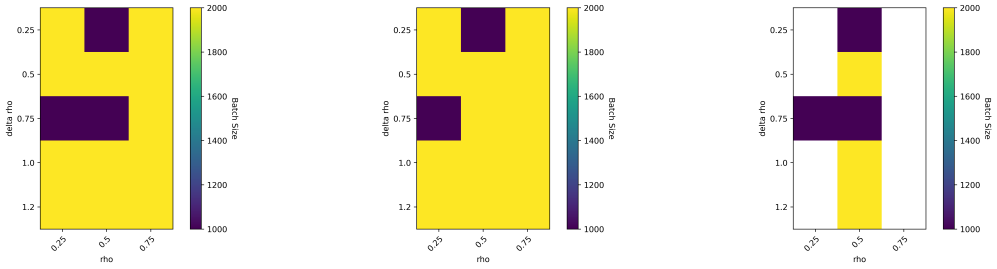
After testing for batch sizes ranging from 100 to 2,000, the performance of all the detection tests were evaluated with respect to the three performance metrics above to define the "best" batch size. For each combination of traffic intensity (ρ) and change in traffic intensity ($\delta\rho$), a batch size was selected following one of the heuristics. The best batches were then color-coded in a heatmap on figures Figure 2.20, Figure 2.21 and Figure 2.22. A square with no value was left blank. The detection delays were computed for batch sizes no greater than 1,000. The detection based off queue lengths had a FAR above the desired threshold value of 0.05. For these instances, no change point could be detected. The test is incomplete which is represented by a white square in figures Figure 2.21c and Figure 2.22c. Hence the heatmap for scenario 3 shows as incomplete for some combinations of change intensity ($\delta\rho$) and batch size.

For scenario 2, looking at time spent in queue provides the smallest batch size. Whereas for scenario 1 and 3, waiting times provide the best performance. In all cases, the queue length performance measure requires the largest batch size to ensure the desired performance.



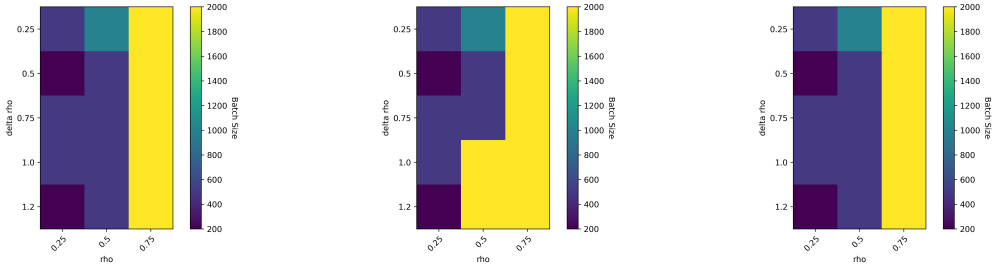
(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 2.20: Heatmaps for Time Spent in Queue



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 2.21: Heatmaps for Queue Length Observations



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 2.22: Heatmaps for Waiting Times

After analyzing the heatmaps, a few conclusions can be drawn. The detections based on age or waiting time in queue are the ones requiring the smallest batch size for all three metrics. Hence those detections have the shortest delays. Any detection based off queue length would require at least 1,000 elements to provide an acceptable test according to our FAR criterion. Higher batch sizes are needed when starting at bigger traffic intensities and for lower changes in traffic intensities. On a few rare instances, the observations based on waiting times could be competitive with the ones based off instantaneous time spent in queue. This is notably the case for $\rho = 0.25$ and $\delta\rho = 1.2$.

2.9 Key take-aways from the simulated experiments

From the comparison of the detection and the properties of the batches, we can draw some conclusions.

- Smaller changes in traffic intensities require longer detection delays.
- Larger batch sizes cause larger detection delays.
- Stronger signals ($\delta\rho$) result in larger correct detection rates
- Larger batch sizes also in general produce higher CDR and true positive rates.
- There is not significant gain in accuracy of increasing the batch, past a certain batch size.
- Observations based on queue length provides the quickest detection for the same batch size and signal conditions.
- Observations based on the time spent in queue provides the most accurate detection for smaller batches, batches of size less than 500. This is especially true when considering the false alarm rates, false negatives and detection delays.
- It can be useful to combine performance measures. This is especially true if the observations have different frequencies.

2.10 Conclusion

In this chapter, we have reviewed a general framework to detect changes in traffic intensities for a M/M/1 queue by performing a generalized likelihood ratio test on the non-overlapping batch mean waiting times, times spent in queue and queue length observations. It was showed that the non-overlapping batch mean procedure creates reasonable normal i.i.d samples with low auto-correlation and moderate to high p-values. Furthermore the test was able to provide high correct detection rates with the proper batch size selection. This high correct detection rates can be to the detriment to the detection time. The higher correct detection rates obtained for the times spent in queue come with longer detection delays. It suggests the possibility to optimize the selection of batch size to optimize detection delay

while controlling for correct detection. The results for different measure types and batch sizes show that different measures had different advantages, suggesting the possibility to combine the measures and run them in parallel online to achieve a high confidence in the change point for unknown traffic utilization and times at which the change points occur.

CHAPTER 3
COMBINING MULTIPLE PERFORMANCE MEASURES FOR CHANGE-POINT DETECTION

It is often possible to monitor more than one performance measure at a time. In the case where each measure can be used to detect a change, how do we adjust the detection results to decide when to stop collecting observations for the test? In the single observation cases, detection tests are designed with Type I error probability (e.g.: $P(NoChange|A+)$) and Type II error probability (e.g.: $P(Change|A-)$) in mind. What do those results and conditions look like for multiple observations? This problem of multiple observations hypothesis testing is treated in two distinct ways. First, some general theoretical bounds on the union of the events are derived. Although, they are expected to be conservative they can be used to complement our understanding of the detection policy and to validate some other results with join probabilities. A second approach looks at the joint conditional probability of a correct detection and the likelihood of the observations given a success in order to derive importance metrics such as CDRs and fars from empirical experiments. A regression model generalizes the results to unseen combinations of traffic intensities. A comparison of the performance of the detection test from a single observation against the performance of a policy that combines all the observations concludes the chapter.

3.1 Union Bounds

Union bounds such as Boole's inequality Equation 3.1 and Bonferroni inequalities are exact results that provide upper and lower bounds to the union and intersection of events provided

that the events are independent.

$$Pr(\bigcup_{i=1}^{n} A_i) \leq \sum_{i=1}^{n} Pr(A_i) \tag{3.1}$$

If we look at the complement, we obtain:

$$Pr(\bigcap_{i=1}^{n} A_i) \geq 1 - \sum_{i=1}^{N} Pr(\bar{A}_i) \tag{3.2}$$

Using the Inclusion-Exclusion principle, it is still possible to obtain exact bounds for unions of events. The Inclusion-Exclusion principle is derived from a well known property of sets

$$\begin{aligned} card(A_1 \cup A_2 \cup \ldots \cup A_n) = \sum_{1 \leq i \leq n} card(A_i) - \sum_{1 \leq i < j \leq n} card(A_i \cap A_j) + \\ \ldots + (-1)^{n-1} card(A_1 \cap A_2 \cap \ldots \cap A_n) \end{aligned} \tag{3.3}$$

$$Pr(A_1 \cup A_2 \cup \ldots \cup A_n) = \sum_{1 \leq i \leq n} Pr(A_i) - \sum_{j=1}^{n} \sum_{i<j} Pr(A_i \cap A_j) + \ldots + (-1)^{n-1} Pr(A_1 \cap A_2 \cap \ldots \cap A_n) \tag{3.4}$$

Let's define some identities that will be used to present other Bonferroni inequalities:

$$\begin{aligned} S_1 &\equiv \sum_{i=1}^{n} Pr(A_i) \\ S_2 &\equiv \sum_{i<j} Pr(A_i \cap A_j) \\ S_3 &\equiv \sum_{i<j<k} Pr(A_i \cap A_j \cap A_k) \\ &\vdots \\ S_l &\equiv \sum_{i_1 < i_2 < \ldots i_l} Pr(A_{i_1} \cap A_{i_2} \cap \ldots \cap A_{i_l}) \end{aligned}$$

Then we have the following two Bonferroni inequalities

$$Pr(\bigcup_{i=1}^{n} A_i) \leq \sum_{j=1}^{l} (-1)^{j-1} S_j \qquad \text{if l is odd and } l \leq n \tag{3.5}$$

$$Pr(\bigcup_{i=1}^{n} A_i) \geq \sum_{j=1}^{l} (-1)^{j-1} S_j \qquad \text{if l is even and } l \leq n \tag{3.6}$$

The previous two inequalities are conservative. A tighter bound is presented in equation Equation 3.7. It was used to validate all the joint probabilities obtained in the next section.

$$\frac{2}{3} S_1 - \frac{1}{3} S_2 \leq Pr(A_1 \cup A_2 \cup A_3) \qquad \leq S_1 - \frac{2}{3} S_2 \tag{3.7}$$

$$S_1 \equiv \sum_{i=1}^{n} Pr(A_i) \tag{3.8}$$

$$S_2 \equiv \sum_{i<j} Pr(A_i \cap A_j) \tag{3.9}$$

3.2 Empirical Evaluation of Joint Distributions

In an online test, waiting times, queue length and time spent in queue observations may all be available but not necessarily simultaneously. This is because queue length and time information may not come from the same device. When more than one of these measures is available, it is possible to combine the observations to increase the power of the test for the same significance level. Combining such tests require the knowledge of the power of individual tests, but also of their correlation. As can be seen in figure Figure 2.4, the best batch size may not be the same for different test type. In this work, multiple computer experiments were performed to obtain the joint conditional probability of a change. These results were then validated against the union bounds in Equation 3.7.

3.2.1 The value of information

Each performance measure comes with different costs and associated delays. It might be more difficult to compute precise waiting times or time spent in a queue rather than cycle time or time spent in the system because of the added geographical precision required to discriminate between queue and servers. Furthermore, wrong decisions also have a cost. This is the case when a change in distribution informs the decision to open or close a server or change the staff level.

3.2.2 Setting for the multiple change-point detection

To develop a combined detection test for multiple measures, we assume that each observation comes sequentially. With each new observation, new batches are formed. For the new batches, a test for the change point is made controlling for the Type II error. The experiment is stopped as soon as a decision can be made with enough confidence. This could mean stopping the tests after a change has been identified at the right significance level. The questions then become:

- Which test or combination of tests to use to minimize detection delay?
- What combination of tests to use to minimize cost while satisfying some detection delay constraints?

3.2.3 General notation for joint probabilities

Just as in the single measure test, we look at the hypothesis test outcome as conditional probabilities Equation 2.52.

The overall approach is to fix the significance level for the combined test and then backtrack the significance levels and the required batch size for each single test.

To obtain the joint significance level, it suffices to generate to generate the joint error probabilities at each batch size.

Instead of computing all of true positives, false positives, false negatives and true negatives outcomes which would results in $4^3 = 64$ combinations, we compute the equivalent probability of a positive or negative result for each test given that a change-point was or was not triggered during a measurement event. Let A, Q, W be the outcomes of correct detection or no detection for tests based of the age of a process, the queue length and the waiting times respectively. The outcomes of the tests can be written as Equation 3.11.

$$Pr(A^{\pm}, Q^{\pm}, W^{\pm}|Changepoint) \tag{3.10}$$

$$Pr(A^{\pm}+, Q^{\pm}, W^{\pm}|NoChangepoint) \tag{3.11}$$

For a given test T_i, the mapping between this definition and the hypothesis test outcome joint probability is as follows:

This redefinition already accounts for impossible combination such as the simultaneous outcome of a true negative and a false negative or a true negative and a true positive.

Once these probabilities are obtained they can be used to obtain bounds on the significance levels for any combination of tests.

$$Pr(Changepoint|A^+, Q^+, W^+) = \frac{\#(A^+, Q^+, W^+, \text{with a changepoint})}{\Delta} \tag{3.12}$$

$$= \frac{\#(A^+, Q^+, W^+, \text{with a changepoint})}{\text{\# of changes}} \tag{3.13}$$

Where

$$\Delta = \#(A^+, Q^+, W^+, \text{ with a changepoint }) + \#(A^+, Q^+, W^+, \text{ with no changepoint })Pr(A^+, Q^+, W^+|Changepoint)$$

3.2.4 Joint Detection Simulation Results

Figures Figure 3.1 and Figure 3.2 illustrate the two possible conditional probabilities that can be extracted from the simulations as defined in equations Equation 3.12 and Equation 3.13.

The conditional probability of a specific combination of tests outcomes given a change Equation 3.13 marks how likely it is to observe a certain outcome given a change. The three-way joint conditional probabilities are easy to manipulate to obtain one-way or two-way probabilities conditioned on the presence or absence of a change. These probabilities relate directly to the true positives, false positives, false negatives and true negatives by equations Equation 2.52. Therefore, they can be used during the analysis of new results but also to fine tune a test to a desired precision level as was done in the single measure case. Figure Figure 3.1 represents the ideal situations of having each test detects an existing change. Each number inside the Venn diagram denotes the probability that can be attributed to that specific outcome. For example, there is a probability of 0.0984 that a test based off the time spent in queue returns positive but the tests based off the waiting times and the queue lengths are both negative. Furthermore, if there is a change, there is a probability of 0.823 that all three tests will detect the change.

The conditional probability of change given an outcome, defined as Equation 3.12, gives the confidence we have in a certain outcome. It directly expresses the probability of seeing a change given an outcome. It can be used during a live test to present the confidence in the tests outcome to the user(s).

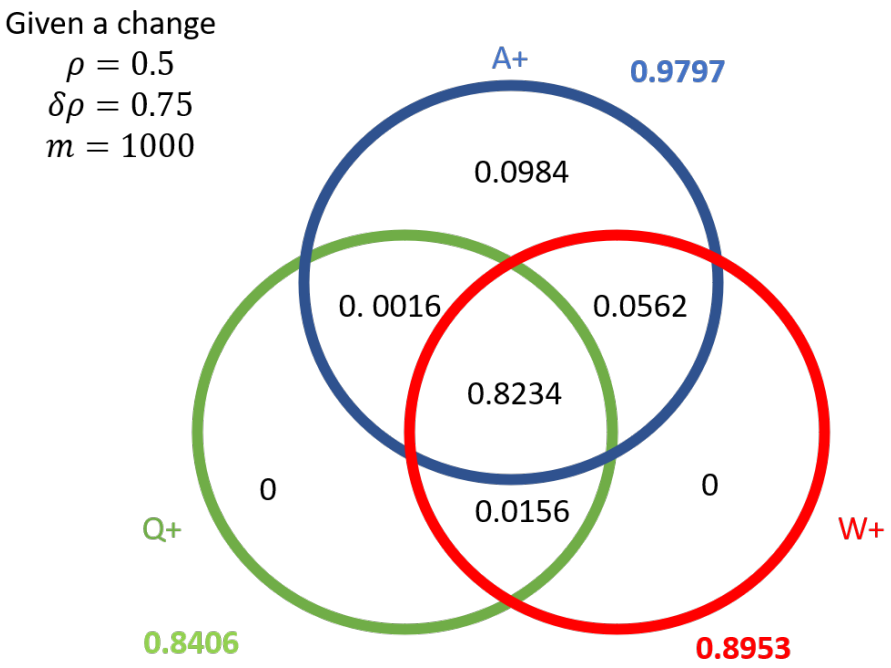


Figure 3.1: Example of Venn diagram of probabilities of joint hypothesis outcome conditioned on change

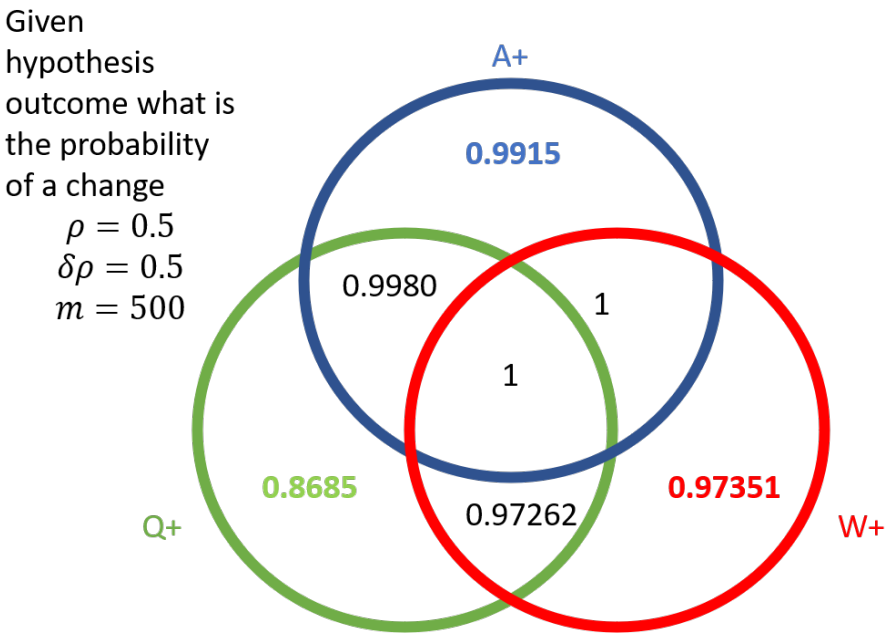


Figure 3.2: Example of Venn diagram of probabilities of change conditioned on hypothesis outcome combination

RMSE	
Cross-validation	0.018
Train Set	0.010
Test Set	0.070

3.2.5 Generalizing the results

We used XGBoost to generalize the results on the the conditional probability of a change given positive or negative detections for our three measures Equation 3.12. For all combinations of batch sizes, traffic intensities and relative changes in traffic intensities, there were 2018 data points to learn from for the different probabilities. The hypothesis was one-hot encoded to form eight new features The training-validation and the test sets were split in a 80-20 ratio. To ensure good generalization, a 5-fold cross-validation was performed while training the model to minimize the root mean square error (RMSE). The results of the training are summarized in table Table **??**.

3.2.6 Naive policy for multiple performance measures

Designing the right policy to detect a change requires combining an optimization over detection delay, type I and type II error. With the statistics obtained in the previous section and the regression presented above, it is possible to pick the most appropriate detection given an array of detection outcomes for different measures and batch sizes. To show what such policy can offer, a naive policy is proposed. For each set of observations, the policy chosen picks the earliest detection point with a change-point likelihood (e.g. $Pr(Success|A+, Q-, W-)$) greater than a given threshold. The threshold is taken to be 0.95 in the experiment. This is a naive policy has it doesn't account for type I and type II error directly.

3.2.7 Batch detection test design

To evaluate the performance of the detection framework, an experiment was performed that varies simulation setting parameters such as batch size, traffic intensities and relative change in traffic intensities. Each combination of parameters was ran 100 times to obtain the precision, sensitivity, correct detection rate and average detection delays for the test. Each simulation run was started from $t = 0s$ and ended at $t = 10,000s$. The change-point is selected at random according to the procedure described in section subsection 2.6.2 with an added difference that one of the relative change in traffic intensities was set to zero to obtain more true and false negative outcomes.

The parameters of the tests are the following:

- Batch sizes: $\{50, 100, 200, 300, 500, 1000, 1500, 2000\}$
- Traffic intensities: $\rho \in \{0.25, 0.5, 0.75\}$
- Relative changes in traffic intensities: $\delta\rho \in \{-0.5, -0.25, 0, 0.25, 0.5, 0.75, 1.0, 1.2\}$

3.2.8 Results from the detection test

Most performance curves for hypothesis testing are meant binary classification problems and do not account for the complexity of dealing with a time series for a change point detection. There is no curve that accounts for the detection delay. It is also not possible to distinguish between a false detection due to the absence of change or to the fact that the change was attributed to a point that appeared before the actual change when looking at batch detection. The CDR vs far plot Figure 3.3 shows how accurate the test is even as it becomes easier to throw a false detection. As can be seen in figure Figure 3.3, even a simple policy can dominate the results obtained by a single measure test over many tests. The price for the added accuracy is a longer average detection delay as reported in tables Table A.1, Table A.2, Table A.3 and Table A.4 in the appendix.

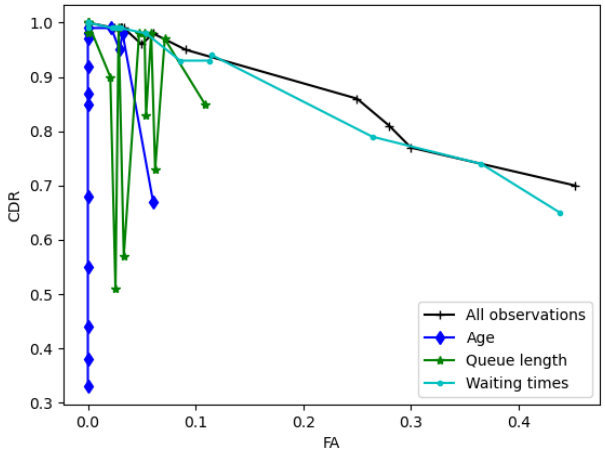


Figure 3.3: CDR vs. far

3.3 Conclusion

The problem when considering multiple performance measure is mainly to combine the detections obtained from multiple observations to create a stronger test, a test with greater power, significance level and lower detection delay. When the performance of the test is known for the individual performance measure, the problem reduces to finding joint conditional probabilities for the success of the detection and using them to design an optimal online detection test. Computing these probabilities is intractable and can only be approximated via techniques such as Expectation Maximization. Two approaches are presented to get closer to an optimal online detection test.

The first approach revisits some well known Bonferroni bounds and sharpen some of the results for the union of probabilities. These results provide a basis to bound the significance and power of different tests. In the second approach, the simulation is used to track multiple joint conditional probabilities of success. A boosted random forest model is then trained on those results to provide a generalization to unseen combinations of batch size, traffic intensities and relative changes in traffic intensities. A naive policy is then introduced and showed to outperform the individual measure tests.

The policy presented in this section is naive. It only considers the empirical results concerning the conditional probability of a success. To obtain robust results, it would be required to also consider probability of the two types of errors and perform a multi-objective optimization on Type I error, Type II error and detection delay to optimize the detection test. If the end goal is to develop a realistic online detection algorithm, the conditional probabilities of a correct or incorrect detections should be bounded or obtained even when combining performance measures with different batch sizes. This is because queue length, waiting times and time spent in queue (age) observations rely on different sensors and different computation methods. It is therefore unreasonable to assume that the observations should be available at the same frequencies and with batches of the same size or reliability.

CHAPTER 4
CONCLUSION

The thesis has treated the problem of gaining valuable information from multiple possibly incomplete observations. The first part of the thesis investigated the use of passenger cellular data to model passenger traffic inside SYD between gates and immigration. A framework to refine the service distribution at immigration within the simulation was presented. The in-loop optimization utilizes incomplete waiting times observations together with the throughput information at the servers to make the updates. The second part of the thesis narrowed the focus on the quick estimation of a change point in a single Markovian queue from a combination of wait times, queue length and time spent in queue observations. The study shows that the information gleaned from the three different observations had different performance characteristics and could be combined to provide a stronger test. The research opens many opportunities. The use of in-the-loop queuing parameters simulations and change-point estimations can lead to highly accurate simulations of queueing simulations that accounts for inherent non-stationarities in the dynamics of queuing systems and alert the system manager or any large congestion in time for the user to react. Further research can be done in combining the in-loop estimation of service rates with adaptive dynamic control or reinforcement learning to mitigate the effects of congestion in an airport or any other high traffic areas. The work on the change-point detections could easily be extended to make the batch detection an online batch detections where the observations are unveiled with time. To make the test framework more realistic, a method to estimate the arrival rate and service rate and hence the traffic intensity would be needed. On the theoretical side, the work could be augmented by deriving strong bonds on the Type I and Type II error conditional probabilities. These joint probabilities could then be used to optimize a detection test on multiple observations taken at different times. It would be a multiple observation

change point detection test with batches of different sizes in our framework.

Appendices

APPENDIX A

CHANGE-POINT DETECTION RESULTS

A.1 Appendix A: Q-Q plots

Normality results

A.1.1 Age in Queue

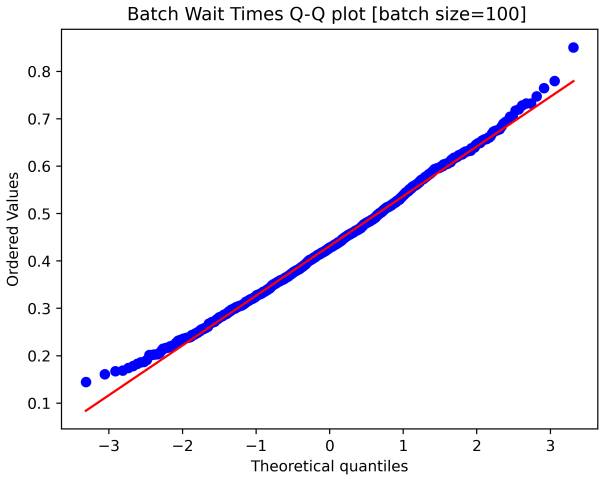


Figure A.1: Age of Process Batch of Size 100 ($\rho_0 = .75$)

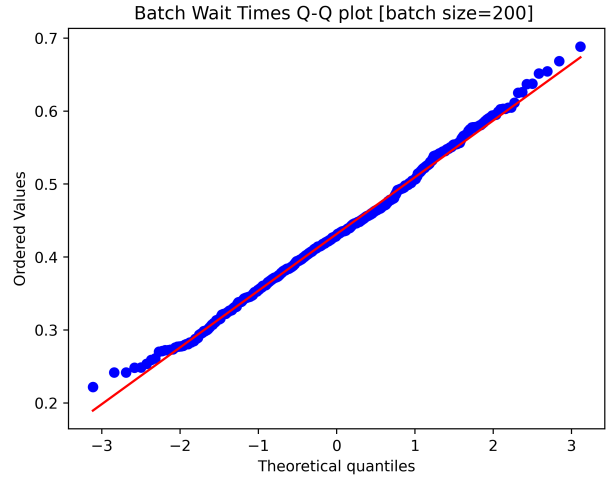


Figure A.2: Age of Process Batch of Size 200 ($\rho_0 = .75$)

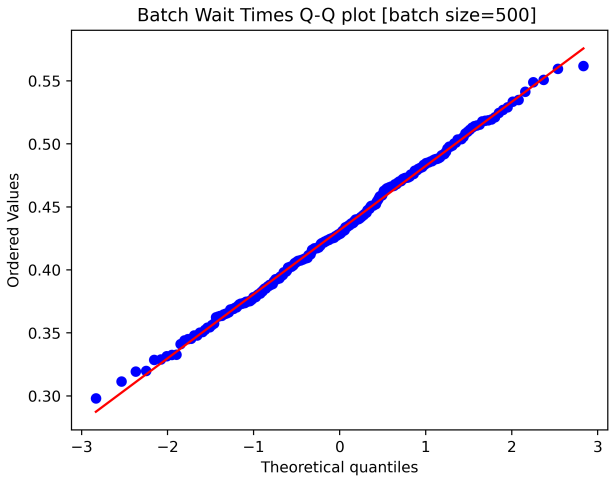


Figure A.3: Age of Process Batch of Size 500 ($\rho_0 = .75$)

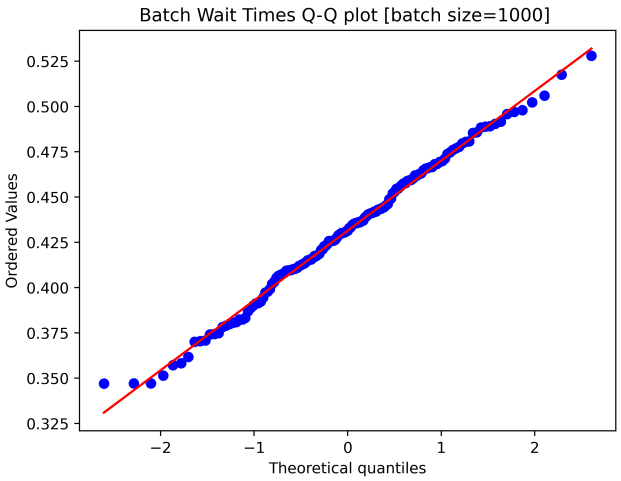


Figure A.4: Age of Process Batch of Size 1000 ($\rho_0 = .75$)

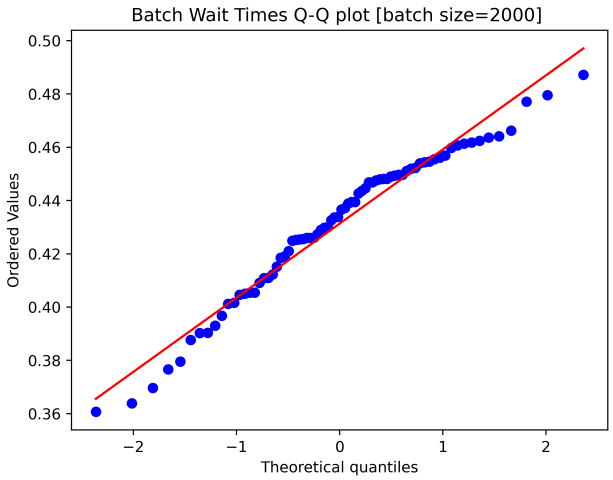


Figure A.5: Age of Process Batch of Size 2000 ($\rho_0 = .75$)

A.1.2 Waiting-Times

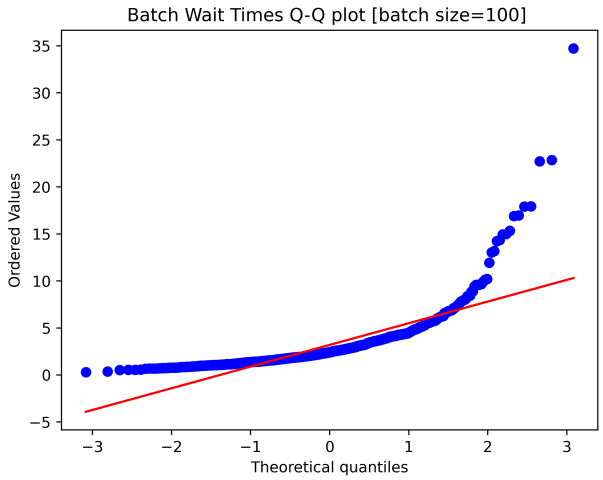


Figure A.6: Wait Times Batch of Size 100 ($\rho_0 = .75$)

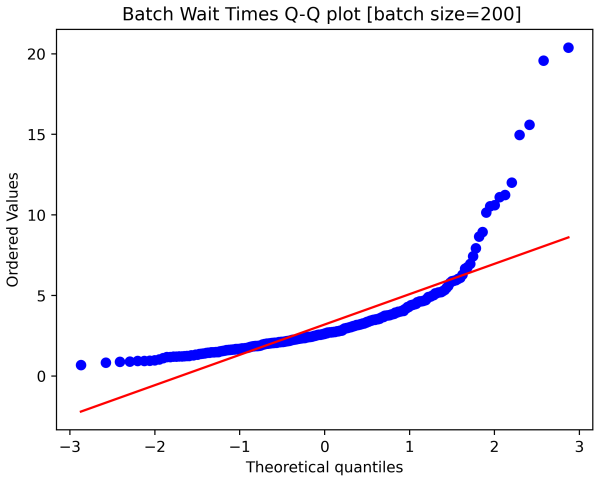


Figure A.7: Wait Times Batch of Size 200 ($\rho_0 = .75$)

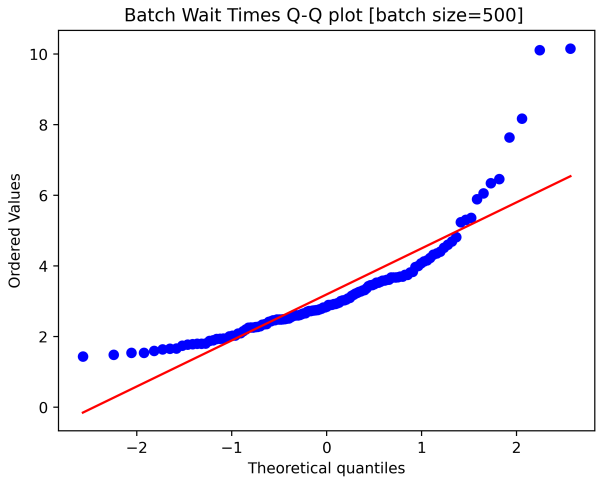


Figure A.8: Wait Times Batch of Size 500 ($\rho_0 = .75$)

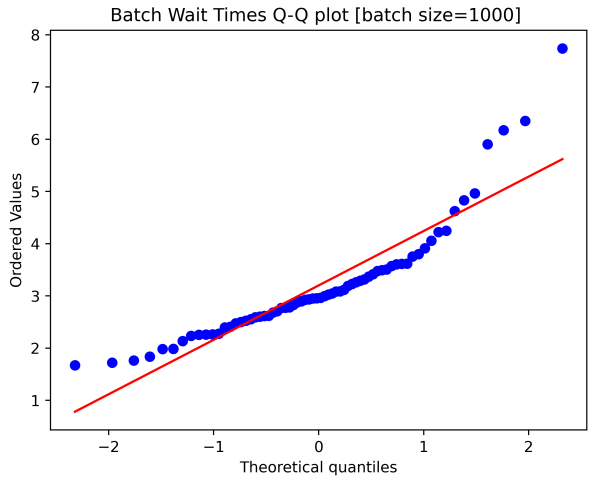


Figure A.9: Wait Times Batch of Size 1000 ($\rho_0 = .75$)

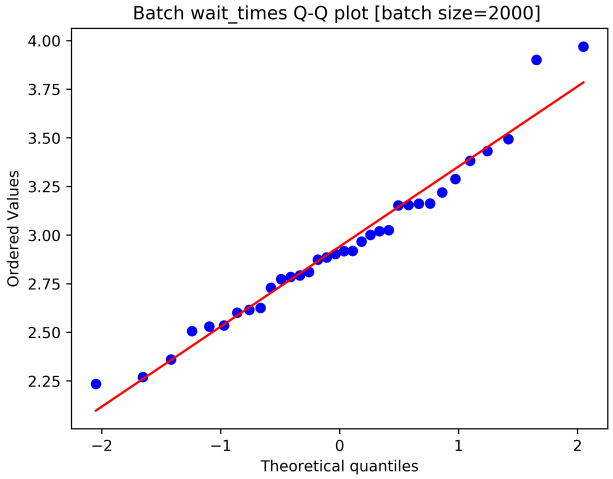


Figure A.10: Wait Times Batch of Size 2000 ($\rho_0 = .75$)

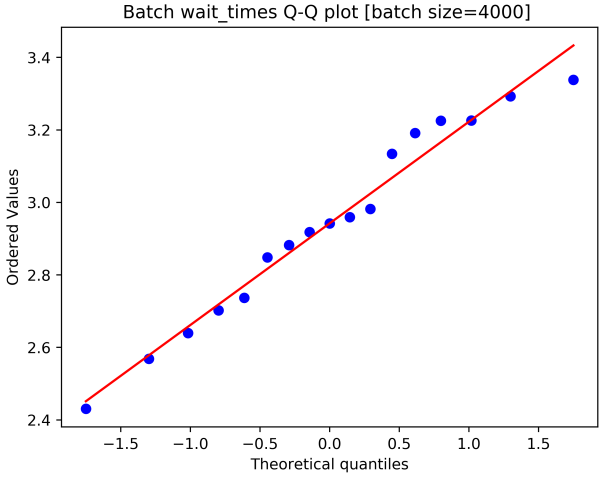


Figure A.11: Wait Times Batch of Size 4000 ($\rho_0 = .75$)

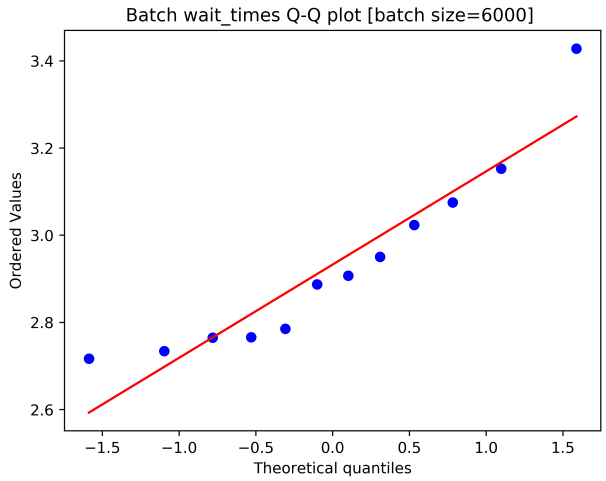


Figure A.12: Wait Times Batch of Size 6000 ($\rho_0 = .75$)

A.1.3 Queue Length

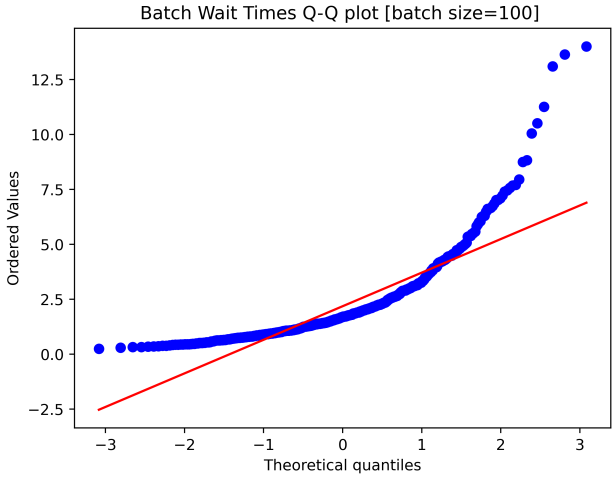


Figure A.13: Queue Length for a Batch of Size 100 ($\rho_0 = .75$)

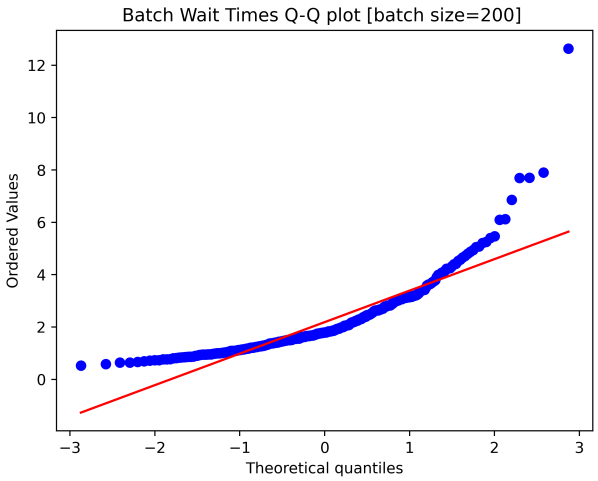


Figure A.14: Queue Length for a Batch of Size 200 ($\rho_0 = .75$)

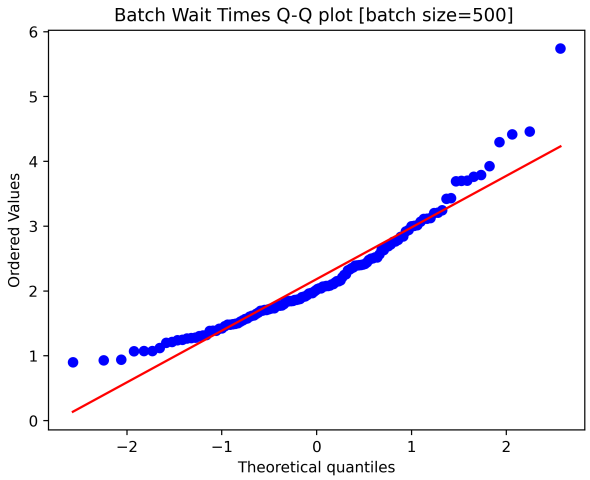


Figure A.15: Queue Length for a Batch of Size 500 ($\rho_0 = .75$)

Figure A.16: Queue Length for a Batch of Size 1000 ($\rho_0 = .75$)

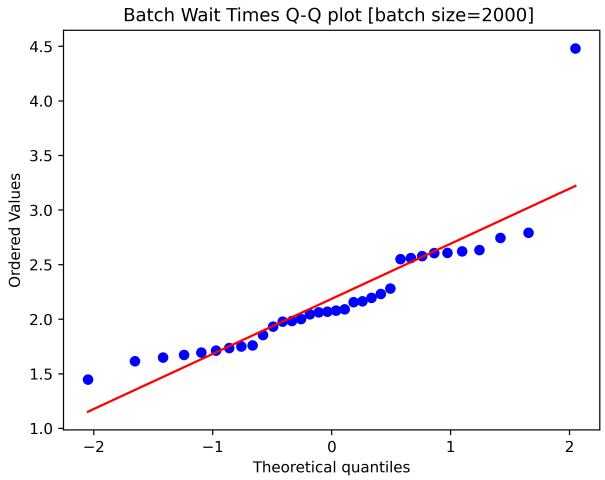


Figure A.17: Queue Length for a Batch of Size 1000 ($\rho_0 = -.75$)

A.2 Appendix B: m.g.f. waiting time derivation

$$E[e^{t\sum_{i=1}^{m} W_i}] \tag{A.1}$$

$$E[e^{tW_1}] = 1 - \rho + \rho\left(\frac{\gamma}{\gamma - t}\right) \qquad \text{for } t < \gamma\gamma = \mu - \lambda = \mu(1-\rho) = \lambda(1-\rho)/\rho$$

Using the Lindley recursion

$$W_{j+1} = (W_j + V_j - T_j)^+, \qquad j = 1, 2, \ldots \tag{A.2}$$

where $V_j \sim Exp(\mu)$ and $T_j \sim Exp(\lambda)$ for $j = 1, 2, \ldots$, are i.i.d.,the conditional m.g.f. of W_{j+1} is obtained as:

A.3 Appendix C: Change-point Detection (CPD) results

A few observations are in order. The detection delay gets smaller with longer batch sizes as can be seen in figures Figure A.19, Figure A.20 and Figure A.21.

For different changes in traffic intensity, we plotted the correct detection rate against the mean time to detection ARL_1. An increase of $\delta\rho$ represents an increase in the strength of the signal. As the signal gets stronger, the CDR increases, figures Figure A.19, Figure A.20 and Figure A.21. Small signals; however, can become hard to detect for larger batch sizes, causing a drop in the true positive rates, see figures Figure A.23a and Figure A.23b.

Larger batch sizes produce better performance both in terms of true positive rates. The growth has a limit; however, as past a certain batch size, no significant gain in accuracy can be gained, as can be observed when comparing the performance changes between a batch of size 500 and 1,000 on figures Figure A.18, Figure A.23a and Figure A.23b. This suggests the possibility to find the "best" batch size that achieves both a high precision and high recall for a given $\delta\rho$.

For larger batch size, ARL_1 monotically increases, see figures Figure A.19, Figure A.20 and Figure A.21. The main reason for that delay is due to the warm-up delay for GLRT. At least 20 samples need to be collected to detect a change. At a batch size of 500, no change could be detected till at least $T = 10,000$.

Except for $\delta\rho = 1.2$, there is a slight drop in CDR for all the tests between a batch of size 500 and a batch of size 1,000. The

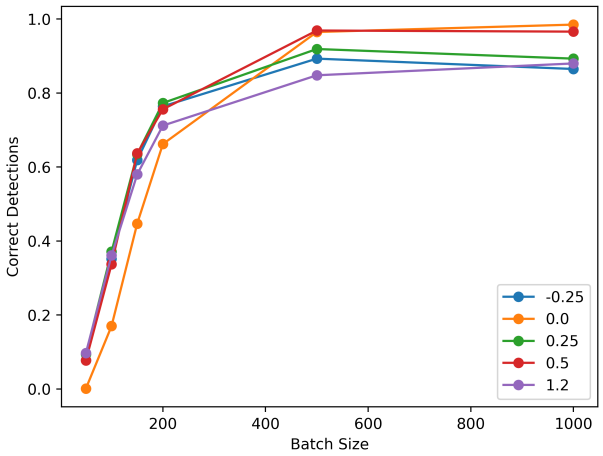
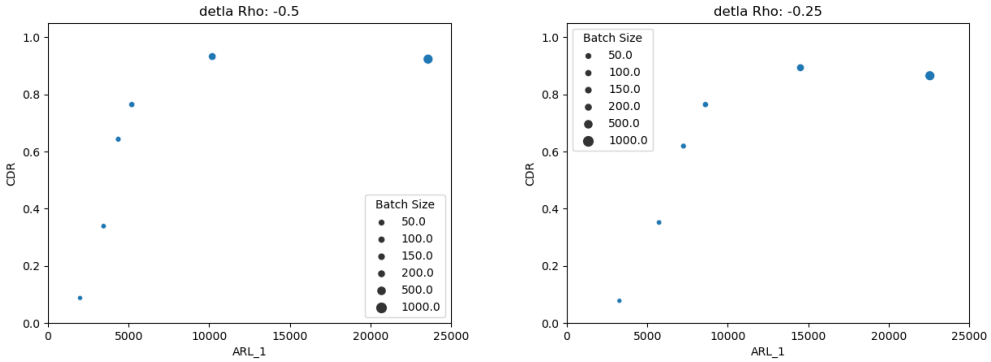


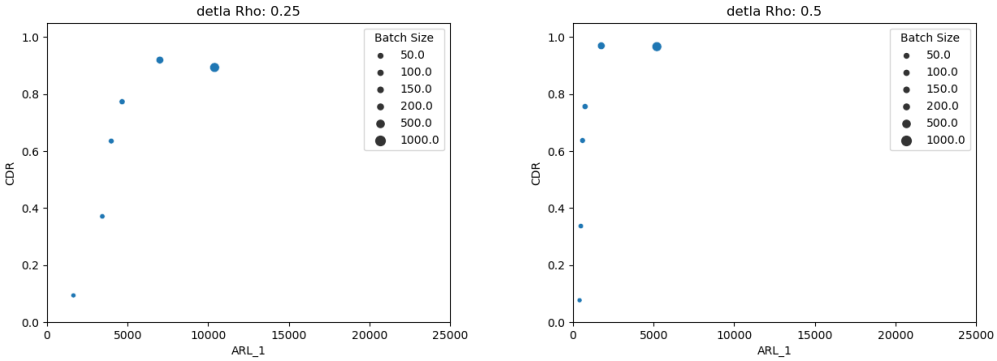
Figure A.18: CDR vs. Batch Size based on Waiting Times ($\rho = 0.5$)



(a) Delta Rho -0.50

(b) $\delta\rho = -0.25$

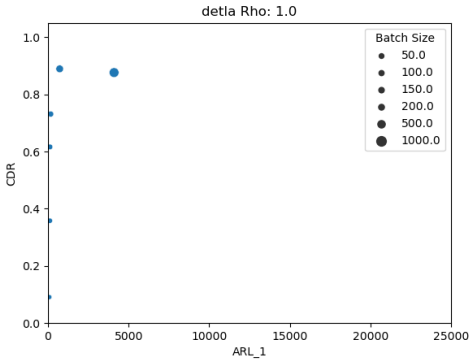
Figure A.19: Correct Detection vs. Detection Delay ($\rho = 0.5$) based-off W_q



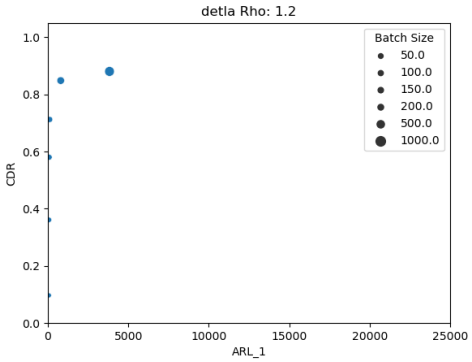
(a) $\delta\rho = 0.25$

(b) Delta Rho 0.50

Figure A.20: Correct Detection vs. Detection Delay ($\rho = 0.5$) based-off W_q

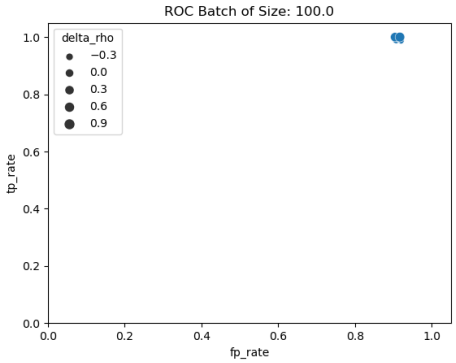


(a) Delta Rho 1.0

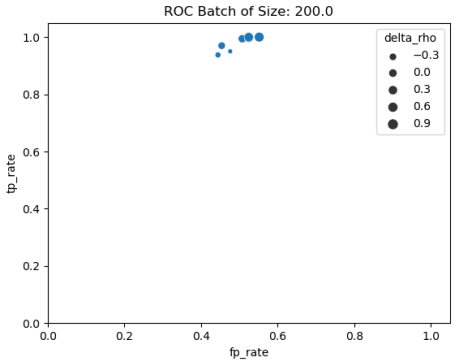


(b) Delta Rho 1.2

Figure A.21: Correct Detection vs. Detection Delay ($\rho = 0.5$) based-off W_q

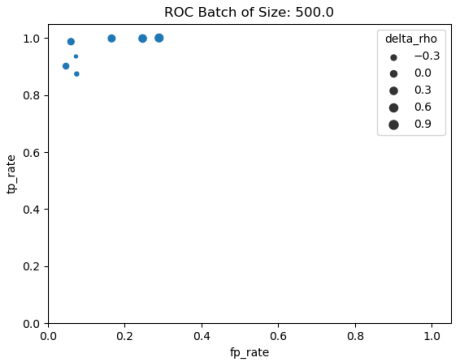


(a) Batch of size 100

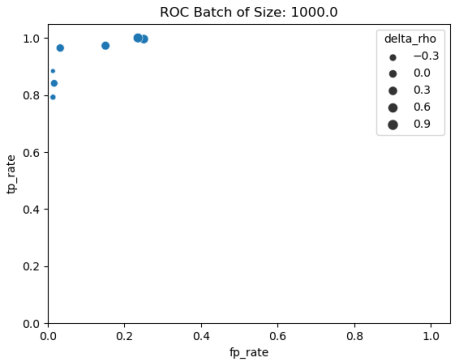


(b) Batch of size 200

Figure A.22: ROC



(a) Batch of size 500



(b) Batch of size 1000

Figure A.23: ROC

A.3.1 In queue Waiting Times Detection Results

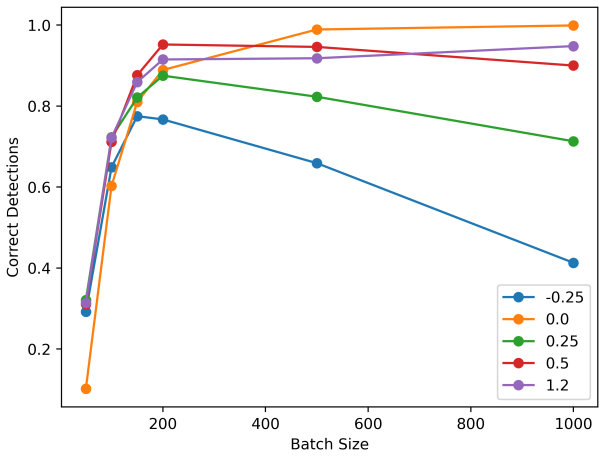
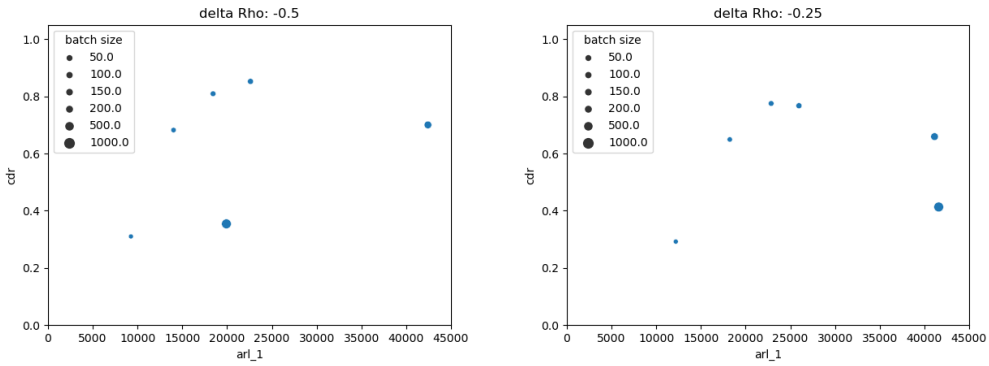
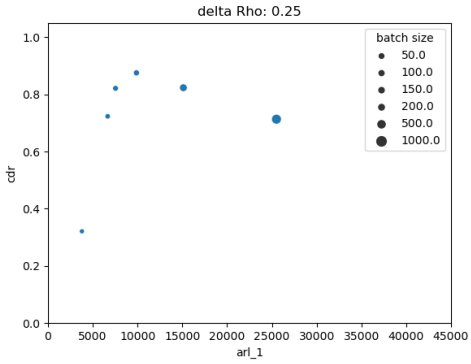


Figure A.24: CDR vs. Batch Size Based on Age of Process ($\rho = 0.5$)

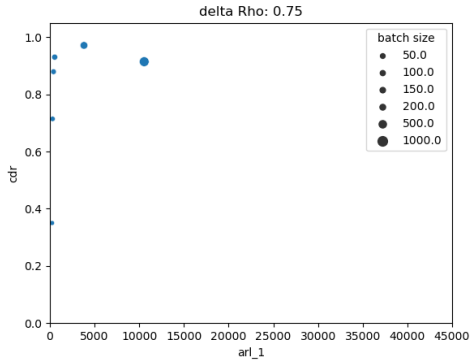


(a) $\delta\rho = -0.50$ (b) $\delta\rho = -0.25$

Figure A.25: Correct Detection vs. Detection Delay for age of a process (-0.5, -0.25)

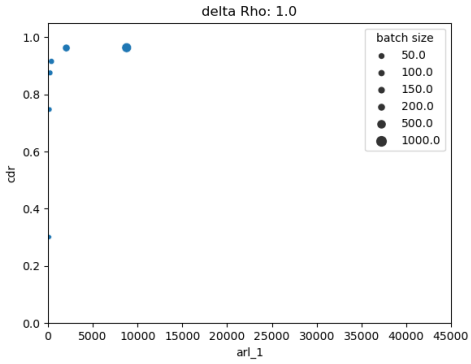


(a) $\delta\rho = 2.5$

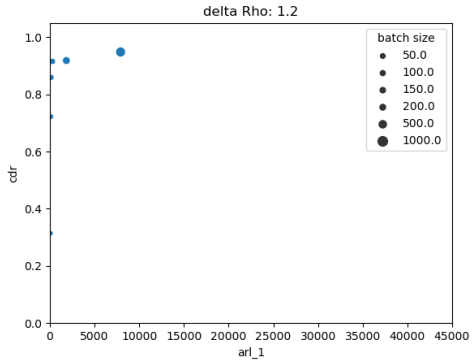


(b) $\delta\rho = 0.75$

Figure A.26: Correct Detection vs. Detection Delay for age of a process (0.25, 0.75)

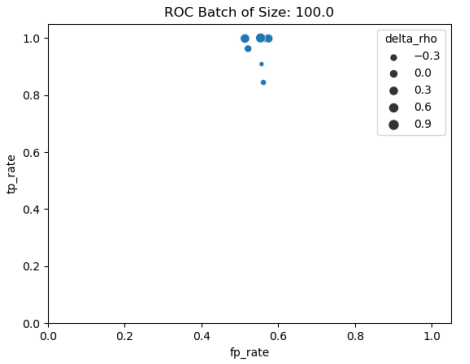


(a) $\delta\rho = 1.0$

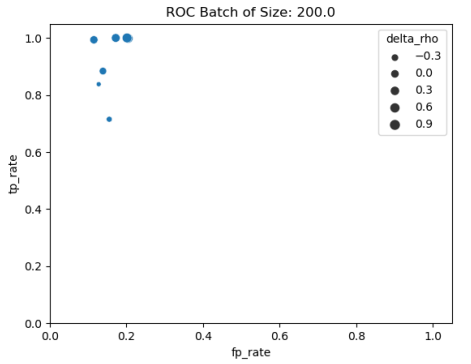


(b) $\delta\rho = 1.2$

Figure A.27: Correct Detection vs. Detection Delay for age of a process (1.0, 1.2)

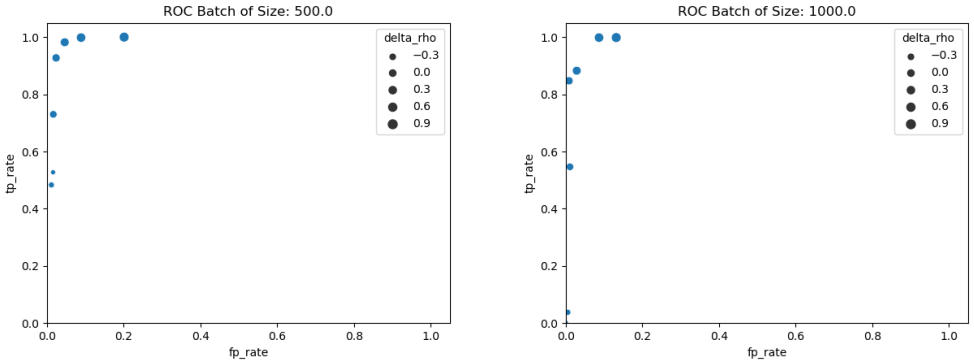


(a) Batch of size 100



(b) Batch of size 200

Figure A.28: ROC for the age of process



(a) Batch of size 500 (b) Batch of size 1000

Figure A.29: ROC for the age of process

A.3.2 Queue Length Detection Results

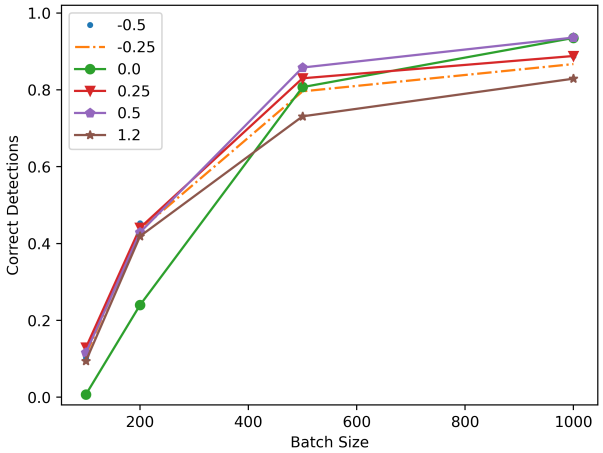
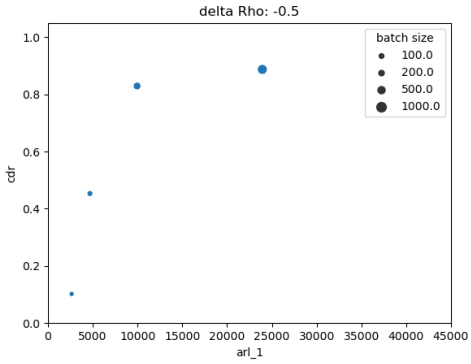
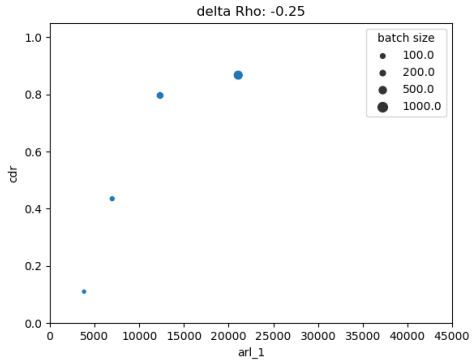


Figure A.30: CDR vs. Batch Size Based on the Queue Length ($\rho = 0.5$)

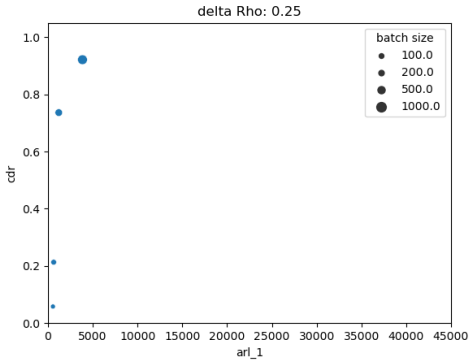


(a) $\delta\rho = -0.50$

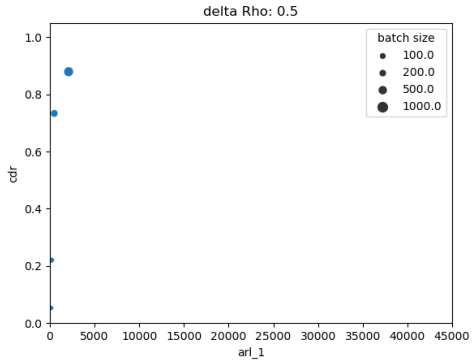


(b) $\delta\rho = -0.25$

Figure A.31: Correct Detection vs. Detection Delay for the Queue Length ($\rho = 0.5$)

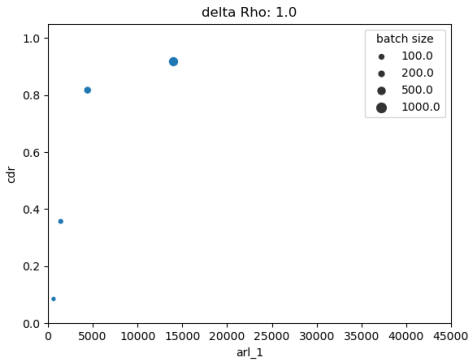


(a) $\delta\rho = 0.25$

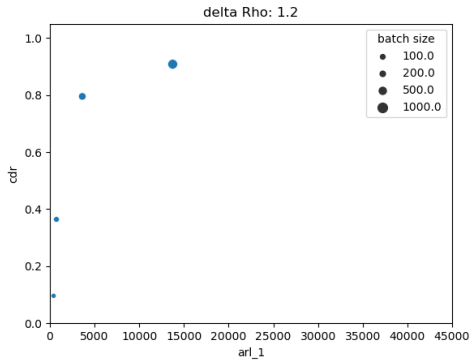


(b) $\delta\rho = 0.50$

Figure A.32: Correct Detection vs. Detection Delay for the Queue Length ($\rho = 0.75$)

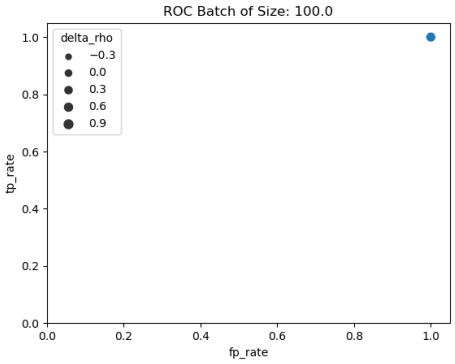


(a) $\delta\rho = 1.0$

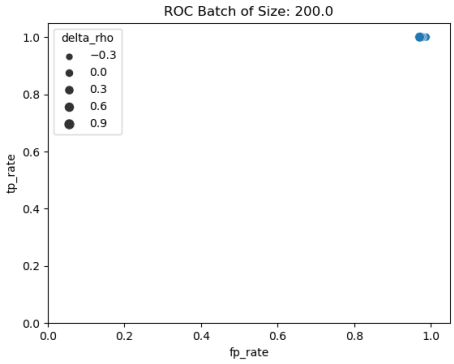


(b) $\delta\rho = 1.2$

Figure A.33: Correct Detection vs. Detection Delay for the Queue Length ($\rho = 0.5$)

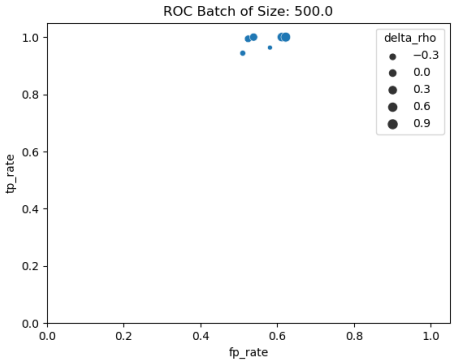


(a) Batch of size 100

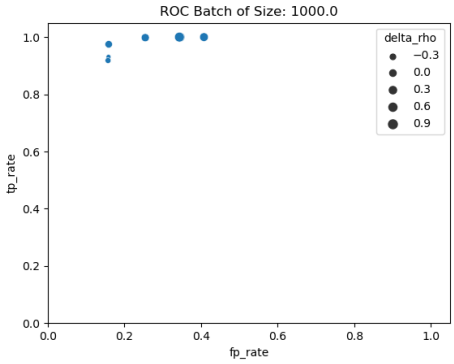


(b) Batch of size 200

Figure A.34: ROC for the Queue Length ($\rho = 0.75$)



(a) Batch of size 500



(b) Batch of size 1000

Figure A.35: ROC for the Queue Length ($\rho = 0.75$)

A.4 Parallel Detection Tests

Batch Size	rho	delta_rho	TP	TN	FN	FP	CDR	Missed Detection Prob	Precision	Recall	tp_rate	fp_rate	ARL_1
50	0.25	-0.5	0	20	0	0	1	0	0	0	0	0	
100	0.25	-0.25	0	20	0	0	1	0	0	0	0	0	
50	0.25	0	0	20	0	0	1	0	0	0	0	0	
50	0.25	0.25	3	7	2	8	0.5	0.727273	0.272727	0.6	0.6	0.533333	2378.197
50	0.25	0.5	9	6	0	5	0.75	0.357143	0.642857	1	1	0.454545	2577.784
50	0.25	0.75	12	4	2	2	0.8	0.142857	0.857143	0.857143	0.857143	0.333333	2826.927
50	0.25	1	11	7	1	1	0.9	0.083333	0.916667	0.916667	0.916667	0.125	3290.804
100	0.25	1.2	13	5	2	0	0.9	0	1	0.866667	0.866667	0	1852.518
50	0.5	-0.5	0	20	0	0	1	0	0	0	0	0	
200	0.5	-0.25	0	20	0	0	1	0	0	0	0	0	
50	0.5	0	0	20	0	0	1	0	0	0	0	0	
50	0.5	0.25	10	7	1	2	0.85	0.166667	0.833333	0.909091	0.909091	0.222222	1827.4
50	0.5	0.5	15	4	0	1	0.95	0.0625	0.9375	1	1	0.2	2401.817
100	0.5	0.75	12	8	0	0	1	0	1	1	1	0	2857.707
500	0.5	1	11	9	0	0	1	0	1	1	1	0	4009
500	0.5	1.2	13	7	0	0	1	0	1	1	1	0	3664.483
100	0.75	-0.5	0	20	0	0	1	0	0	0	0	0	
50	0.75	-0.25	0	20	0	0	1	0	0	0	0	0	
100	0.75	0	0	20	0	0	1	0	0	0	0	0	
300	0.75	0.25	13	6	0	1	0.95	0.071429	0.928571	1	1	0.142857	2708.859
50	0.75	0.5	11	9	0	0	1	0	1	1	1	0	4301.92
1000	0.75	0.75	15	5	0	0	1	0	1	1	1	0	4944.002
50	0.75	1	14	6	0	0	1	0	1	1	1	0	6128.572
50	0.75	1.2	16	4	0	0	1	0	1	1	1	0	6389.88

Table A.1: Results for a test on all 3 observations

m	rho	delta_rho	Correct_Detection	TP	TN	FP	FN	Precision	Recall	tp_rate	fp_rate	ARL_1
	0.25	-0.5	1	0	20	0	0	0	0	0	0	
50	0.25	-0.25	1	0	20	0	0	0	0	0	0	
	0.25	0	1	0	20	0	0	0	0	0	0	
50	0.25	0.25	0.3	0	6	0	14	0	0	0	0	
50	0.25	0.5	0.5	1	9	0	10	1	0.090909	0.090909	0	1287.459
50	0.25	0.75	0.55	4	7	0	9	1	0.307692	0.307692	0	5188.317
50	0.25	1	0.7	8	6	0	6	1	0.571429	0.571429	0	3475.093
200	0.25	1.2	0.85	10	7	0	3	1	0.769231	0.769231	0	2778.164
50	0.5	-0.5	1	0	20	0	0	0	0	0	0	
50	0.5	-0.25	1	0	20	0	0	0	0	0	0	
	0.5	0	1	0	20	0	0	0	0	0	0	
50	0.5	0.25	0.45	3	6	0	11	1	0.214286	0.214286	0	2988.766
300	0.5	0.5	0.7	6	8	0	6	1	0.5	0.5	0	2012.648
200	0.5	0.75	0.95	11	8	0	1	1	0.916667	0.916667	0	2507.866
500	0.5	1	0.95	11	8	0	1	1	0.916667	0.916667	0	2972.175
500	0.5	1.2	1	12	8	0	0	1	1	1	0	3918.172
	0.75	-0.5	1	0	20	0	0	0	0	0	0	
	0.75	-0.25	1	0	20	0	0	0	0	0	0	
50	0.75	0	1	0	20	0	0	0	0	0	0	
100	0.75	0.25	1	14	6	0	0	1	1	1	0	2376.718
500	0.75	0.5	1	13	7	0	0	1	1	1	0	3661.441
1000	0.75	0.75	0.95	13	6	0	1	1	0.928571	0.928571	0	5583.784
500	0.75	1	1	12	8	0	0	1	1	1	0	6814.269
100	0.75	1.2	1	11	9	0	0	1	1	1	0	3970.344

Table A.2: Results for a test based off age observations

Batch Size	rho	delta_rho	TP	TN	FP	FN	CDR	Precision	Recall	fp_rate	tp_rate	ARL_1
50	0.25	-0.5	0	20	0	0	1	0	0	0	0	
50	0.25	-0.25	0	20	0	0	1	0	0	0	0	
50	0.25	0	0	20	0	0	1	0	0	0	0	
50	0.25	0.25	3	8	0	9	0.55	1	0.25	0	0.25	3226.013
50	0.25	0.5	2	6	0	12	0.4	1	0.142857	0	0.142857	1540.747
100	0.25	0.75	9	9	0	2	0.9	1	0.818182	0	0.818182	2565.601
50	0.25	1	15	4	0	1	0.95	1	0.9375	0	0.9375	1669.389
50	0.25	1.2	15	4	0	1	0.95	1	0.9375	0	0.9375	1970.81
50	0.5	-0.5	0	20	0	0	1	0	0	0	0	
50	0.5	-0.25	0	20	0	0	1	0	0	0	0	
100	0.5	0	0	20	0	0	1	0	0	0	0	
50	0.5	0.25	9	10	0	1	0.95	1	0.9	0	0.9	3444.496
50	0.5	0.5	14	5	1	0	0.95	0.933333	1	0.166667	1	3630.902
300	0.5	0.75	13	7	0	0	1	1	1	0	1	2309.338
200	0.5	1	14	6	0	0	1	1	1	0	1	3312.504
50	0.5	1.2	13	7	0	0	1	1	1	0	1	3401.458
50	0.75	-0.5	0	20	0	0	1	0	0	0	0	
100	0.75	-0.25	0	20	0	0	1	0	0	0	0	
100	0.75	0	0	20	0	0	1	0	0	0	0	
50	0.75	0.25	13	7	0	0	1	1	1	0	1	2787.172
300	0.75	0.5	17	3	0	0	1	1	1	0	1	3653.426
500	0.75	0.75	14	6	0	0	1	1	1	0	1	4038.855
500	0.75	1	11	9	0	0	1	1	1	0	1	3141.415
300	0.75	1.2	13	7	0	0	1	1	1	0	1	2816.774

Table A.3: Results for a test based off queue observations

Batch Size	rho	delta_rho	TP	TN	FP	FN	CDR	Precision	Recall	fp_rate	tp_rate	ARL_1
50	0.25	-0.5	0	20	0	0	1	0	0	0	0	
100	0.25	-0.25	0	20	0	0	1	0	0	0	0	
50	0.25	0	0	20	0	0	1	0	0	0	0	
50	0.25	0.25	9	5	3	3	0.7	0.75	0.75	0.375	0.75	3193.508
50	0.25	0.5	9	8	2	1	0.85	0.818182	0.9	0.2	0.9	2813.45
50	0.25	0.75	8	9	2	1	0.85	0.8	0.888889	0.181818	0.888889	1349.277
100	0.25	1	19	1	0	0	1	1	1	0	1	2221.911
100	0.25	1.2	10	9	1	0	0.95	0.909091	1	0.1	1	1867.452
50	0.5	-0.5	0	20	0	0	1	0	0	0	0	
100	0.5	-0.25	0	20	0	0	1	0	0	0	0	
50	0.5	0	0	20	0	0	1	0	0	0	0	
200	0.5	0.25	7	12	1	0	0.95	0.875	1	0.076923	1	3319.809
50	0.5	0.5	13	7	0	0	1	1	1	0	1	1972.853
300	0.5	0.75	12	8	0	0	1	1	1	0	1	3119.401
300	0.5	1	12	5	3	0	0.85	0.8	1	0.375	1	2125.51
50	0.5	1.2	12	8	0	0	1	1	1	0	1	3012.321
50	0.75	-0.5	0	20	0	0	1	0	0	0	0	
100	0.75	-0.25	0	20	0	0	1	0	0	0	0	
100	0.75	0	0	20	0	0	1	0	0	0	0	
300	0.75	0.25	15	4	1	0	0.95	0.9375	1	0.2	1	2916.906
100	0.75	0.5	14	6	0	0	1	1	1	0	1	4820.327
500	0.75	0.75	14	5	1	0	0.95	0.933333	1	0.166667	1	4084.533
200	0.75	1	12	8	0	0	1	1	1	0	1	4154.059
100	0.75	1.2	10	8	2	0	0.9	0.833333	1	0.2	1	2121.593

Table A.4: Results for a test based off wait observations

A.5 Results of the Comparison Multiple Observation Test

From the receiver operating curve Figure A.36, one sees that the multiple observations provides a high

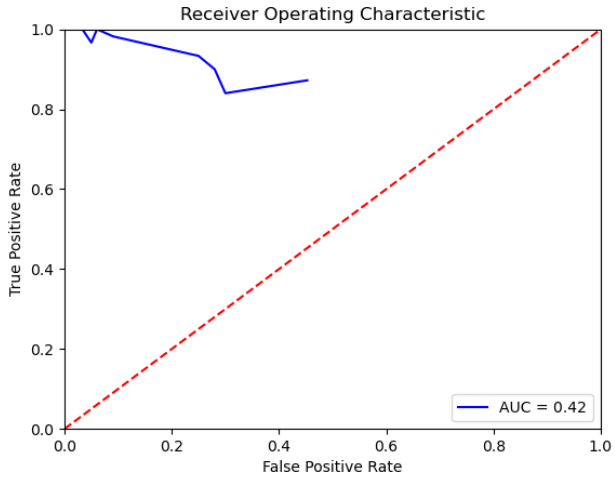


Figure A.36: Receiver Operating Curve

A lift chart relates the number of correct detections to the total rate of detections. It is desired that the curve be concave up, above the bisector, to represent the fact that the test is accurate not simply accepting all the changes.

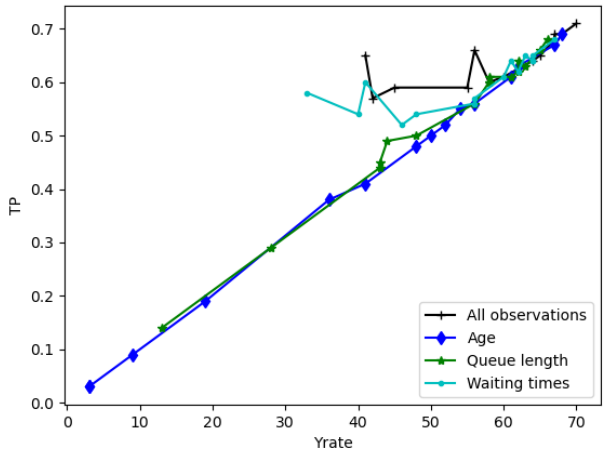


Figure A.37: Lift Chart

REFERENCES

[1] L. Kleinrock, *Queueing Systems*. Wiley Interscience, 1975, vol. I: Theory, (Published in Russian, 1979. Published in Japanese, 1979. Published in Hungarian, 1979. Published in Italian 1992.)

[2] S. M. Ross, *Introduction to Probability Models (Twelfth Edition)*, S. M. Ross, Ed. Academic Press, 2019, pp. 193–291, ISBN: 978-0-12-814346-9.

[3] ——, *Stochastic Processes (Second Edition)*, S. M. Ross, Ed. Wiley, 1995, ISBN: 978-0471120629.

[4] R. G. Gallager, *Stochastic Processes: Theory for Applications*. Cambridge University Press, 2013.

[5] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*, 1st. USA: Cambridge University Press, 2013, ISBN: 1107027500.

[6] BTS, "On time performance - flight delays at a glance," Bureau of Transportation Statistics, Report, Aug. 2019.

[7] M. Ball, C. Barnhart, M. Dresner, M. Hansen, K. Neels, A. Odoni, E. Peterson, L. Sherry, A. Trani, and B. Zou, *Total Delay Impact Study*, Oct. 2010.

[8] R. De Neufville and A. R. Odoni, *Airport systems : planning design, and management*. New York: McGraw-Hill, 2003.

[9] I. E. Manataki and K. G. Zografos, "A generic system dynamics based tool for airport terminal performance analysis," *Transportation Research Part C: Emerging Technologies*, vol. 17, pp. 428–443, 2009.

[10] A. Marzuoli, E. Boidot, E. Feron, P. B. C. Van Erp, A. Ucko, A. Bayen, and M. Hansen, "Multimodal impact analysis of an airside catastrophic event: A case study of the asiana crash," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 587–604, 2016.

[11] A. Marzuoli, P. Monmousseau, and E. Feron, "Passenger-centric metrics for air transportation leveraging mobile phone and twitter data," IEEE.

[12] P. Monmousseau, A. Marzuoli, C. Bosson, E. Feron, and D. Delahaye, "Doorway to the united states: An exploration of customs and border protection data," IEEE.

[13] *Sydney kingsford smith international airport (syd)*, https://www.world-airport-codes.com/australia/kingsford-smith-7232.html, Accessed: 2016-11-01.

[14] T. The Bureau of Infrastructure and R. E. (BITRE), "Airport traffic data," Australian Government, Department of Infrastructure, Regional Development and Cities, Canbeira, Australia, Airport Traffic Statisical Report, Aug. 2018, pp. 29–32.

[15] L. G. Birta and G. Arbez, *Modelling and simulation exploring dynamic system behaviour*, 2nd ed., ser. Simulation Foundations, Methods and Applications. London: Springer, 2007, vol. XVIII, p. 437, ISBN: 9781846286216.

[16] J. J. Fruin, *Pedestrian Planning and Design.* Metropolitan Association of Urban Designers and Environmental Planners, 1971.

[17] S. B. Young, "Evaluation of pedestrian walking speeds in airport terminal," *Transportation Research Record*, vol. 99,

[18] A. M. Law and W. D. Kelton, *Simulation Modeling & Analysis*, 3rd. New York: McGraw-Hill, Inc, 2000.

[19] J. Q. Hale, "An investigation of model-based approaches in solving a variety of global optimization problems," Ph.D. dissertation, Georgia Institute of Technology, 2017.

[20] J. Hu, M. C. Fu, and S. I. Marcus, "Stochastic optimization using model reference adaptive search," in *Winter Simulation Conference*, M. E. Kuhl, Ed., pp. 811–818.

[21] ——, "A model reference adaptive search method for global optimization," *Operations Research*, vol. 55, no. 3, pp. 549–568, 2007.

[22] A. Forsgren, P. E. Gill, and M. H. Wright, "Interior methods for nonlinear optimization," *SIAM Review*, vol. 44, no. 4, pp. 525–597, 2002.

[23] M. C. Cario and B. L. Nelson, "Autoregressive to anything: Time-series input processes for simulation," *Operations Research Letters*, vol. 19, no. 2, pp. 51–58, 1996.

[24] M. C. Cario, "Modeling and simulating time series input processes with artafacts and artagen," in *Simulation Conference, 1996. Proceedings. Winter*, IEEE, pp. 207–213, ISBN: 0780333837.

[25] A. S. Alfa, *Queueing Theory for Telecommunications: Discrete Time Modelling of a Single Node System*. Springer Publishing Company, Incorporated, 2010, p. 247, ISBN: 1441973133, 9781441973139.

[26] D. M. Hawkins and K. D. Zamba, "Statistical process control for shifts in mean or variance using a changepoint formulation," *Technometrics*, vol. 47, no. 2, pp. 164–173, 2005.

[27] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[28] ——, "An improvement to wald's approximation for some properties of sequential tests," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 16, no. 1, pp. 136–139, 1954.

[29] P. Granjon, "The cusum algorithm a small review," [Online; accessed 14-February-2020], 2013.

[30] L. Tze Leung, "Information bounds and quick detection of parameter changes in stochastic systems," *IEEE Transactions on Information Theory*, vol. 44, no. 7, pp. 2917–2929, 1998.

[31] Y. Mei, "Sequential change-point detection when unknown parameters are present in the pre-change distribution," *The Annals of Statistics*, vol. 34, no. 1, pp. 92–122, 2006.

[32] Y. Cao, L. Xie, Y. Xie, and H. Xu, "Sequential change-point detection via online convex optimization," *Entropy*, vol. 20, no. 2, p. 108, 2018.

[33] G. J. Ross and N. M. Adams, "Two nonparametric control charts for detecting arbitrary distribution changes," *Journal of Quality Technology*, vol. 44, no. 2, pp. 102–116, 2012.

[34] G. J. Ross, "Sequential change detection in the presence of unknown parameters," *Statistics and Computing*, vol. 24, no. 6, pp. 1017–1030, 2014.

[35] ——, "Parametric and nonparametric sequential change detection in R: The cpm package," *Journal of Statistical Software*, vol. 66, no. 3, pp. 1–20, 2015.

[36] ——, "Sequential change detection in the presence of unknown parameters," *Statistics and Computing*, vol. 24, no. 6, pp. 1017–1030, 2014.

[37] C. Alexopoulos and A. F. Seila, "Output analysis: Output analysis for simulations," in *Proceedings of the 32nd Conference on Winter Simulation*, ser. WSC '00, Orlando, Florida: Society for Computer Simulation International, 2000, pp. 101–108, ISBN: 0780365828.

[38] T. Aktaran-Kalaycı, C. Alexopoulos, N. T. Argon, D. Goldsman, and J. R. Wilson, "Exact expected values of variance estimators for simulation," *Naval Research Logistics*, vol. 54, no. 4, pp. 397–410, 2007.

[39] N. M. Steiger and J. R. Wilson, "Convergence properties of the batch means method for simulation output analysis," *INFORMS Journal on Computing*, vol. 13, no. 4, pp. 277–293, 2001.

[40] D. J. Daley, "The serial correlation coefficients of waiting times in a stationary single server queue," vol. 8, no. 04, p. 683, 1968.

[41] P. M. Morse, "Stochastic properties of waiting lines," *Journal of the Operations Research Society of America*, vol. 3, no. 3, pp. 255–261, 1955.

[42] N. U. Prabhu, *Stochastic storage processes : queues, insurance risk, and dams*. New York: New York : Springer-Verlag, 1980.

VITA

U.S. and French dual Citizen

Professional Expertise

- Air Traffic Management
  - Automation of decision making processes in Air Traffic Management
  - Traffic Flow Management for congestion mitigation in the National Airspace
- Operations Research
  - Formulation of mathematical programs for multiple applications including path planning, linear regression, scheduling and network flow applications
  - Design, implementation and analysis of discrete-event simulations
  - Regression and auto-regression analysis
  - Design and Development of Discrete-event Simulations
  - Simulation optimization
- Communication
  - Presentation and teaching of technical/non-technical subjects to a wide audience
- Leadership
  - Communication of clear goals and guidelines to effectively coordinate team actions
  - Mentoring by providing technical support and access to network and resources to lab members

Professional Experience

- **Warner Media - Ad Monetization Group**–Atlanta, GA May 2020-Present

  Operations Research Analyst

  - Inventory Management

    Spearheaded the modernization of scheduling methods for the scheduling of advertiser's spots on linear TV using reinforcement learning.

  - Production Code Management

    Debugged, QA tested and modified mixed integer linear programming models in production while consulting with stakeholders to find the right design.

  - Assessed the added value of new data sources in the forecasting capabilities of the team.

- **Georgia Institute of Technology: Air Transportation Lab**–Atlanta, GA August 2012-Present

  Graduate Research Assistant for Dr. John-Paul Clarke

  - Thesis: *Model-based Approach to the Utilization of Heterogeneous Data in the Optimization of Passenger Airport Terminal Systems.*
  - Research Purpose: The research applies stochastic modeling, simulation and optimization to design efficient procedures for the transport of passengers inside airports.
  - Performed a study of passenger flow traffic at Sydney International Airport using Python and one year of flights and cellphone data.
  - Developed a method to improve the estimation of arrival and departure rates parameters for queueing simulation.
  - Updated **Perl** and **C++** scripts for the MIT Extensible Network Simulation (MEANS) software to make it useful for a major airline.
  - Coordinated the work of four graduate students to implement path planning heuristics for the aerial surveillance of areas spanning more than 100 km^2.
  - Airbus Aerial:
    - Developed and implemented a mixed integer linear program for a swarm of Unmanned Aircraft Systems (UAS) to explore different locations simultaneously and generate an optimum path for a minimum number of UAS under time and energy constraints
    - Led a team of three new graduate students through the software development process, and coordinated the research efforts.
  - Société Internationale de Télécommunications Aéronautiques (SITA):
    - Performed a longitudinal study of passenger flows at Sydney International Airport using Python and SQL to process flights and MAC addresses datasets
    - Created a discrete-event simulation of foot traffic in C++ for the Sydney International Airport, which was integrated in a model-predictive tool
    - Currently use the aforementioned simulation to optimize input by selecting minimum risk sample

- **FlightSafety International**–Atlanta,GA May 2015-July 2016

  Part-time Engineer I:

  - Reviewed and improved upon the state-of-the-art Simulated ATC Environment (SATCE) in C++. To achieve this end, connected the simulation to actual data and formulated a mathematical program to prevent collisions within the simulation.

- Selected by the executive leadership to create and develop a novel Flight Simulator Concept within one year
- Led the design and development of a new reduced order turboprop engine model, simulation as well as the associated controller
- Demonstrated significant cost savings of the initial design while providing 80% of the original model's capabilities

Education

- **Georgia Institute of Technology**−Atlanta, GA December 2021
  Ph.D. in Aerospace Engineering
  - Dissertation: *Model-based Approach to the Utilization of Heterogeneous Data in the Optimization of Passenger Airport Terminal Systems.*

- **Georgia Institute of Technology**−Atlanta, GA Graduation August 2019
  M.S. in Operations Research
  - GPA: 3.49/4.0

- **Georgia Institute of Technology**−Atlanta, GA Graduated August 2013
  M.S. in Aerospace Engineering with a specialization in Flight Mechanics and Control
  - GPA: 3.91/4.0
  - Awarded the GEM National Fellowship

- **Georgia Institute of Technology**−Atlanta, GA Graduated December 2011
  B.S. in Aerospace Engineering
  - GPA: 3.57/4.0
  - Graduated with highest honors

Publications

- **Preprints**

  H. Nikoue, A. Marzuoli, J.P. Clarke, E. Feron, J.Peters. *Passenger Flow Predictions at Sydney International Airport: A Data-Driven Queuing Approach.* Aug. 20, 2015, arXiv:1508.04839

Leadership Experience

- **Georgia Tech Black Graduate Student Association**−Atlanta, GA January 2019 - May 2019
  Membership chair

- Planned and organized bi-weekly general body meetings
- Invited academic and non-academic presenters to give talks on issues pertinent to graduate students
- Presided over general body meetings

- **Georgia Tech Salsa Club**–Atlanta, GA January 2017 - April 2018
  Instructor and social chair
  - Instructed more than 60 students on fundamentals salsa dancing techniques and steps
  - Organized outings, and raised member participation to outings from 15% to 25%

Skills

- **Programming Languages and Environments**
  - C\C++, Python (intermediate), Linux, Vim , LaTeX
- **Prototyping Languages**
  - R (basic), MATLAB (proficient), SQL (MySQL, MariaDB), Mathematica, Gurobi
- **Other Softwares and Computing Tools**
  - AutoCAD, Autodesk Inventor, Solidworks, GIT, CMake, vim, Simpy
- **Languages**
  - English (fluent), French (native proficiency)

https://www.linkedin.com/in/harold-nikoue/