EXPLORING AIRPLANE BOARDING

EXPLORING AIRPLANE BOARDING: MODEL BASED APPROACH

By MYLES MARIN, B.Sc.

A Thesis Submitted to the School of Graduate Studies in Partial Fulfilment of the Requirements for the Degree Master of Science

McMaster University MASTER OF SCIENCE (2018) Hamilton, Ontario (Mathematics)

TITLE: Exploring Airplane Boarding: Model Based Approach AUTHOR: Myles Marin, B.Sc. (McMaster University) SUPERVISOR: Dr. M. Lovric NUMBER OF PAGES: x, 91

LAY ABSTRACT

The airline industry is crucial to economic growth and the number of passengers is increasing every year, reaching increases of 250 million. In light of this, the International Air Transport Association believes it is imperative to increase airline efficiency. Since airplanes generate revenue while they are in the air, we look to minimize the time they are on the tarmac, also known as airplane turnaround time. Passenger boarding has shown to constitute 60% of airplane turnaround time, identifying this step as a bottleneck. In this study, we investigate passenger boarding strategies utilized by airline companies and documented by existing literature. We then propose more efficient passenger boarding methods for the Airbus A320/ Boeing 737 and Airbus A380-800/ Boeing 777-300ER airline classes. With the Airbus A380-800/ Boeing 777-300ER having limited documentation, we define and propose new boarding strategies that improve the passenger boarding time.

ABSTRACT

The airline industry is crucial to economic growth and the number of passengers is expected to increase by approximately 250 million from 2017 to 2018. IATA believes it is imperative to increase airline efficiency to maintain sustainability. Passenger boarding is known to constitute 60% of airplane turnaround time, identifying this step as rate determining. In this study, we investigate passenger boarding strategies utilized by airline companies and strategies simulated by existing literature. We propose more efficient passenger boarding methods for the Airbus A320/ Boeing 737 and Airbus A380-800/ Boeing 777-300ER airline classes. We use GNU Octave to construct a cellular automaton model that operates on the interaction of a $2-$dimensional matrix M_{person}, representing passenger attributes, and a $4-$dimensional matrix M_{steps}, representing the cells of the airplane in consideration. We consider row interferences, aisle interferences, luggage stow away time, and general passenger delays. We find that boarding strategies which minimize passenger interferences are the most efficient for both airline classes. Interestingly, some boarding strategies for the Airbus A320/ Boeing 737 are more efficient when boarding at 30 passengers/ min instead of 60 passengers/ min. However, the Airbus A380-800/ Boeing 777-300ER shows slower boarding rates for any boarding frequency less than 60 passengers/ min, suggesting the size of the airplane layout determines sensitivity to boarding frequency. With the Airbus A380-800/ Boeing 777-300ER having limited documentation, we define and propose new boarding strategies, OBFOIZ and OBFOISA, that improve on the passenger boarding times in the existing literature for this double-aisled airplane.

ACKNOWLEDGEMENTS

I would first like to thank my thesis supervisor Dr. Miroslav Lovric of the Mathematics & Statistics Department at McMaster University for his advice and direction throughout this research. Prof. Lovric was open to new ideas and always had an open door for discussion. A large thank you to Prof. Lovric for providing the appropriate guidance when needed and unparalleled support.

I would also like to acknowledge Dr. Matt Valeriote and Dr. Adam Van Tuyl of the Mathematics & Statistics Department at McMaster University for being secondary readers of this thesis, I am grateful for their support and feedback.

Finally, I express my dearest thank you and appreciation to my parents and fiancé for providing utmost support and continuous encouragement throughout my years of study and through the process of writing this thesis. This accomplishment would not have been possible without them. Thank you.

Author

Myles Marin

DECLARATION OF ACADEMIC ACHIEVEMENT

I, Myles Marin, declare this thesis to be my own work. I am the sole author of this document. No part of this work has been published or submitted for publication or for a higher degree at another institution.

To the best of my knowledge, the content of this document does not infringe on anyone's copyright.

My supervisor, Dr. Miroslav Lovric, and the members of my supervisory committee, Dr. Matt Valeriote and Dr. Adam Van Tuyl, have provided guidance and support at all stages of this project. I completed all of the research work.

Contents

List of Figures

1 Introduction

The International Air Transport Association (IATA) describes the airline industry as "The Business of Freedom", in which the global movement of goods and people is a positive force in our world. The aviation industry is massive in its own right, but its input in global economic growth is crucial, providing greater access to resources and international capital markets [1,8,9].

In 2016, the IATA announced that airline profitability reached \$35.6 billion, marking the highest recorded net profit margin of 5.1% in airline history [8]. With oil prices and operating costs on the rise, the IATA forecasted a slight dip in 2017 revenue and profit margins, equating to \$29.8 billion in net profit. The IATA saw improvement on the net profit prediction in 2017 with airline profitability reaching \$35.4 billion [8]. The IATA asserts that the airline industry is standing on solid ground with sustainable levels of profitability being charted. However, the IATA predicts that strong demand and airline efficiency constitute significant factors in determining net profitability in 2018. Moreover, the IATA predicts 2018 will see an increase in passengers to 4.3 billion, a 6.0% increase from 2017. In light of these details, it is justifiable for research dedicated to improving the efficiency of the airline industry [1,2,3,4,5].

Inherent to the term "airline industry", airplanes are only generating revenue when they are in the air since a greater volume of aircraft in the air over a given period results in larger economic income [1,5,6,7]. As a result, the airline industry has been subject to numerous studies to help improve airline efficiency, profitability, and customer satisfaction while the airlines are on the ground. **Airplane turnaround time** defines the time for which an airplane is on the ground. Specifically, turnaround time commences when the airplane is "chocked" on the tarmac and prepares for the next flight and airplane turnaround time ends when the chocks are removed and the airplane begins its journey to the runway (see Figure 1) [5,6]. Aircraft preparation is detailed and several steps are involved. During turnaround time, an airplane must undergo passenger and luggage removal, cabin cleaning, catering re-establishment, fuel tanks refuel, and luggage and passenger embarkment. While some of these procedures can happen simultaneously, passenger embarkment is the last step. Since safety is an airline company's primary concern, this step tends to be initiated once cabin cleaning and fuelling are complete [6,8].

Although the turnaround process has been streamlined over countless flights, statistical results expose passenger boarding as the most time consuming component, constituting 60% of the total turnaround time [1,6,8]. In 1998, Boeing noted that passenger boarding times slowed by more than 50% since 1970, and passenger boarding was the the most time consuming step in airplane turnaround time. Remarkably, this trend remains with variable strategies still employed by airline companies [1,6,8,9].

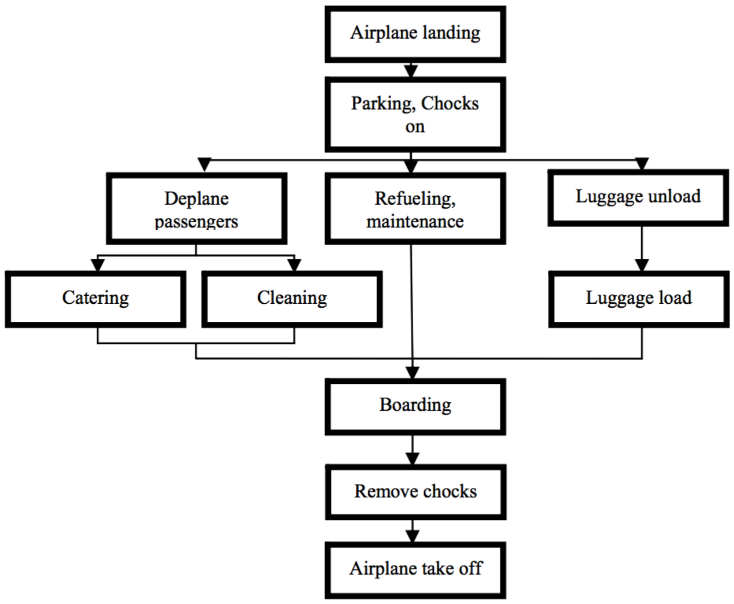


Figure 1: Airplane Turnaround Time diagram, adopted from Albert Steiner and Michel Philipp [5,6].

1.1 Factors in Passenger Boarding

To understand how the passenger boarding process can be improved, we need to understand the obstacles that passengers face when boarding an airplane, ultimately delaying them from reaching their seats. The most obvious obstacle is passenger-passenger interference, in which passengers prevent other passengers from reaching their desired seats. Therefore, the minimization of boarding time is directly related to minimizing the number of passenger-passenger interferences [1,6,8,9]. Giitsidis & Sirakoulis 2016 categorize every passenger interference as either an aisle interference or seat interference and, for purposes of constructing a simulation, these two interferences are almost exclusive. An **aisle interference** can be described as the prevention of boarding along the aisle the passenger is situated. For example, a passenger further along the aisle is waiting for a seat to be emptied or is stowing his/her luggage, causing all passengers in the aisle to wait for this action to be complete. A **seat interference** is when the passenger has found the row where his/her seat is located and discovers the seat, or seats, between the passenger's seat and the aisle is occupied by one or more passengers [1,5,7,8,10].

From these definitions, we can see how seat interferences may cause aisle interferences. Let's construct a scenario on a single-aisle plane with two seats on each side of the aisle. Passengers A, B, and C board the plane in respective order. Passenger A sits in the aisle seat of row n. Passenger B is designated to the window seat of row n and must wait for Passenger A to leave their seat, allowing Passenger B to sit. While this is occurring, Passenger C must be seated in row m such that $m > n$, causing Passenger C to wait behind passenger B. Therefore, the seat interference between Passenger A and B is causing an aisle interference between Passenger B and C. Extending this further, the aisle interference between Passenger B and C will most likely cause a chain of aisle interferences behind Passenger C to the entrance of the airplane.

Passenger-passenger interference forms the basis for understanding passenger boarding and constructing computer-based simulations. To have realistic parameters implemented in a model, we must consider the factors involved in initiating passenger-passenger interferences. Some of these include the following:

a) frequency of passengers arriving at the airplane entrance

b) walking speed inside the airplane, possibly dependent on sex, height, age, and number of family members

c) luggage stow away time dependent on similar factors to part b)

d) passenger walks past his/her row, having to retrace steps to locate their seats

The latter is a worst case scenario and greatly increases the time needed to board the plane [8].

1.2 Objectives

The flow of passengers in a single-aisled plane is well documented [7,8,9,10,11]. These studies confirm that the majority of airline companies will benefit from changing their boarding strategies to the more efficient methods outlined in the literature. Giitsidis & Sirakoulis 2016 investigated the most efficient way of boarding a single-aisled airplane, seating 150 passengers, and the most efficient way of boarding a double-aisled airplane, seating 540 passengers. As noted in their paper, the single-aisled airplane layout and passenger volume resembles the popular Airbus A320/ Boeing 737 and the larger airplane layout resembles the Airbus A380-800 and Boeing 777-300ER [8].

While simulations of single-aisled airplanes are well documented, the results for the double-aisled airplane mark new territory and are not well supported. In light of this, the aim of this study is to construct a homegrown computer-based model that verifies and improves on the boarding strategies for the Airbus A320/ Boeing 737 and Airbus A380-800/ Boeing 777-300ER, both of which were put forward by Giitsidis & Sirakoulis 2016. Specifically, we look to implement efficient boarding methods from single-aisled airplane studies in the larger airplane and develop original, efficient boarding methods that improve on the methods implemented by Giitsidis & Sirakoulis 2016. As Giitsidis & Sirakoulis 2016 have initiated passenger boarding simulations on a double-aisled airplane, without existing literature to support, this paper will make reference to Giitsidis & Sirakoulis 2016 throughout to expand on the work they accomplished.

Giitsidis & Sirakoulis 2016 enrich their model by altering the passenger boarding method, frequency, the walking speed of passengers, and considering passengers' sex, age, and height. Furthermore, they introduce the possibility of passengers to mistakenly walk past their seat. Some of these extensions will also be included into the homegrown model constructed in this study to, more accurately, validate the results put forward by Giitsidis & Sirakoulis 2016 and existing documentation [7,8,9,10,11].

2 Passenger Dynamics Models

A variety of methods have been adopted to reveal the dynamics involved in crowds and clusters of people, including fluid-dynamic, cellular automata, and agent-based methods [8,12,13].

2.1 Fluid Dynamic Models

To capture fluid crowd dynamics, continuum models for pedestrian dynamics are well-suited since individuals are treated as particles which are subject to forces, sometimes viewed as *social forces* [8,13,14,15]. Helbing *et al.* 2000 used a generalized, fluid-dynamic force model, focusing on the uncoordinated phenomenon of escape panic in crowds [12]. While this is well suited for representing panic behaviour, this type of model is less suitable for capturing the structured nature of airplane boarding, where passengers are obeying single file, unidirectional flow along an aisle before traversing to their seats.

2.2 Cellular Automata & Agent-based Modelling

Moving away from particle-based models simulating fluid crowd dynamics, models using cellular automata (CA) provide discrete spatial representation and time steps. The CA landscape consists of a lattice of cells, each representing a unit of the entire lattice. Each cell is given a "state" resembling the status of occupancy. As the model moves through discrete time steps, each cell in the lattice expresses an evolution of statuses. To achieve this, transition rules must be defined for each cell to express the alteration of statuses. CA transition rules are based on the concept of a **cell neighbourhood**, the set of cells which are defined to be neighbours of the central cell. The evolution of a cell will depend on its status and the status of each neighbouring cell [8,13,16,17]. Likewise, the neighbouring cell of a central cell will likely have the central cell as its neighbour.

John Conway's Game of Life is a traditional example of a cellular automaton simulating the basic principals of survival. Each cell can either be *dead* or *alive*, where the number of "living" neighbouring cells dictates the status of the central cell at the next time step. Conway's Game of Life will start with an initial set of cells that are alive, and then the lattice is given a transition function, determining the status of each cell at the next time step [13,16,17]. A typical transition function, with biological interpretation, for Conway's Game of Life can be described in the following way:

a) If a cell has less than 3 alive neighbours, it will be dead in the next time step due to lack of neighbours and support.

b) If a cell has more than 5 alive neighbours, it will be dead in the next time step due to overcrowding and overconsumption of resources.

The balance between not enough neighbours (a neighbour being a cell that “touches” the centralized cell) and too many neighbours determines the state of the central cell at the next time step. Generally, CA models consider individuals as homogeneous entities and individuals in the model share equal attributes. In some extensions of CA models, moving towards agent-based models, individuals are treated heterogeneously. In these models, we can find individuals with variable attributes, such as age, gender, height, mobility, etc. [7,8,13,17].

While agent-based modelling is based on the basic principals of CA, heterogenous agents yield different types of agent-to-agent interaction, bringing the perception of agents by other agents into the model. In agent-based modelling, each agent operates under its own profile and the underlying inner loop, in the simulation, proceeds agent by agent. This is unlike CA-based models, where the lattice is a uniform dissection of space and the simulation inner loop proceeds cell by cell, applying the same transition rule to all cells. With this in mind, agent-based modelling ideas can be captured with CA by extending the list of parameters for the transition rules to act on, or by extending the idea of a neighbourhood to greater regions [13,17].

2.2.1 Motivation for using CA

CA move to the forefront of available methods in which to conduct this study given their ability to sufficiently showcase collective group dynamics through localized interaction while executing on ordinary computers, capable of simulating their dynamics with great accuracy and reasonably low run time [8]. The added benefit of using a CA model rests in the number of previous studies available to validate the results and assumptions of the homegrown CA-based model constructed in this study [14,15,17,18,19,20]. In light of this, we seek to validate and extend the recent work of Giitsidis & Sirakoulis 2016 using a homegrown CA-based model.

2.2.2 A Formal Definition of CA

A cellular automaton is a discrete time model of a physical system that places emphasis on localized, individual-to-individual interaction. In collaboration with Stanislaw Ulam in 1952, John von Neumann constructed the basic ideas of CA. Von Neumann saw that historical problems in science focused on energy, power, force, and motion. Von Neumann believed that future problems in science would be more concerned with control, programming, information processing, communication, organization, and systems. Von Neumann’s original idea was to assemble a linkage of “mechanical devices” capable of reproducing themselves. Ulam suggested Von Neumann consider a grid of moving agents operating under specified sets of rules. With simultaneous development of the modern digital computer, it became clear that CA would prove to be extremely useful for modelling physical and biological systems. Since then, CA have served as an effective model for witnessing global behaviour in systems where local interactions form the fundamental interaction in the system [8,16,18,21,22].

2.2.3 Cellular Automata: A formal definition

a) A lattice, or grid, of cells covering k-dimensional space.

b) A set $C(s,t) = \{C_1(s,t), C_2(s,t), ..., C_n(s,t)\}$ of variables attached to each cell s of the lattice, giving the local state of each cell at the specific time step t.

c) A rule $R = \{R_1, R_2, ..., R_n\}$ specifying the evolution of the states $C(s,t)$ over time and in the following way:

$$C_j(s, t+1) = R_j\big(C(s,t), ..., C(s+\delta, t)\big)$$

where $s + \delta_m$ designate the m cells which belong to a given neighbourhood of the cell s.

This definition was provided by Giitsidis & Sirakoulis 2016 [8]. The transition rule, R, is identical for all cells in the lattice and is applied simultaneously, resulting in synchronous dynamics from which inhomogeneities can be introduced [8]. Moreover, the new state of a particular cell s is a function of the previous state of the cell, s at the previous time step, and the state of the cells belonging to its designated neighbourhood at the previous time step. The designated neighbourhood for each cell is predefined and influences the dynamics of CA; the neighbourhood determines the size of region for which cells will interact [8,15,16]. In two-dimensional CA, there are two common neighbourhoods:

a) The von Neumann neighbourhood, consisting of the centralized cell to be updated and the cells located in nearest cardinal directions (North, South, East, West). The von Neumann neighbourhood includes 5 cells [8,16].

b) The Moore neighbourhood consists of all cells belonging to the Von Neumann neighbourhood, as well as the cells located in the nearest ordinal directions (Northeast, Northwest, Southeast, Southwest). The Moore neighbourhood includes 9 cells [8,16].

2.2.4 Limitations

A common limitation of CA is the anisotropy induced from the discrete space. In essence, this is when the model begins to show characteristics of the grid, neighbourhood, and transition rule instead of the physical model being represented. In the case of using CA for this study, we are asking the cells at the back of the airplane to undergo the transition rule before progressing unidirectionally towards the front of the airplane. Anisotropy can be mitigated by decreasing the cell size and increasing the size of the lattice under the same dimension. While these limitations present drawbacks to using CA for "normal" sized lattice structures, it is well documented that an increase in lattice size heavily increases the computational time required for performing CA simulations, unless parallel computing is available for running each cell separately [8,16]. To increase stochasticity, we introduce a factor that prevents a passenger from moving in their cell with 10% chance.

3 Boarding Methods

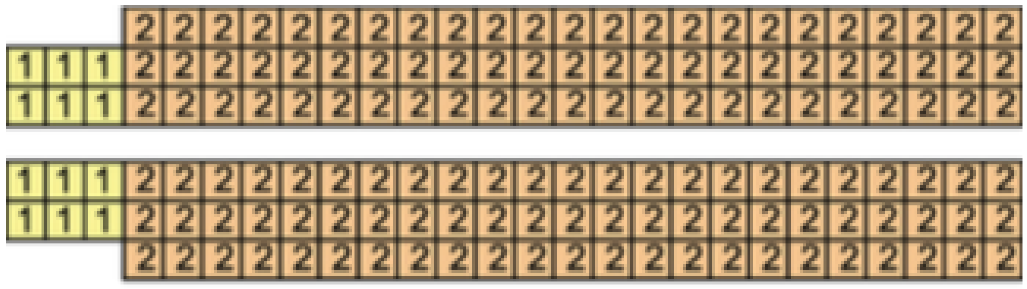
3.1 Boarding Layouts Considered

Recall that we will be investigating airplane boarding for the Airbus A320 and Boeing 737, as well as the Airbus A380 and Boeing 777-300ER. Giitsidis & Sirakoulis 2016 represent the layout for the Airbus A320 and Boeing 737 with 26 rows and 150 seats, where the first 3 rows have 2 seats on each side of the aisle and the remaining 23 rows have 3 seats on each side of the aisle (Fig. 2 (a)). Our representation of this airplane includes 3 seats on each side of the aisle (abbreviated to 3-3) for 25 rows, amounting to the same number of passengers boarding: 150. Our layout simplifies the boarding layout with emphasis placed on investigating standard class seats (Fig. 3 (a)). Similarly, for the Airbus A380 and Boeing 777-300ER, Giitsidis & Sirakoulis 2016 use a layout showing 3 seats outside of both aisles and 6 seats in between (3-6-3) for 45 rows, where the 45 rows are split into 4 sections separated by 3 lateral aisles. The lateral aisles create sections of 10, 15, 12, and 8 rows, respectively, front-to-back (Fig. 2 (b)) [8].

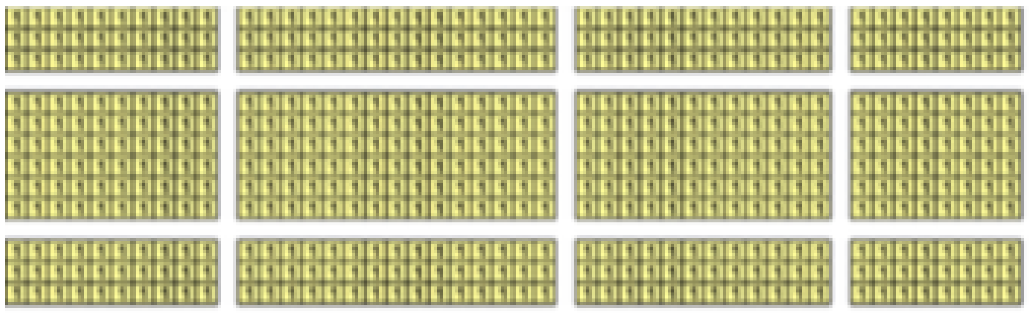
After consultation with current seating layouts of the Airbus A380 and Boeing 777-300ER families, we found that all airplanes follow a seating arrangement of 3-4-3, with 3 seats outside of each aisle and 4 seats in between. While these airplanes accommodate 540 passengers, the amount of seats for standard class seats ranges from 399 to 557 for the Airbus 380-800 and 304 to 385 for the Boeing 777-300ER. There are 3 versions of the Airbus A380-800: The Airbus A380-800 3 class version 1, the Airbus A380-800 3 class version 2, and the Airbus A380-800 2 class. The Airbus A380-800 2 class airplane has the most standard class seats with 557; however, this seating layout has an upper level holding approximately 150 standard class seats. Since this paper does not consider boarding passengers on the second level, we consider the Airbus 380-800 to seat between 399 and 427 standard seats [23,24,25,26,27].

Next we consider the sectioning of standard seats. Front-to-back, the Airbus A380-800 3 class version 2 and the Airbus A380-800 2 class both contain 4 sections of rows on the lower level, each section separated by emergency rows and rows for utilities. The number of rows per section is approximately 8, 14, 13, and 8 from front-to-back. Explicitly, in both of these versions of the Airbus A380-800, the first section can be represented by a section of 8 rows, each with 10 seats, and the second section can be represented by 14 rows, each with 10 seats, and so on. In contrast, the Airbus A380-800 3 class version 1 has a fourth section of approximately 5 rows, resulting in approximately 30 less seats on the lower level than the aforementioned versions of the Airbus A380-800 [23,24,25,26,27].

The Boeing 777-300ER series also has 3 versions. Looking at the Boeing 777-300ER 3 class version 1, we have front-to-back sectional row counts of 5,13, and 12, respec-



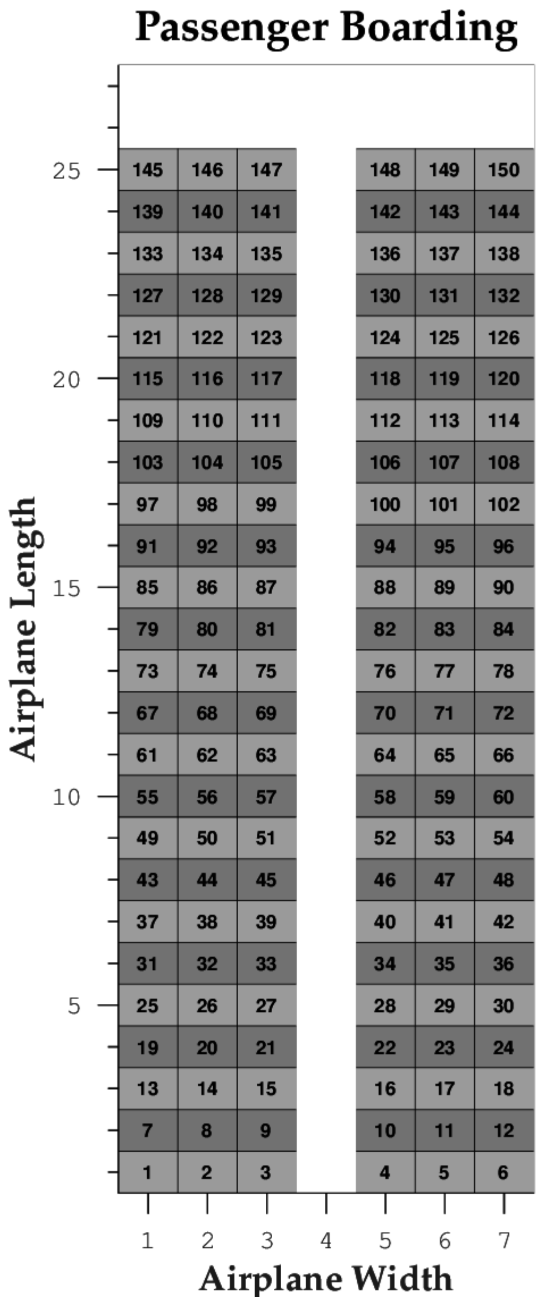
(a) Airbus A320 and Boeing 737



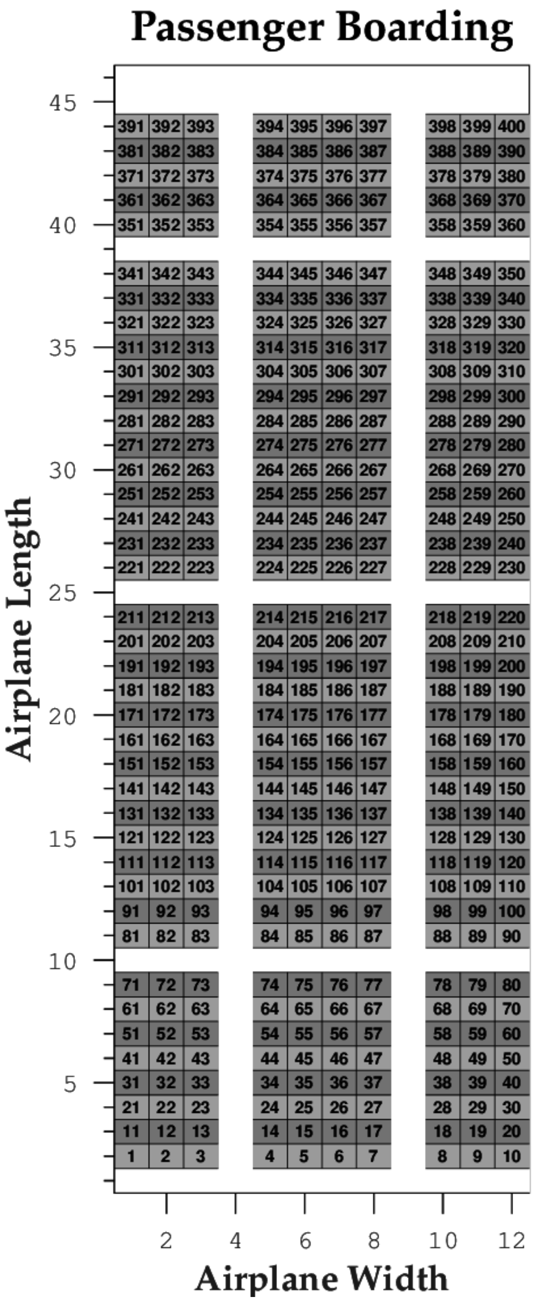
(b) Airbus A380 and Boeing 777-300ER

Figure 2: Giitsidis & Sirakoulis 2016 representation of the two airplane layouts. The numbers in each cell were used to identify groupings of passengers to board. For the purpose of (a), the numbers show the different seats for business class seating vs. standard seating. For (b), the numbers show that all seats are representative of standard seating. The simulations were run from left to right [8].

tively, whereas the Boeing 777-300ER 3 class version 2 has front-to-back sectional row counts of 5,13, and 13, respectively. Lastly, the Boeing 777-300ER 2 class differs from these layouts and has front-to-back sectional row count of 14, 13, and 12 [23,24,25,26,27]. Given the variation of boarding layouts within the Airbus A380-800 and Boeing 777-300ER families, this paper will serve to represent the Airbus A380-800 with 400 seats, adopting the row counts of 8, 14, 13, and 5 for each section from front-to-back. This seating layout serves well to represent the maximum number of passengers boarding a Boeing 777-300ER, specifically 15 passengers over the Boeing 777-300ER 2 class capacity, while having a sufficient number of passengers to consider boarding an Airbus A380-800. Therefore, the template for the double-aisle airplane layout will be the Airbus A380-800 with 400 passengers in an arrangement of 3-4-3, where the first section will have 8 rows of seating, followed by 14, 13, and 5 rows, respectively, in the subsequent sections (Fig. 3 (b)) [23,24,25,26,27].



(a) Airbus A320 and Boeing 737



(b) Airbus A380 and Boeing 777-300ER

Figure 3: The representation of the airplane layouts used in this study. (a) 1 main aisle with 3 seats on each side: 3-3 arrangement. The passengers enter from the centre aisle at the bottom. (b) 2 main aisles with 3 intermediate emergency rows allowing for lateral flow of passengers: 3-4-3 arrangement. The passengers enter from the bottom right corner of the layout, traverse the entrance row, and then get directed upwards.

3.2 Boarding Strategies used by Airline Companies

Many airline companies implement sectionalized boarding strategies, in which passenger boarding priority is more involved. First class individuals, families with small children, frequent flyers, online check-in individuals, and passengers which purchased their tickets with certain credit cards will be given first priority when boarding commences [8]. Following priority boarding, we have the following common boarding strategies used by designated airline companies, all of which are depicted in Figure 5. We reiterate that our model omits business class seating, constituting the left-most 3 columns of each layout (first 3 rows of the airplane) shown in Figure 5. Instead, the 3 rows of 2 business seats are replaced by 2 rows of 3 standard seats, equating to 25 rows of standard seating, as mentioned above.

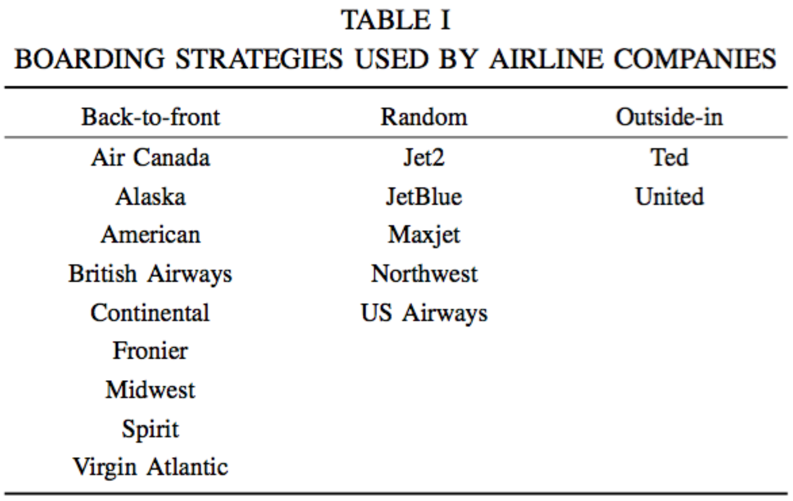


TABLE I
BOARDING STRATEGIES USED BY AIRLINE COMPANIES

Back-to-front	Random	Outside-in
Air Canada	Jet2	Ted
Alaska	JetBlue	United
American	Maxjet	
British Airways	Northwest	
Continental	US Airways	
Fronier		
Midwest		
Spirit		
Virgin Atlantic		

Figure 4: Boarding strategies used by airline companies following priority boarding [8].

3.3 Outlining boarding strategies

The boarding strategies which are about to be discussed are taken from the aforementioned papers and relevant literature on the aircraft boarding problem. These boarding strategies will be implemented in the CA model to recreate and validate existing simulations. Modified and new boarding strategies will be discussed in the results section. To reiterate, all of the following boarding strategies are considered to take place after priority boarding in this study.

3.3.1 Random Boarding Strategy

Serving as the most traditional boarding strategy is the **Random** boarding strategy. In this strategy, passengers board the plane in random order relative to their seating arrangement. From the perspective of an airline company, this method is the most convenient and economical as no extra steps are required, other than the action of asking passengers to board the plane. This strategy is shown in Figure 5 (a) [1,7,8,11,28].

3.3.2 Back-to-Front Boarding Strategy

As it sounds, the **Back-to-Front** boarding strategy involves passengers boarding the plane in groups, where the groups for the back of the plane are sent in first. For example, if 100 people are to board the aircraft, groups of 20 are sent in at a time. This is depicted in Figure 5 (b) [1,7,8,11,28].

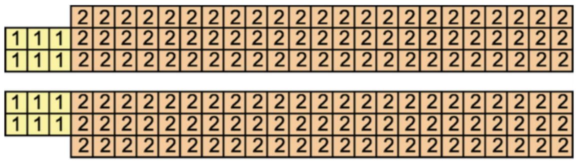
This method was proposed for reducing aisle interferences given the inherent nature of unidirectional flow in airplane boarding. Specifically, suppose two passengers are boarding an airplane in the following order, passenger A boards first and is followed by passenger B. Now suppose passenger A's seat is closer to the front of the plane than passenger B, then passenger B must wait for passenger A to stow their carry-on and take their seat. If passenger B entered the plane ahead of passenger A, then, given the same seating arrangement, an extra wait time is not required as passenger A would find their seat while passenger B is liberated to continue down the aisle to their designated row and seat. The simple logic of this strategy justifies its use by airline companies globally [1,7,11].

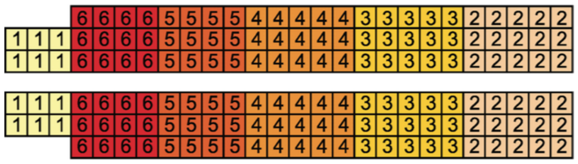
3.3.3 Rotating Zone Boarding Strategy

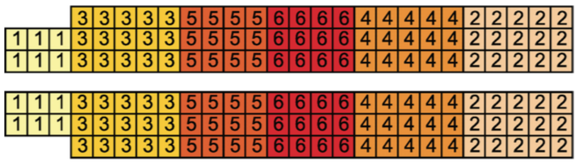
The **Rotating Zone** boarding strategy uses the same grouping as the **Back-to-Front** boarding strategy, but the groups do not necessarily need to enter the aircraft in the aforementioned "back-to-front" order. Figure 5 (c) shows a possible zone rotation where the third group entering the plane is seated in the front [8].

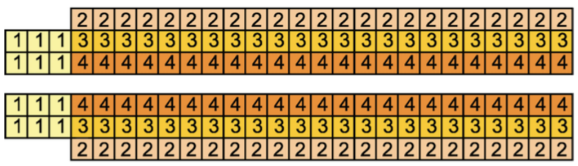
3.3.4 Random Outside-In Boarding Strategy

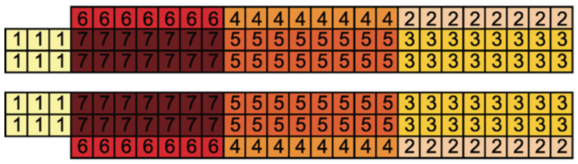
The **Random Outside-In** boarding strategy involves the passengers being divided into groups based on the position of their seat relative to the window. For example, for a plane with 1 centre aisle and 3 seats on either side, the passengers would be divided into groups of 3 where all window seats are sectioned into the first group, the middle seats would comprise the second group, and the aisle seats would make up the third


(a) Random boarding strategy


(b) Back-to-Front boarding strategy


(c) Rotating Zone boarding strategy


(d) Random Outside-In boarding strategy


(e) Block boarding strategy

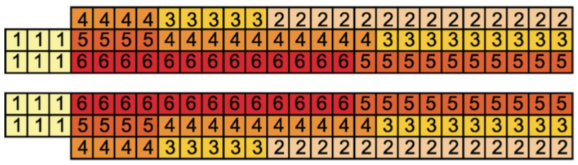

(f) Pyramid boarding strategy

Figure 5: Giitsidis & Sirakoulis 2016 representation of relevant boarding strategies for the Airbus A320 and Boeing 737. Each collective group of equal numbers represents a section of the airplane that will board collectively. In (a), all cells labelled as “1” will board collectively at random. Once the passengers with seats labelled as “1” have entered the airplane, then the cells labelled as “2” will be permitted to board, and so on. Hence, (a) represents the random boarding strategy since standard seating is boarding at random [8].

and final group to board the plane. The strategy is random since within each of the 3 groups entering the plane, there does not exist and ordering of passengers, virtually analogous to the random boarding strategy performed 3 times. The inclusion of the term “outside-in” follows from the logic behind the Back-to-Front boarding strategy, this time referring to the ordering of passengers along the width of the plane. Figure 5 (d) shows this boarding strategy [1,7,8,9,11,28].

3.3.5 Block Boarding Strategy

The **Block** boarding strategy encompasses ideas from the Back-to-Front, Rotating Zone, and Random Outside-In boarding strategies. First, in similar fashion to the Back-to-Front boarding strategy, passengers are arranged into groups which will board the plane back-to-front. Within these groups, the passengers are further divided into smaller components such that each group will board outside-in, mirroring the

Random Outside-In boarding strategy. We achieve the following hybrid boarding strategy depicted in Figure 5 (e) [8,28].

3.3.6 Reverse Pyramid Boarding Strategy

The block boarding strategy with less restrictions defines the **Reverse Pyramid** boarding strategy. This strategy comprises Back-to-Front and Random Outside-In boarding ideas while excluding the need to board by block formation. The boarding groups will still board at random within each group. This boarding strategy is shown in Figure 5 (f) [1,7,8].

3.3.7 Steffen Boarding Strategy

A combination of the Back-to-Front and Random Outside-In boarding strategies. The **Steffen** boarding strategy boards the passengers from back-to-front, working inward from the window seats. Once all passengers with window seats have boarded the plane, passengers with seats between the window and aisle seats are permitted to board. The novelty of the Steffen strategy is that passengers board back-to-front for every other seat, giving enough space for passengers to stow luggage. This strategy is the new benchmark for achieving the fastest boarding time [7,8,29]. An outline of this strategy is described in the results section.

3.4 Airplane Boarding Literature

In 1998, Boeing released an article in the magazine “Aero” describing a program PEDS (Passenger Enplane/ Deplane Simulation) which was used to compare the flight preparation times between a Boeing 757-300 and Boeing 757-200. The article gives an historical viewpoint on the steadily increasing airplane turnaround time from 1975 to 2000 [5,7,9,10]. The article also shined light on implementing alternative boarding strategies to reduce boarding time. The intention was to show that an airplane could be prepared to fly in a reasonable amount of time, independent of airplane size. For this reason, Boeing began collecting statistical data on procedures taking place during aircraft preparation. Boeing released the following outline, see Figure 6, for the duration of procedures involved in preparing the Boeing 757-300 and Boeing 757-200 for flight. Historically, it is evident that boarding an airplane has been a critical component for increased airplane turnaround time [5,7,8,9,10].

Van den Briel *et al.* 2005 studied multiple variations of Back-to-Front and Random Outside-In boarding strategies with a variable time delay for each designated interference; each interference had its own randomized time delay [7,8]. Steiner and Philipp

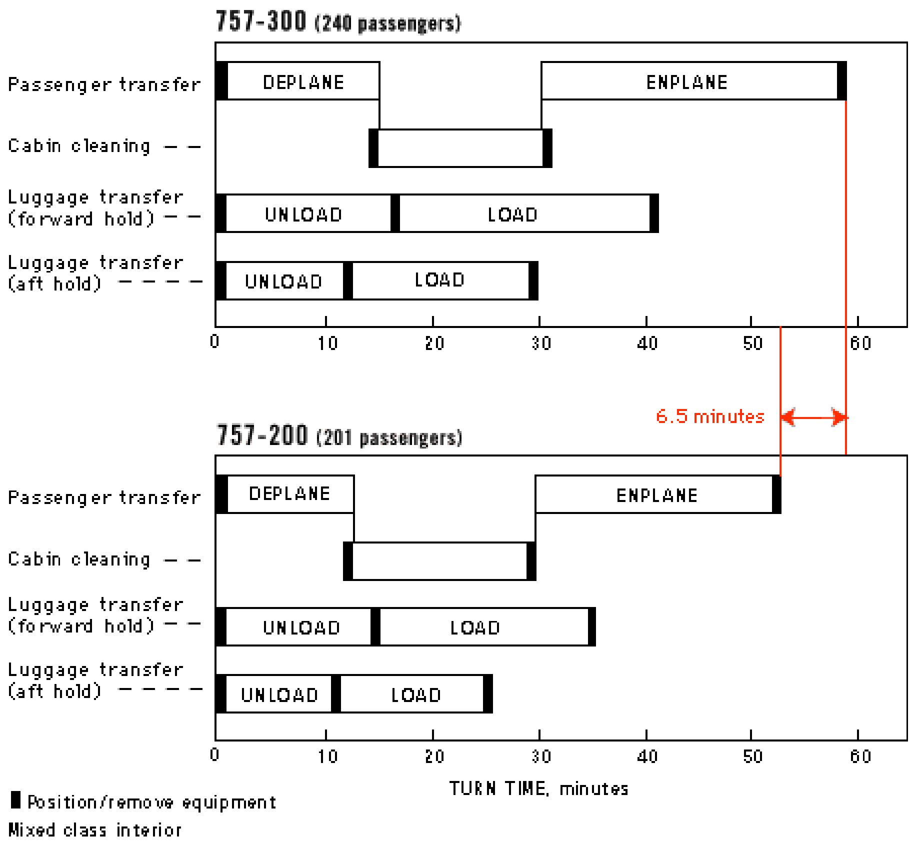


Figure 6: Duration of Airplane Preparation Procedures as outlined by Boeing's PEDS program in 1998 [9].

studied the airplane boarding problem beginning with the check-in procedure, the boarding strategies included Random and Back-to-Front [5]. Steiner and Philipp consider more complex boarding methods to require excess staff and methodology to be practical and carried out successfully [5,8]. Steffen 2008 observed that the time required for passengers to stow away their carry-on was a major factor in delaying passenger boarding time [29]. Steffen 2008 applied the Markov Chain Monte Carlo Optimization algorithm with computer simulations to discover an optimal boarding strategy. Steffen 2008 proposed that passengers board such adjacent passengers boarding the plane do not have sequential row numbers, giving each passenger enough row space to reduce the time spent stowing luggage [29]. Wallace 2013 proposed, and coined, the "Flying Carpet" boarding method as the ideal boarding method [30]. Wallace's ideas have common ground with Steffen 2008, in that Wallace agrees that the arrangement of the carry-on is the most time consuming process in passenger boarding. Wallace sections passengers into groups of 20 to 30 and randomly boards them onto the airplane such that passengers can experience more space to stow away luggage [30]. The term "Flying Carpet" comes from the way the passengers are oriented prior to boarding: in the waiting room there exists a seating arrangement, or “carpet” resembling the aircraft seat layout, from which passengers are standing in the arrangement they will be seated on the plane. This idea allows passengers to develop familiarity with the layout prior to boarding and prevents mixing of passengers that do not belong to the boarding group [8,30].

4 Constructing the Model

4.1 An Overview

The CA model employed in this study was constructed in GNU Octave, an open-source scientific programming language largely compatible with MATLAB®. The model incorporates an extension of the von Neumann neighbourhood. There are 4 main components of the model:

1) A background image of the airplane's seating layout, as shown in Figure 3.
2) A preset seating arrangement that orders each person in the form of a matrix, M_{person}, designating the ordered seat number (shown in Figure 3), calculated row and column numbers, and attributes of each passenger.
3) The 4−dimensional matrix, M_{steps}, on which the model operates using the M_{person} matrix as a seed for each subsequent person; M_{steps} numerically shows the occupancy of the airplane's cells over discrete time.
4) A matrix M_{p} containing 3 "layers" of points superimposed on the background image allowing the points to be toggled "on" and "off" depending on the occupancy of a person in a particular cell; each point represents a passenger's position in the airplane and each cell contains up to 1 passenger.

4.2 The Background Image

The background image was constructed using Octave's image function, which takes a n x m matrix of values, $M_{\text{background}}$, and converts each matrix entry into a coloured pixel of a an image with n x m pixels. Each cell of the image constitutes a space of occupancy on the airplane. Since the number of rows and columns belong to the set of natural numbers $\mathbb{N}$, we also identify each space in the aisle between two rows as a single cell. Every cell can occupy up to 1 passenger and is represented by 1 pixel of the image. For emphasis on the movement of passengers along each aisle, if a passenger is not obstructed by another passenger then this passenger is permitted to move after restrictions put in place by the model; if a passenger can move to the next row, unobstructed and not restricted by interferences or a condition in the model, then the passenger must move.

The value of each entry of $M_{\text{background}}$ determines the transparency of the colour for that pixel in the image. The colour is set by Octave's colormap function. For the purpose of this model, the colour map was set to greyscale. Although the rows and columns of the plane are distinguished by black lines, the application of the greyscale colour scheme to the rows of the airplane aesthetically help with this differentiation. Hence, some rows of $M_{\text{background}}$ have smaller numerical values and some have larger numerical values, giving rise to some rows in the background image having lighter shades of grey than others.

4.3 The Seating Arrangement

Once the background image is in place, it is necessary to initialize the variables. The number of rows (variable: **numofrows**) and columns (variable: **numofcolumns**) are defined for the seating arrangement of the airplane. It is also necessary to define the number of aisles and emergency rows, as we can subtract these numbers from the total to get the amount of seats available for occupancy on the airplane. The numofrows variable includes the emergency rows. The emergency rows permit passengers to traverse the airplane from one aisle to the other, should the airplane have more than 1 aisle. The numofcolumns variable includes the main aisles, which passengers use to walk the length of the airplane.

To illustrate the use of variables we refer to Figure 3 (b). Here, numofrows is 46 due to 1 entrance row (empty row where the passengers enter from the right) and the 5 emergency rows. The last two emergency rows serve as overflow of passengers. Hence, the number of emergency rows is defined to be 5 resulting in 40 rows of seats. Similarly, numofcolumns is 12 and the number of aisles is defined to be 2. So for the Airbus A380-800 we have 10 columns of seats, adopting the 3-4-3 arrangement. Reiterating, columns 4 and 8 represent the aisles and do not contain any seats. Once the dimensions of the plane are initialized, we calculate the number of seats on the plane and equate this to the number of passengers (variable: **numofpassengers**) which will board.

As shown in Figure 3, for each layout the seats are ordered from 1 to numofpassengers and each seat is given an ordered seat number (variable: **ordered seat number**). The ordered seat number shows the total number of seats available and becomes is useful given the way the passengers are defined and introduced to the airplane. The passengers enter the airplane via an array, which is given numbers from 1 to numofpassengers, equating the number of available seats. Arranging and permuting the ordered seat numbers in the array yields different boarding scenarios.

4.4 The Operating Matrices for Boarding the Airplane

The array is loaded into the entrance of the airplane one index of the array per time step, starting from index 1. Since the seating layout is $2-$dimensional, we need to break the ordered seat numbers into components to make them interpretable by Octave in the form of matrix entries. The calculated row and column numbers are calculated using modular arithmetic with Octave's ceil and rem functions. The components represent the respective row and column for each passenger's preassigned ordered seat number. These components need to be kept in memory so that Octave can ask if the person has reached their desired row, and then column, at each time step. So, to utilize the components, we need to load more than an $1-$dimensional array into the airplane entrance. Therefore, we construct a matrix, M_{person}, to store the ordered seat number, the calculated row number, and the calculated column number. M_{person} has size $3-$by$-$numofpassengers. The aforementioned array holding the ordered seat numbers are now inserted into the first row of M_{person}, with each row underneath holding the calculated row and column numbers. If additional attributes are provided, such as "time required to stow away luggage", additional rows can be added. To illustrate this, a column of a $4-$by$-$numofpassengers variation of M_{person} is outlined below, with each row of each column representing the following 4 attributes of each passenger (variable: **numofattributes**), respectively:

1) The ordered seat number
2) The row of the seat number
3) The column of the seat number
4) Time required to stow away luggage

If the passengers are boarding the plane by ascending ordered seat number, then the first 15 passengers will have ordered seat numbers from 1 to 5 and we have the following representation of M_{person}:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & \dots \\ 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & \dots \\ 1 & 2 & 3 & 5 & 6 & 7 & 1 & 2 & 3 & 5 & 6 & 7 & 1 & 2 & 3 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \end{bmatrix}$$

Let this array be referred to as $M_{\text{person ordered}}$. The passengers ordered seat number is shown in the first row, which corresponds to the seat numbers displayed in Figure 3 (a). The second row shows the row number and the third row shows the column number. We notice that a column number of 4 is not listed in M_{person} since the fourth column refers to the aisle for the Airbus A320/ Boeing 737. Also notice that the fourth row is defaulted to 0 because the time required to stow away luggage was defaulted to zero. As we can see in Figure 3, the airplane layout is $2-$dimensional. Inserting the ordered seat numbers, row numbers, and column numbers into a $2-$dimensional matrix for a single cell is not possible unless we give each cell additional cells. Hence, the $2-$dimensional airplane layout is fitted with a third dimension: 2 additional layers capable of storing the calculated row and columns numbers, respectively. We can visualize this in Figure 7.

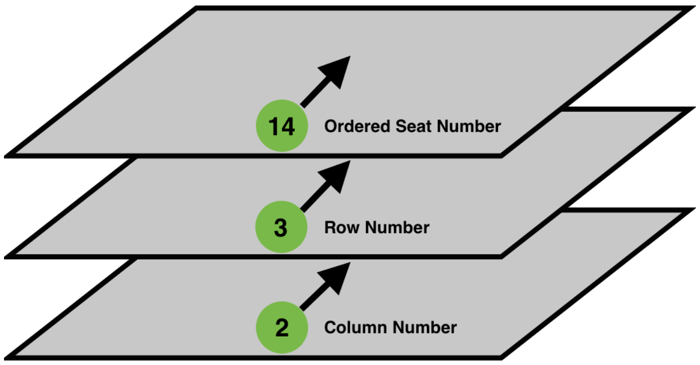


Figure 7: A visualization of M_{steps} with only inserting the components for the ordered seat number 14, without the fourth component for luggage stow away time. The ordered seat number, row number, and column number agree with Figure 3 (a). All 3 numbers then move together along their respective layers: 1, 2, 3 from top to bottom. This image is a modified version of the original by Geoff Richards, found in Wikimedia Commons [31].

This 3−dimensional matrix is a subspace of M_{steps} because M_{steps} needs to represent all 3−dimensional matrices over discrete time. So M_{steps} becomes a 4−dimensional matrix with time as the fourth dimension. Explicitly, the M_{steps} matrix shows the numerical occupancy of the airplane's cells over discrete time. M_{steps} is a 4−dimensional matrix of the following approximate size (numofrows can be padded for overflow of passengers along an aisle):

(numofrows) x (numofcolumns) x (totalsteps) x (numofattributes)

4.4.1 Boarding the Airplane: M_{person} & M_{steps}

When the first column of M_{person} is inserted into the airplane, at a maximum rate of 1 column per discrete time step, the M_{steps} matrix assumes the column given by M_{person} , i.e. the column in M_{person} is loaded into M_{steps} where the passengers are designated to enter the airplane. Let's take M_{person} to be ordered such that the passengers board the plane from seat 1 to seat numofpassengers, as displayed in $M_{\text{person ordered}}$. Consider the Airbus A320/ Boeing 737 with numofpassengers = 150. We refer to each passenger as "passenger N" where N is the ordered seat number for that passenger. The first passenger is passenger 1 by definition. Passenger 1 enters the plane if the entrance of the plane is empty, meaning if M_{steps} has a value of 0 for the coordinate M_{steps}(row=1, column=4, time=1, layer=1) (recall from previous figure that layer 1 corresponds to the top layer of M_{steps}). Since the airplane is empty, the entrance cell is empty by default and the ordered seat number, row number, and column number can be entered into their respective layers.

Now, the model checks if passenger 1 has reached their row by checking layer 2, specifically M_{steps}(row=1, column=4, time=1, layer=2). If the row number assigned to the passenger agrees with the current row of passenger 1, then check layer 3 for the column number. Insert to the left if the column number is greater than the aisle number, else insert to the right. We note that an insertion only happens if the cell the passenger needs to enter is vacant. Consequently, the second passenger can enter the plane once passenger 1 has vacated the entrance cell, leading to the next point. To move passengers to a new cell, the model must scan the airplane rows and columns against the flow of passengers: outside−in along the rows and and back−to−front along the aisle. In other words, the model must consider moving passengers that boarded earlier. For example, suppose passengers A and B board the plane with passenger A ahead of passenger B by 1 cell. Then if the model asks passenger B to move first, passenger B will remain in the current cell at the next time step since passenger A is occupying the cell of interest. If passenger A is asked to move first and successfully vacates the cell passenger B wishes to occupy, then passenger A and B can both advance in one time step, as desired.

4.4.2 Toggling Points to Display Occupancy

The M_{steps} matrix is used to numerically keep track of all passengers in the airplane at each time step, such that we understand the cells currently occupied and the cells to be occupied. On top of having M_{steps} to indicate the present of passengers, we need to represent passengers as points overtop the background image using M_{steps} as the source. The matrix M_{p}, standing for "matrix of points", is constructed with 1 dimension less than M_{steps}. Specifically, M_{p} is M_{steps} without time. M_{p} also has 3 layers, with the first layer holding the ordered seat number for each cell holding a seat in the airplane layout. When a passenger does not occupy a cell with a seat, these numbers are *toggled* "on".

When a passenger occupies a cell with a seat, this number is toggled "off" and the passenger's ordered seat number presents itself in the cell. Of course when a passenger occupies a cell without a seat, the passenger's assigned ordered seat number presents itself in the cell. The passenger's ordered seat number is coded into layer 3 and layer 2 holds a green circle (point) to visually represent the passenger. When a passenger occupies a cell, layers 2 and 3 are toggled "on" and layer 1 is toggled "off". If the cell represents an aisle or emergency row, layer 1 is always toggled "off" by default. Since a passenger will pass through seats to reach their own seat, these layers in M_{p} are useful. When this happens, we want to see the passenger's number traverse the seats, masking the ordered seat numbers of the seats underneath, to keep track of the passenger's destination.

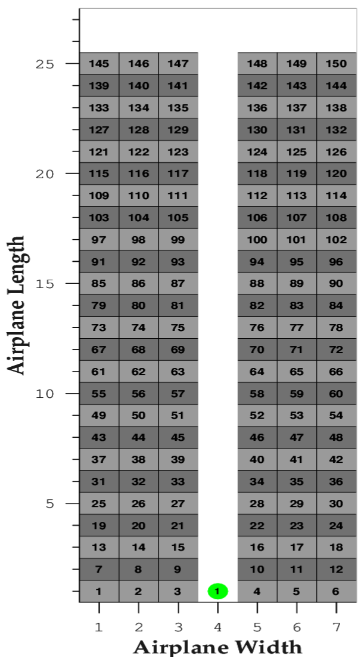
4.5 Model Operation and Assumptions

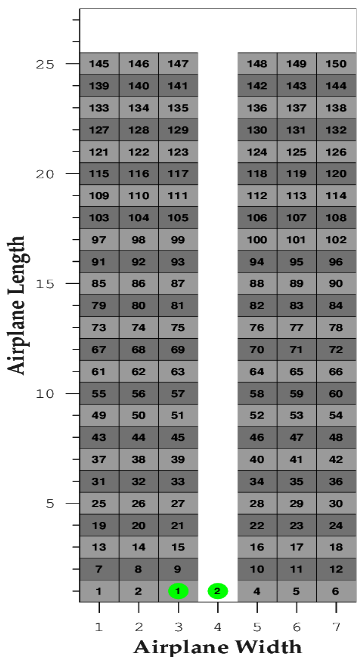
Below we view our first representation of passengers boarding an Airbus A320/ Boeing 737 by ascending ordered seat number, from 1 to 150 in M_{person}. Recall the matrix $M_{\text{person ordered}}$. We note that this is not a boarding strategy used by airline companies, it only serves as a boarding method to illustrate the CA model.

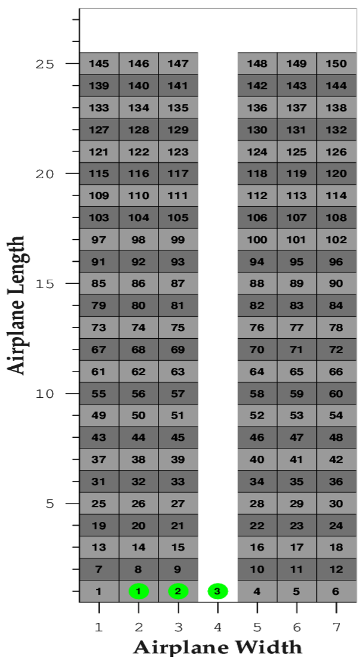
4.5.1 Interferences and Shuffling

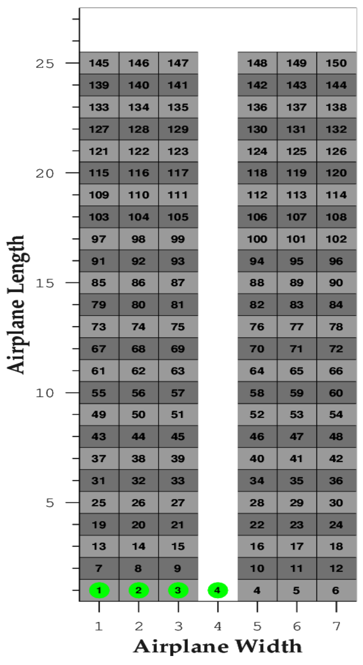
In Figure 8 (a) we see the first passenger board the plane with the assignment following the first column of $M_{\text{person ordered}}$. Ordered seat number 1 indicates row 1, column 1. The model asks if the row number of the passenger matches the row number the passenger is situated in. Since this is true, we see the first passenger move towards their seat in time step 2. Following this, passenger 1 has vacated their cell in time step 2, allowing passenger 2 board the plane simultaneously. Passenger 2 also meets the row criteria and follows passenger 1 in time step 3. This repeats until Figure 8 (e), where passenger 4 vacates the entrance square and moves to the right, allowing passenger 5 to board the plane. Passenger 4 has found their seat, but passenger 5 needs to move past passenger 4 to get to their seat. In the next time step, we see that a **shuffle** has been performed. Passenger 4 recognizes that they need to vacate their seat to allow passenger 5 to reach their seat. Passenger 4 shuffles out of row 1 at the next time step and into row 2, shown in Figure 8 (f). It is in this logic where the model extends the idea of the von Neumann neighbourhood, and closer to the Moore neighbourhood, since passenger 4 will not vacate their seat unless the aisle cell in the second row is vacated. Given the exclusiveness of this interaction to shuffling, we still consider the neighbourhood to extend from the von Neumann definition.

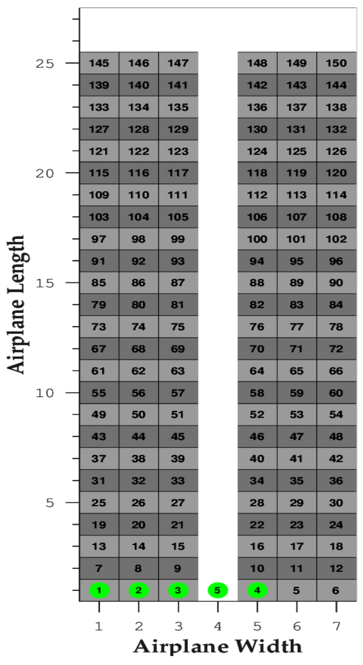
Figure 9 (a) shows passenger 5 entering the fourth column of row 1 with passenger 4 moving into passenger 5's former spot. We note that it is important that passenger 4 moves into passenger 5's former spot instead of passenger 6, as this would cause passenger 6 to prevent passenger 4 from moving downward. We consider Passenger 6 allowing passenger 4 to re-enter their seat a gesture commonly observed when boarding. We consider this logic an assumption in this model. The boarding continues with Figure 9 (b) showing passengers 4 and 5 effectively seated and passenger 6 boarding the plane. Just as before, passenger 6 needs to pass through passengers 4 and 5 to reach their desired seat. To achieve this, a **double shuffle** is performed, shown from Figure 9 (c) to Figure 9 (f). Once passengers 4, 5, and 6 are seated, passenger 7 boards the airplane and the process repeats. Each shuffle requires the passenger in the aisle to remain in their cell if the row they wish to occupy is filled with


(a) Time step = 1


(b) Time step = 2


(c) Time step = 3


(d) Time step = 4


(e) Time step = 5

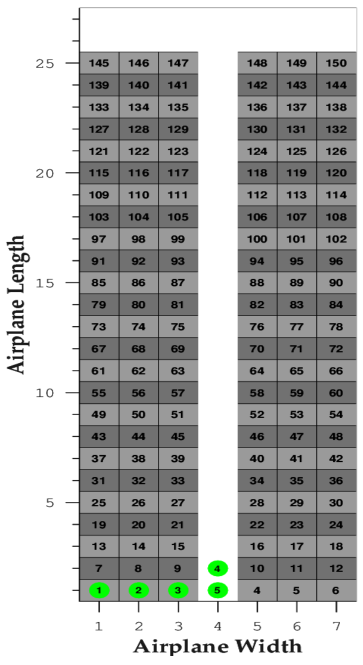
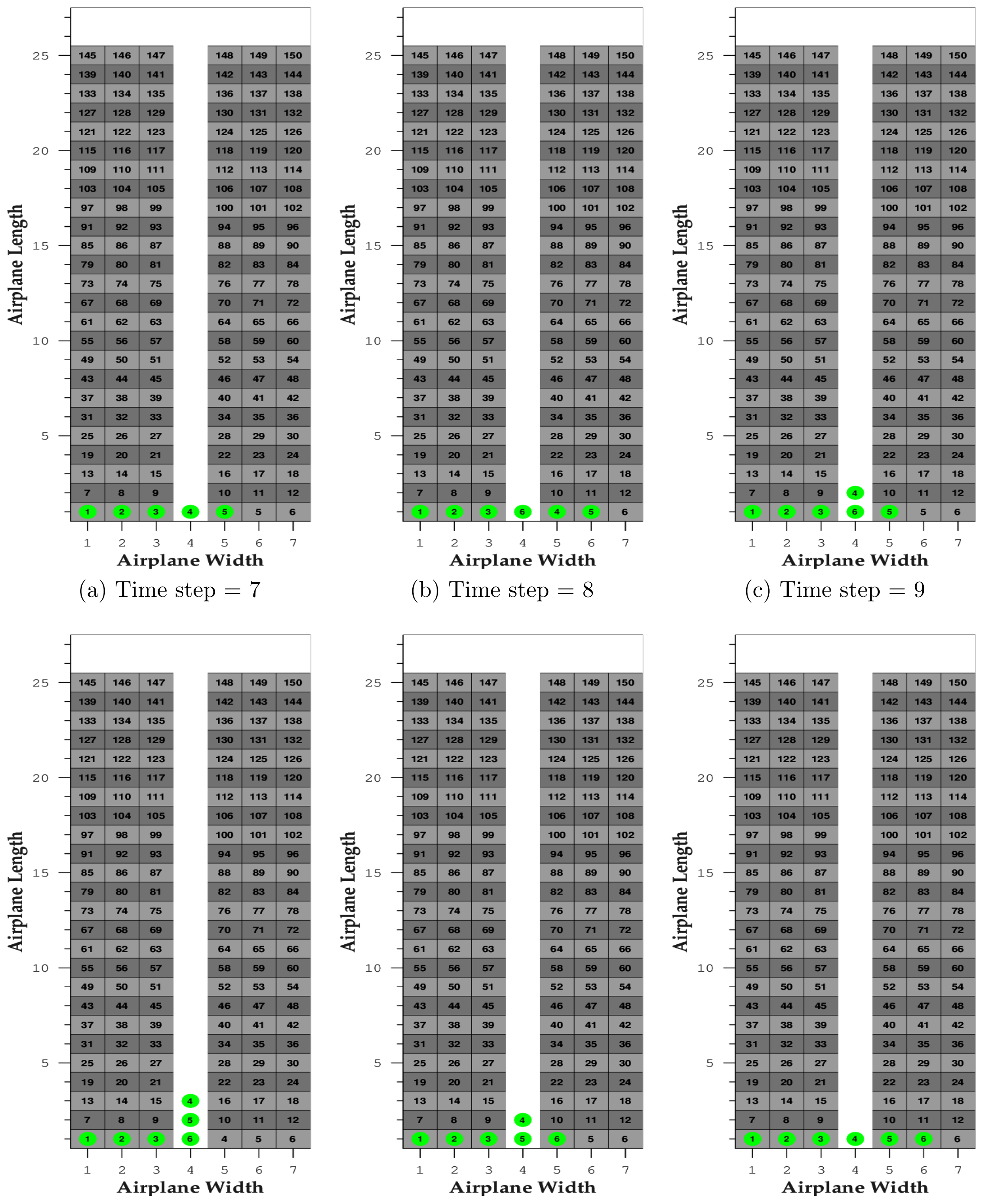

(f) Time step = 6

Figure 8: Boarding the Airbus A320/ Boeing 737. Initiation of a single shuffle is shown.



(a) Time step = 7 (b) Time step = 8 (c) Time step = 9

(d) Time step = 10 (e) Time step = 11 (f) Time step = 12

Figure 9: Boarding the Airbus A320/ Boeing 737. A double shuffle is performed.

passengers obstructing their path. We also note that each shuffle requires passengers already seated to leave their seats and fill in empty aisle space in rows above the row of consideration. If the shuffle requires multiple occupied aisle cells to be vacated, then each passenger in the shuffle, whom requires vacancy, remains stationary and waits for the passenger(s) causing the obstruction to vacate their cell(s). Thus, passengers can remain in their cell at the next time step and subsequent time steps. This level of obstruction is more apparent in relevant boarding methods for the Airbus 320/ Boeing 737, as well as in passenger boarding for the Airbus A380-800/ Boeing 777-300ER.

Given the nature of the layout for the Airbus A380-800/ Boeing 777-300ER, a double shuffle is the largest shuffle required. We first note that the seating on the outside of the aisles for the larger layout is the same layout as the smaller airplane layout. Secondly, the 4 seats between the two aisles receive passengers from the nearest aisle, leaving 2 seats to be filled from each aisle. The left-most seats (2 of 4) only receive passengers from the left aisle and the right-most seats (2 of 4) only receive passengers from the right aisle, as executed by the flight attendants for this family of airplanes [24,25,26,27]. Therefore, the highest level of shuffling performed on the Airbus A380-800/ Boeing 777-300ER can be reduced to the shuffling logic required for the Airbus 320/ Boeing 737. To illustrate this, Figure 10 shows the initiation of a double shuffle for the Airbus A380-800/ Boeing 777-300ER with the Random boarding strategy. Passenger 150 has reached their designated seat, indicated by the red rectangle, and a double shuffle must be performed to allow passenger 150 to enter their seat. We reiterate that luggage stow away time remains at 0 for this illustration of Random boarding for the Airbus A380-800/ Boeing 777-300ER. The subsequent time steps are shown in Figure 11. Notice how passengers 259 and 280 are prevented from continuing until passengers 148, 149, and 150 have performed the shuffle. This becomes more apparent when luggage stow away time is applied to each passenger, an attribute which will be discussed later in this section.

When boarding the Airbus A380-800/ Boeing 777-300ER, passengers from the ramp enter the plane from the side and turn up the aisle. In an Airbus A320/ Boeing 737, the entrance can be reduced to having passengers board from the centre, as illustrated in Figures 8 and 9. Supporting this reduction is the existence of 1 aisle. It is not necessary to extend the entrance if passengers are only influenced by the flow of passengers along a single aisle. For the Airbus A380-800/ Boeing 777-300ER, the entrance is more interesting. The passengers naturally board from the side and the model incorporates this to observe the influence each aisle has on the flow of passengers at the entrance. As the model shows, interferences from either aisle can cause delays in boarding, by backing up passengers on the entrance row up to the boarding ramp [24,25,26,27]. It seems intuitive that an interference, caused in the nearest aisle, is more likely to cause blockage in the entrance row than the aisle farther from the entrance. The distance of nearest aisle to the entrance is comprised of less cells, not retaining as many passengers to mitigate the effects of an aisle interference on the flow of passengers at the entrance. The general result is the prevention of passengers from flowing along the entrance row to their respective aisle, which is possibly unobstructed, since the entrance row is obstructed by a backup of passengers from an aisle interference.

4.5.2 Time: Time steps & Luggage Stow Away Time

The number of time steps for passengers to board an airplane under a given boarding strategy is sufficient enough when comparing boarding strategies to each other. Explicitly, if boarding strategy A completes in t time steps and boarding strategy B completes in $t + 100$ time steps, then we can conclude that boarding strategy A is quicker than B regardless of the defined amount of time per time step. With this in mind, it is necessary to define the amount of time per time step to have tangible results comparable to experimental studies and accessible to the airline industry. Recall that at most one passenger can occupy a cell of the layout at any given time step. Giitsidis & Sirakoulis 2016 determine that the cell size for the smaller plane is 0.92 x 0.92 m^2, and the cell size for the larger plane is approximately 1.1 x 1.1 m^2. Given the identical airplanes used in our study, with analogous boarding layouts, we also assume the average cell size of 1 x 1 m^2. Giitsidis & Sirakoulis 2016 document a range of walking speeds of passengers boarding an aircraft, from 1 to 1.5 m/sec, covered by previous studies. It is also reported that the slower passenger boarding rates are supported experimentally [8]. Considering our model involves the accompaniment of luggage with each passenger, we opt for a movement of 1 m/sec to remain consistent with Giitsidis & Sirakoulis 2016 [8]. Under this assumption, we have the boarding frequency of passengers to be at most 1 passenger per second. This rate also matches the speed of passengers on the airplane. Existing literature shows that as boarding frequency decreases, the differences in boarding time between boarding strategies lessens. However, the time to board increases with passengers less frequently boarding the plane [7,8].

In light of increasing stochasticity, we introduce an interference factor allowing any passenger in any aisle may remain stationary in their cell 10% of the time. This factor is an assumption designed to emulate passenger confusion relating to the layout and seat location, the time taken for passengers to converse with other passengers, and any problems involving physical limitations and/ or luggage disturbances caused by the uncoordinated movement of passengers. We note that this factor does not include the time required to stow away luggage. As previously mentioned, the time required to stow away luggage is an integer assigned to each passenger, where the integer resides in the fourth row of M_{person}. If the passenger is assigned a luggage stow away time of 5, when the passenger reaches their row, the passenger must wait 5 time steps before allowed to take their seat or begin a shuffle. Specifically, the luggage stow away time

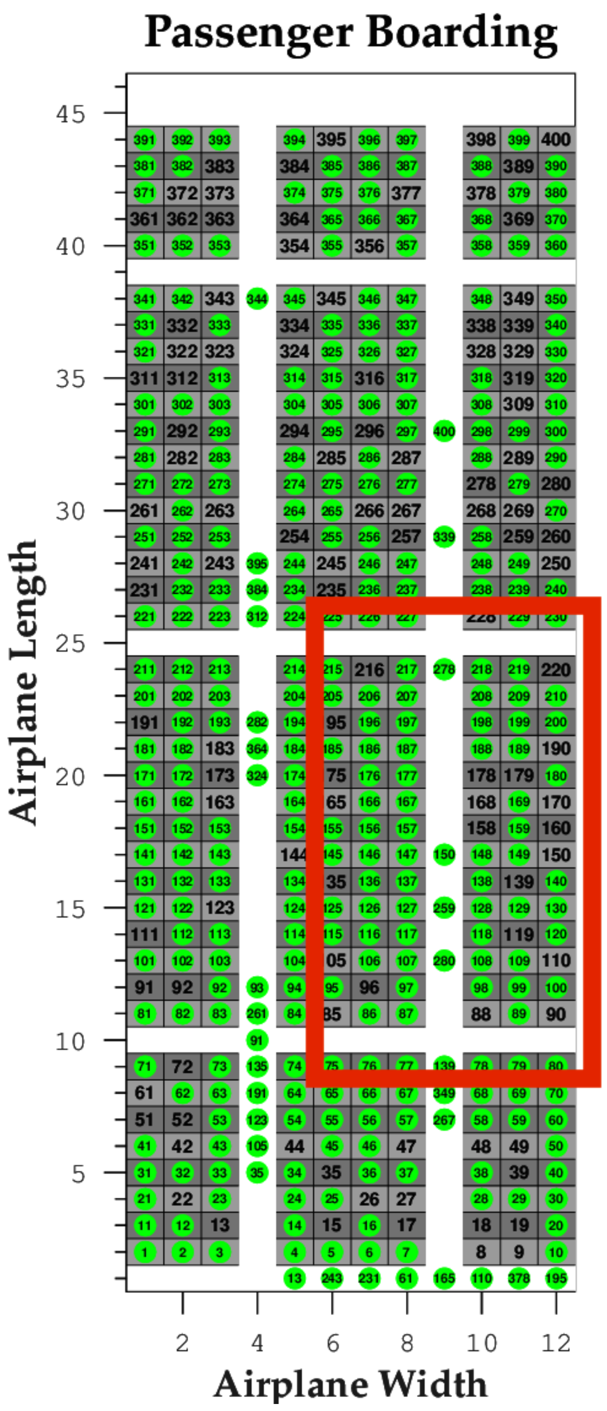
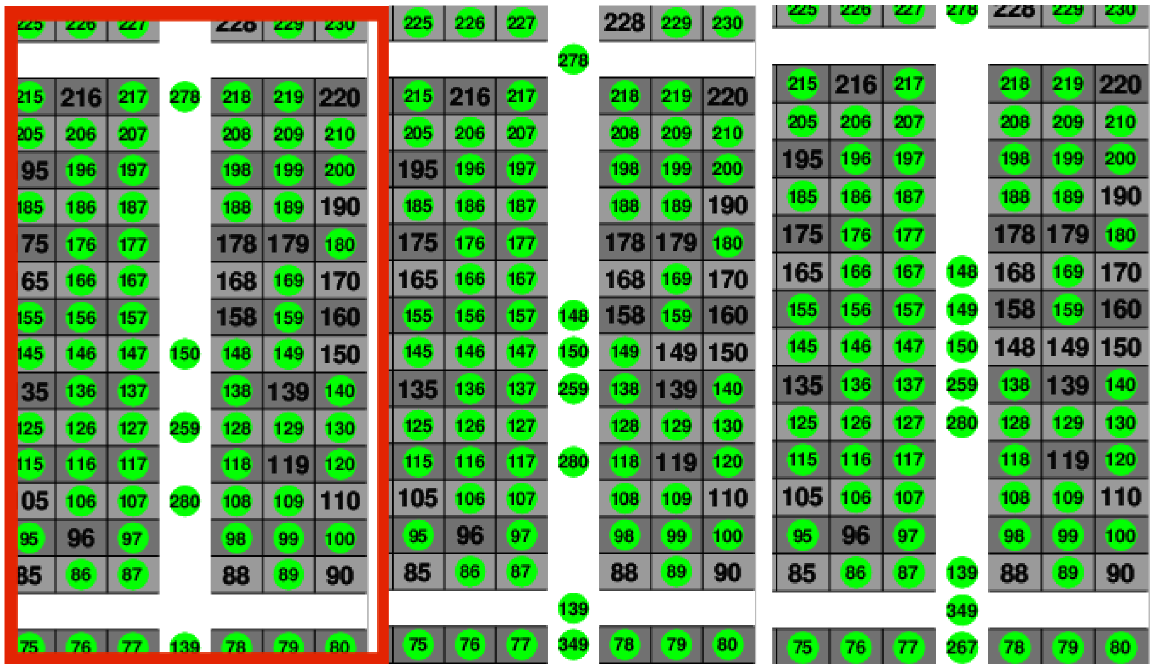


Figure 10: Random boarding the Airbus A380-800/ Boeing 777-300ER. Passenger 150 has reached their designated seat.


(a) Time step = 322 (b) Time step = 323 (c) Time step = 324

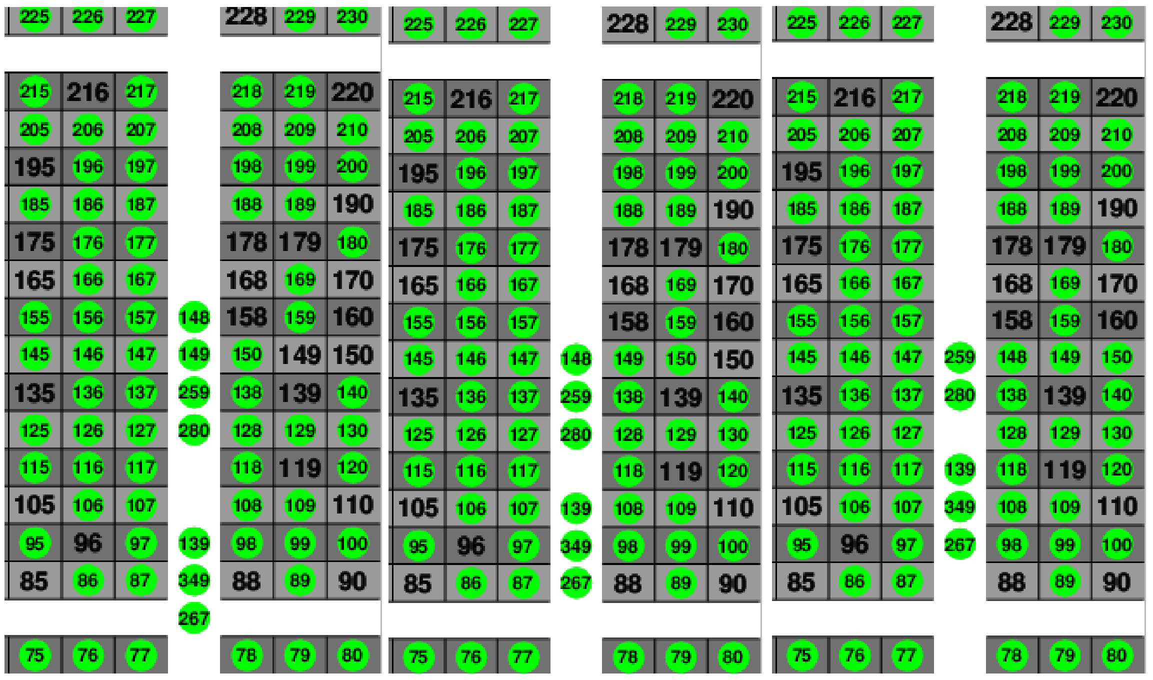

(d) Time step = 325 (e) Time step = 326 (f) Time step = 327

Figure 11: Random boarding the Airbus A320/ Boeing 737. Initiation of a double shuffle by passenger 150.

is reduced by 1 at every time step, until the stow away time reaches 0. Since 5 time steps corresponds to 5 seconds, the passenger stows their luggage for 5 seconds.

Qiang S-J *et al.* 2014 and Landeghem & Beuselinck 2002 both use a bin occupancy model where passengers are randomly assigned 1, 2, or 3 pieces of luggage with percentages of λ, θ, and $1-\lambda-\theta$ respectively. The time required to stow away luggage, in time steps, is then given by

$$t_{stow} = \alpha + \frac{\beta N_p}{[(\gamma+1)-(N_e+N_p)]}$$

where N_e is the number of bags already stowed and N_p is the number of bags carried by the passenger. α represents the minimum time to stow away luggage, β serves as a correction coefficient, and γ is associated with the capacity of the luggage rack. For more information on this model, please refer to Qiang S-J *et al.* 2014 and Landeghem & Beuselinck 2002 [7,10]. Qiang S-J *et al.* 2014 provide a distribution of passengers carrying 1, 2, and 3 bags of luggage aboard the plane. The normal load is given by 60% of passengers carrying 1 bag, 30% carrying 2 bags, and 10% carrying 3 bags. With parameters provided by Landeghem & Beuselinck 2002, we computed the average stow away time for passengers carrying 1 bag, 2 bags, and 3 bags to be approximately 5 sec, 13 sec, and 29 sec respectively. Given the distribution of passengers carrying each number of bags, this equates to an average stow away time of approximately 10 seconds per passenger. Instead of assigning 60% of passengers a stow away time of 5 seconds, 30% of passengers a stow away time of 13 seconds, and 10% of passengers a stow away time of 29 seconds, we incorporated Octave's lognrnd function. The lognrnd function is a function of μ and σ, representing the mean and standard deviation for the normal distribution respectively, which generates an array of random numbers from the lognormal distribution [32]. Specifically, we found that if $\mu = 1.675$, $\sigma = 1.2$, and the maximum time allowed for stowing luggage, $t_{\max}$, is 90 seconds, then the mean stow away time for luggage is approximately 10 seconds. Explicitly, we ran the following function 1000 times and took the mean of the values. We then averaged this mean 100 times to obtain a mean of 10.458 seconds.

$$f(\mu{=}\ 1.675,\ \sigma{=}\ 1.2,\ t_{\max}{=}\ 90) = \text{round}(\min(\text{lognrnd}(1.675,1.2),\ 90))$$

The passengers in the airplane will be assigned a random number generated by this function to determine each of their stow away time. The function rounds the values to make them discrete and applicable to discrete time steps. Reiterating, the maximum amount of time a passenger can use is given by $t_{\max} = 90$. The use of this function and the maximum time allowed to stow luggage are assumptions of this model.

5 Results and Discussion

Given the assumptions and parameters discussed above, the boarding strategies displayed in Figure 12 (a) were simulated for the Airbus A320/ Boeing 737 (see Appendix A). All strategies were previously discussed in detail, with the following exceptions: Front-to-Back, Rotating Zone 2, Steffen, and Ordered Back-to-Front Outside-In Zigzag (OBFOIZ). Front-to-Back boarding was achieved by reversing the boarding order of Back-to-Front boarding. Rotating Zone 2 was achieved by permuting the boarding order of Rotating Zone. Recall the boarding sections for Rotating Zone from Figure 5 (e). Let all seats labelled with a 3 be section 1, let all seats labelled with a 5 be section 2, let all seats labelled with a 6 be section 3, let all seats labelled with a 4 be section 4, and let all seats labelled with a 2 be section 5 (i.e we created an ascending order of sections from front-to-back, partitioned by seat labels). Then, the Rotating Zone boarding strategy boards the following sections in this order: 5, 1, 4, 2, 3. The Rotating Zone 2 boarding strategy boards the sections in this order: 5, 3, 4, 2, 1. This order was chosen in attempt to minimize interferences between passengers.

As previously mentioned, the Steffen boarding strategy is a combination of Back-to-Front and Random Outside-In boarding. Additionally, the Steffen strategy boards the passengers for every other aisle. Recall the layout for the Airbus A320/ Boeing 737 in Figure 3 (a). Then the Steffen boarding strategy boards the passengers 145, 133, 121, ... , 1 onto the plane first, in this specific order. The second group of passengers to board are 150, 138, 126, ... , 6. A list of all groups that board according to the Steffen boarding strategy are listed below. We note that the passengers board in the presented order of groups, starting with group 1, and board in presented order within each group.

1) 145, 133, 121, ... , 25, 13, 1
2) 150, 138, 126, ... , 30, 18, 6
3) 139, 127, 115, ... , 31, 19, 7
4) 144, 132, 120, ... , 36, 24, 12
5) 146, 134, 122, ... , 26, 14, 2
6) 149, 137, 125, ... , 29, 17, 5
7) 140, 128, 116, ... , 32, 20, 8
8) 143, 131, 119, ... , 35, 23, 11
9) 147, 135, 123, ... , 27, 15, 3
10) 148, 136, 124, ... , 28, 16, 4

11) 141, 129, 117, ... , 33, 21, 9

12) 142, 130, 118, ... , 34, 22, 10

The OBFOIZ boarding strategy was constructed in attempt to achieve a faster boarding time than provided by the Steffen boarding strategy. As displayed in Figure 12 (a), this strategy serves as the most efficient boarding strategy for the Airbus A320/ Boeing 737. The OBFOIZ boarding strategy boards according to the following groups, with each group order preserved:

1) 145, 144, 133, ... , 13, 12, 1

2) 150, 139, 138, ... , 18, 7, 6

3) 146, 143, 134, ... , 14, 11, 2

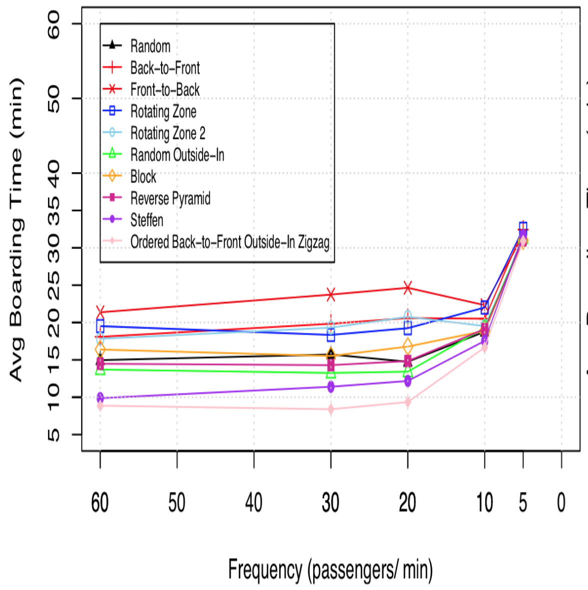
4) 149, 140, 137, ... , 17, 8, 5

5) 147, 142, 135, ... , 15, 10, 3

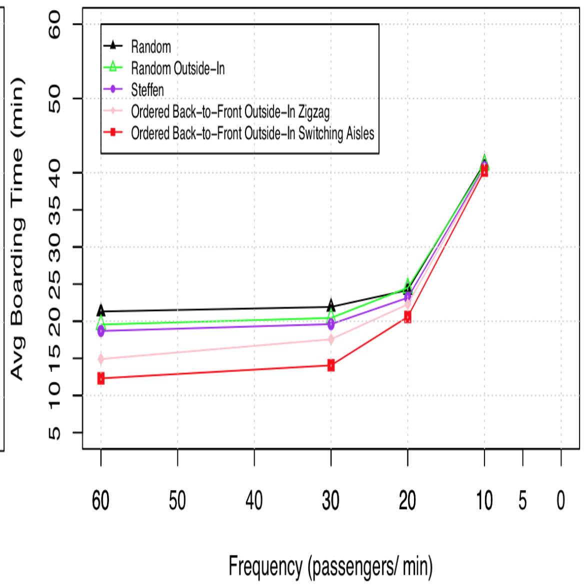
6) 148, 141, 136, ... , 16, 9, 4

Figure 12 (a) shows the average boarding time for all boarding strategies given variable frequencies of passengers entering the plane. In the simulations, the fastest rate of boarding was given by 60 passengers/ min and the slowest rate of boarding was given by 5 passengers/ min. To achieve each average boarding time, the simulation was run 5 times for each boarding strategy at each frequency. Then the average of the 5 runs was taken and plotted as a single point. Since there are 10 boarding strategies with 5 points per strategy, each having 5 runs per point, we have a total of 500 runs constituting the results in Figure 12 (a). Looking at 60 passengers/ min, Figure 12 (a) shows the OBFOIZ boarding strategy to be the most efficient with an average boarding time of 8.87 minutes. The Steffen boarding strategy follows with an average boarding time of 9.87 minutes. The Random Outside-In boarding strategy achieves the third fastest time with an average boarding time of 13.71 minutes. The Reverse Pyramid boarding strategy is not far off with an average boarding time of 14.48 minutes. As expected, the Front-to-Back boarding strategy yields the slowest boarding time since the amount of interferences increased with this boarding strategy.

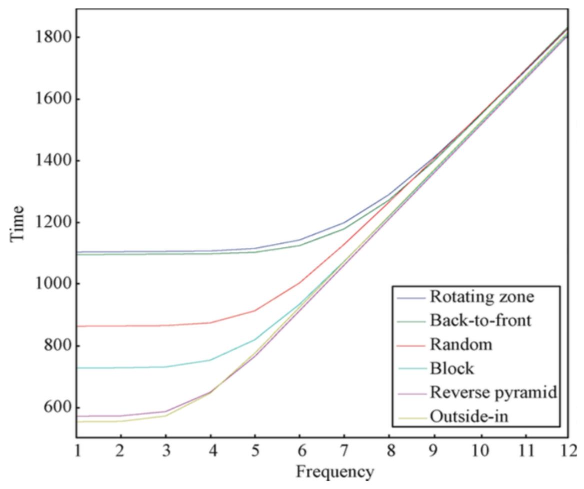
Figure 12 (c) shows the results for Giitsidis & Sirakoulis 2016, graphing the average boarding time (in time steps) against the frequency of passengers entering the Airbus A320/ Boeing 737. To convert the frequency values in Figure 12 (c) to passengers per min, Giitsidis & Sirakoulis 2016 shows that 1 corresponds to $\frac{60}{1} = 60$ passengers/ min. Likewise, a frequency of 2 corresponds to $\frac{60}{2} = 30$ passengers/ min. Giitsidis & Sirakoulis 2016 report the Outside-in (Random Outside-In) boarding strategy as the quickest method of boarding this airplane with a time of approximately 500 time steps, corresponding to approximately 8.33 minutes. Since our model ran the



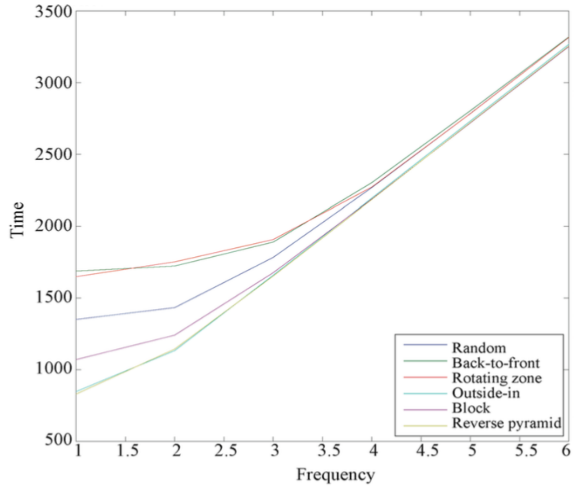
(a) Airbus A320/ Boeing 737



(b) Airbus A380-800/ Boeing 777-300ER



(c) Airbus A320/ Boeing 737



(d) Airbus A380-800/ Boeing 777-300ER

Figure 12: Each image shows the average boarding time of passengers for the frequency of passengers boarding the plane for each boarding strategy. (a): The boarding methods considered in this study for the smaller airplane. (b): The boarding methods considered in this study for the larger airplane. (c) & (d): The boarding methods considered by Giitsidis & Sirakoulis 2016 for the smaller airplane and larger airplane, respectively. Giitsidis & Sirakoulis 2016 show number of time steps on the y-axis against passenger frequency on the x-axis. The average boarding time (in minutes) can be calculated by taking the time step values along the y-axis and dividing by 60 [8].

Random Outside-In boarding strategy in 13.71 minutes, and our boarding times are, on average, higher than Giitsidis & Sirakoulis 2016, we conclude that the OBFOIZ boarding strategy is an improvement with a boarding time of 8.87 minutes. Moreover, the OBFOIZ boarding strategy is an improvement on the Steffen boarding strategy for all boarding frequencies, the current benchmark for efficient boarding [8,29]. We attribute the differences in boarding times, between Giitsidis & Sirakoulis 2016 and this study, for each boarding strategy to the differences in this model's assumptions and interference logic. Specifically, the shuffle, 10% stochastic factor, and log-normal number generator are unique to this study and play an important role in increasing the boarding time for our model.

For the Airbus A320/ Boeing 737, the effect of frequency on the boarding strategies is consistent between these two studies. All boarding strategies show that a boarding rate of 30 passengers/ min does not slow the boarding time by any noticeable amount. In fact, our model shows some strategies show an improvement in boarding time when the rate of passengers is decreased from 60 passengers/ min to 30 passengers/ min. This can be attributed to the number of interferences being less for 30 passengers/ min compared to 60 passengers/ min. The slowest boarding strategies are unaffected by boarding frequency until boarding frequency is decreased to 12 passengers/ min. In contrast, the more efficient boarding methods begin to show sensitivity when boarding frequency is decreased to 15 passengers/ min, indicated by the increasing slope. As the frequency of passengers decreases below 10 passengers/ min, we see a convergence of boarding strategies in both studies.

The Airbus A380-800/ Boeing 777-300ER used boarding strategies comparable to the smaller airplane. The most efficient boarding strategies for the Airbus A320/ Boeing 737 were simulated on the larger airplane, with the Random boarding strategy was also included (see Appendix B). Before analyzing the results obtained for the Airbus A380-800/ Boeing 777-300ER, we need to comment on the implementation of boarding strategies from the smaller airplane. The Random and Random Outside-In boarding strategies are self-explanatory in their sectioning and boarding. The Steffen, OBFOIZ and Ordered Back-to-Front Outside-In Switching Aisles (OBFOISA) boarding strategies have not been defined on the larger airplane. As with the Airbus A320/ Boeing 737, we openly defined OBFOIZ and OBFOISA on the larger airplane. The implemented Steffen strategy on the larger airplane layout is an interpretation in this study, since this strategy was only defined for a single-aisled airplane in the literature [8,29].

Recall the layout for the Airbus A380-800/ Boeing 777-300ER in Figure 3 (b). The OBFOIZ boarding strategy boards according to the following groups, such that the groups board in order. We note that the order of passengers within each group is also preserved when boarding:

1) 391, 390, 371, ... , 30, 11, 10

2) 400, 381, 380, ... , 21, 20, 1
3) 392, 385, 372, ... , 25, 12, 5
4) 395, 382, 375, ... , 22, 15, 2
5) 396, 389, 376, ... , 29, 16, 9
6) 399, 386, 379, ... , 26, 19, 6
7) 393, 384, 373, ... , 24, 13, 4
8) 394, 383, 374, ... , 23, 14, 3
9) 397, 388, 377, ... , 28, 17, 8
10) 398, 387, 378, ... , 27, 18, 7

The OBFOISA boarding strategy boards according to the following groups, such that the groups board in order. We note that the order of passengers within each group is also preserved when boarding:

1) 391, 381, 371, ... , 21, 11, 1
2) 400, 390, 380, ... , 30, 20, 10
3) 392, 382, 372, ... , 22, 12, 2
4) 399, 389, 379, ... , 29, 19, 9
5) 395, 385, 375, ... , 25, 15, 5
6) 396, 386, 376, ... , 26, 16, 6
7) 393, 383, 373, ... , 23, 13, 3
8) 397, 387, 377, ... , 27, 17, 7
9) 394, 384, 374, ... , 24, 14, 4
10) 398, 388, 378, ... , 28, 18, 8

The Steffen boarding strategy boards according to the following groups, such that the groups board in order. We note that the order of passengers within each group is also preserved when boarding:

1) 391, 371, 351, ... , 51, 31, 11
2) 400, 380, 360, ... , 60, 40, 20
3) 381, 361, 341, ... , 41, 21, 1
4) 390, 370, 350, ... , 50, 30, 10

5) 392, 372, 352, ... , 52, 32, 12
6) 395, 375, 355, ... , 55, 35, 15
7) 382, 362, 342, ... , 42, 22, 2
8) 385, 365, 345, ... , 45, 25, 5
9) 396, 376, 356, ... , 56, 36, 16
10) 399, 379, 359, ... , 59, 39, 19
11) 386, 366, 346, ... , 46, 26, 6
12) 389, 369, 349, ... , 49, 29, 9
13) 393, 373, 353, ... , 53, 33, 13
14) 394, 374, 354, ... , 54, 34, 14
15) 383, 363, 343, ... , 43, 23, 3
16) 384, 364, 344, ... , 44, 24, 4
17) 397, 377, 357, ... , 57, 37, 17
18) 398, 378, 358, ... , 58, 38, 18
19) 387, 367, 347, ... , 47, 27, 7
20) 388, 368, 348, ... , 48, 28, 8

Figure 12 (b) shows the average boarding time for all boarding strategies given variable frequencies of passengers entering the Airbus A380-800/ Boeing 777-300ER. In the simulations, the fastest rate of boarding was given by 60 passengers/ min and the slowest rate of boarding was given by 5 passengers/ min. Identical to the smaller airplane, the simulation was run 5 times for each boarding strategy at each frequency to achieve the average boarding time. The average was then plotted as a single point. Since there are 5 boarding strategies with 5 points per strategy, each having 5 runs per point, we have a total of 125 runs to obtain the results in Figure 12 (b).

Across all boarding frequencies, we see that the OBFOISA boarding strategy is the most efficient, followed by the OBFOIZ and Steffen boarding strategies. For 60 passengers/ min, the OBFOISA boarding strategy had an average boarding time of 12.31 min. The OBFOIZ boarding strategy demonstrates efficiency at 60 passengers/ min with an average boarding time of 14.91 min. The Steffen boarding strategy, while slower than both constructed boarding strategies OBFOIZ and OBFOISA, proves to be more efficient than Random Outside-In with an average boarding time of 18.69 min. Random Outside-In is the fourth most efficient boarding strategy with an average boarding time of 19.55 min.

Figure 12 (d) shows the results for Giitsidis & Sirakoulis 2016 graphing the average boarding time (in time steps) against the frequency of passengers entering the Airbus A380-800/ Boeing 777-300ER. The conversion of frequency to passengers per minute is identical to that of the smaller airplane. Giitsidis & Sirakoulis 2016 report the Outside-in (Random Outside-In) and Reverse Pyramid boarding strategies as the quickest methods of boarding this airplane with times of approximately 800 time steps, corresponding to approximately 17.5 minutes. Since our model ran the Random Outside-In boarding strategy in 19.55 minutes, and our boarding times are, on average, higher than Giitsidis & Sirakoulis 2016, we conclude that the OBFOIZ and OBFOISA boarding strategies are an improvement, with OBFOISA demonstrating the greatest efficiency. The Steffen strategy was an improvement on the Random Outside-In boarding strategy, an expected result based on its performance with the single-aisle airplane. Similar to the smaller airplane simulations, we attribute the differences in boarding times, between Giitsidis & Sirakoulis 2016 and this study, for each boarding strategy to the differences in this model's assumptions and interference logic.

Analogous to the Airbus A320/ Boeing 737, the effect of frequency on the boarding strategies was consistent between these two studies. However, unlike the smaller airplane, a boarding rate of 30 passengers/ min slowed the boarding time by a noticeable amount when decreased from 60 passengers/ min. This suggests that the increased size of the airplane layout increased the sensitivity of the boarding time to passenger frequency, shown in Figure 12 (b) and (d). No strategies showed an improvement in boarding time when decreasing the rate of passengers from 60 passengers/ min to any slower rate. This can be explained by the extra aisle space available per passenger, reducing the amount of interferences between passengers. For the Airbus A320/ Boeing 737, we noticed that only the boarding frequencies, with rates around 10-20 passengers/ min, began to show an increase in boarding time. It is likely that this occurred given the increase in available space. Given that the larger airplane naturally has more space, we attribute the increased sensitivity of average boarding times on boarding frequency to the extra space. Unlike the smaller airplane, the Airbus A380-800/ Boeing 777-300ER shows even the slowest boarding strategies are affected by decreasing boarding frequency to 30 passengers/ min. Interestingly, the more efficient boarding methods showed the greatest sensitivity to decreasing boarding frequency, noted by the steeper slopes in Figure 12 (b) and (d) between 60 passengers/ min and 30 passengers/ min. As the frequency of passengers decreases below 20 passengers/ min, we see a convergence of boarding strategies in both studies.

6 Conclusions

In this study, we constructed a CA model that allowed for validation and improvement of existing boarding strategies on the Airbus A320/ Boeing 737 and the Airbus A380-800/ Boeing 777-300ER. Specifically, the OBFOIZ and OBFOISA boarding strategies were defined to give passengers the greatest amount of space and mitigate interferences. These boarding strategies proved to be the most efficient boarding methods among all strategies considered for both airplanes. The OBFOISA boarding strategy was constructed such that the passengers would utilize available space in both aisles, to further mitigate interferences. The results also suggest that passenger frequency plays an important role in the determining the boarding time of a strategy. For the Airbus A320/ Boeing 737, the Rotating Zone 2, Block, Reverse Pyramid, Random Outside-In, and OBFOIZ boarding methods saw a decrease in boarding time when 60 passengers/ min was reduced to 30 passengers/ min. The Airbus A380-800/ Boeing 777-300ER did not show this response for any decrease in boarding frequency. This supports the explanation that the size of the layout and the amount of aisle space, relative to the number of seats, determine sensitivity to passenger boarding frequency. Given the positive results obtained in defining new boarding strategies for the Airbus A380-800/ Boeing 777-300ER, future studies can redefine existing boarding strategies, across the aisles, and develop new boarding strategies to improve boarding times. In addition, machine learning algorithms can be implemented to achieve quicker boarding results, leaving decision on “path choice” to the passengers.

7 Citations

1. Dementieva Y, Helkey D R. *Efficiently Boarding an Airplane: A Modeling Based Approach.* Emmanuel College Department of Mathematics, 2013.

2. Another Strong Year for Airline Profits in 2017 [Online]. Available at: http://www.iata.org/pressroom/pr/Pages/2016-12-08-01.aspx, December, 2016.

3. Aviation Economic Benefits [Online]. Available at: https://www.iata.org/publications/economics/Reports/ 890700-aviation-economic-benefits-summary-report.pdf, July, 2007.

4. Strong Airline Profitability Continues in 2018 [Online]. Available at: http://www.iata.org/pressroom/pr/Pages/2017-12-05-01.aspx, December, 2017.

5. Steiner M P, Philipp M. Speeding up the airplane boarding process by using pre-boarding areas. In: Proceedings of the 9th Swiss Transport Research Conference. SVWG, Ascona, Switzerland, 2009, 1—30

6. Mas S, Juan A A, Arias P, Fonseca P. *A Simulation Study Regarding Different Aircraft Boarding Strategies.* Springer-Verlag Berlin Heidelberg, Heidelberg, Germany, (Eds.) MS 2013, LNBIP 145, 2013.

7. Qiang S-J, Jia B, Xie D-F, Gao Z-Y. Reducing Airplane Boarding Time by Accounting for Passengers' Individual Properties: A Simulation Based on Cellular Automaton. *Journal of Air Transport Management*, 2014, 40: 42—47 Available at: https://www.sciencedirect.com/science/article/abs/pii/ S096969971400074X

8. Giitsidis T, Sirakoulis G Ch. Modeling Passengers Boarding in Aircraft Using Cellular Automata. *IEEE/CAA Journal of Automatica Sinica*, 2016, 3(4): 365—384

9. Marelli S, Mattocks G, Merry R. The role of computer simulation in reducing airplane turn time. AERO Magazine, 1998, 1

10. Landeghem H V, Beuselinck A. Reducing Passenger Boarding Time in Airplanes: A Simulation Based Approach. *European Journal of Operational Research*, 2002, 142: 294—308

11. Menkes H. Airplane boarding: how to board a 150 seat airplane in less than 15 minutes? [Online], Available: http://menkes76.com/projects/boarding/ boarding.htm, April, 2013.

12. Helbing D, Farkas I, Vicsek T. Simulating dynamical features of escape panic. *Nature*, 2000, 407(6803): 487—490

13. Bandini S, Manzoni S, Vizzari G. Situated Cellular Agents: A Model to Simulate Crowding Dynamics. *IEICE TRANSACTIONS on Information and Systems*, 2004, E87-D(3): 669—676

14. Michael B. *Cities and Complexity: Understanding Cities with Cellular Automata, Agent-Based Models, and Fractals.* The MIT Press, Cambridge, MA, USA, 2017.

15. Kohler T A, *Putting Social Sciences Together Again: An Introduction to the Volume.* Oxford University Press, New York, NY, USA, 2000.

16. Chopard B. *Cellular Automata Modeling of Physical Systems.* In: Meyers R. (Eds.) Encyclopedia of Complexity and Systems Science. Springer, New York, NY, USA, 2009.

17. Wolfram S. *A new kind of science.* Wolfram Media Inc., 2002.

18. Margolus N, Toffoli T. Cellular Automata Machines. *Complex Systems*, 1987, 1: 967—993

19. Kapral R, Showalter K. 1995. *Chemical waves and patterns.* Springer Science & Business Media, Dordrecht, Netherlands, 1995.

20. Wu-Pong S, Cheng C-K. Pharmacokinetic Simulations Using Cellular Automata in a Pharmacokinetics Course. *American Journal of Pharmaceutical Education*, 1999, 63(1): 52—55

21. Burks A W. *Von Neumann's Self-Reproducing Automata.* Publishers Clearing House, Springfield, VA, USA, June, 1969.

22. Von Neumann J, Burks A W. *Theory of Self-reproducing Automata.* Champaign, IL, USA: University of Illinois Press, 1966.

23. SeatGuru - Emiratas Seat Maps [Online].
Available at: https://www.seatguru.com/airlines/Emirates_Airlines/, April, 2018.

24. Boarding Airbus A380 (Lufthansa) [Online].
Available at: https://www.youtube.com/watch?v=Q3x2UJ-rDPM, July, 2011.

25. Boarding Emirates A380 [Online].
Available at: https://www.youtube.com/watch?v=JZs_dKihLEU&t=173s, March, 2013.

26. Boarding Lufthansa's Airbus A380-800 [Online].
Available at: https://www.youtube.com/watch?v=cjg9483nSBQ, March, 2018.

27. Qatar Airways Boeing 777-300 Boarding, Taxi and Takeoff from Hamad Int. airport [Online].
Available at: https://www.youtube.com/watch?v=tTom6YQwNH4, December, 2017.

28. Milne J R, Kelly A R. A new method for boarding passengers onto an airplane [Online]. *Journal of Air Transport Management*, 2014, 34: 93—100
Available at: http://dx.doi.org/10.1016/j.jairtraman.2013.08.006

29. Steffen J H. Optimal boarding method for airline passengers. *Journal of Air Transport Management*, 2008, 14(3): 146—150

30. Wallace R. The Flying Carpet [Online].
Available at: http://roundpegin.com/innovations/aviation/aircraft-boarding-the-flying-carpet/, November, 2013.

31. Richards G. Planes parallel [Online].
Available at: https://commons.wikimedia.org/wiki/File:Planes_parallel.svg, July, 2007.

32. MathWorks [Online].
Available at: https://www.mathworks.com/help/, April, 2018.

8 Appendix A: Airbus A320/ Boeing 737 Passenger Boarding

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%% AIRBUS A320/ BOEING 737 PASSENGER BOARDING %%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% construct matrix to hold averages for each run
mat = zeros(1,5);

% for loop for each of the 5 runs
for major = 1:5

close()
clear -x mat major
clc
hold off
more off



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INITIALIZE GLOBAL VARIABLES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

numofrows = 25; % 10 and then 14, 2 - 3 - 2
numofcolumns = 7; % num of columns of seats (including aisles)
aisles = 1; % walking aisles to find seat on plane



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CONSTRUCT IMAGE MATRICES OF AIRPLANE SEATING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

matrix = repmat([1,1,1,0,1,1,1;1.4,1.4,1.4,0,1.4,1.4,1.4],12,1);

matrix = [matrix; [1,1,1,0,1,1,1]; [0,0,0,0,0,0,0]; [0,0,0,0,0,0,0]];
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PRODUCING THE IMAGE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

colourmin = min(min(matrix));
colourmax = max(max(matrix));
colourrange = [ colourmin 1.8*colourmax ];

% colourbar scaled image
imagesc(matrix, colourrange);
colormap(flipud(gray))
axis xy

% ranges of graph
xlim([0.5,numofcolumns + 0.5])
ylim([0.5,numofrows + 2.5])


set(gcf(),'paperunits','normalized', ...
'paperposition',[0,0,0.4,0.7]);

set(gca(), ...
'Box' , 'off' , ...
'TickDir' , 'out' , ...
'TickLength' , [.02 .02] , ...
'XMinorTick' , 'off' , ...
'YMinorTick' , 'on' , ...
'YGrid' , 'off' , ... % dotted grid system
'XGrid' , 'off' , ...
'XColor' , [.1 .1 .1], ... % faded axes grayscale
'YColor' , [.1 .1 .1], ...
'LineWidth' , 1 , ...
'Fontsize' , 11)
title('Passenger Boarding' , ...
'FontSize',18 , ...
'FontWeight','bold' , ...
'Color','black' , ...
'FontName' , ...
'Garamond')
xlabel('Airplane Width' , ...
'FontSize',15 , ...
'FontWeight','bold', ...
'Color','black' , ...
'FontName','Garamond')
ylabel('Airplane Length' , ...
'FontSize',15 , ...
'FontWeight','bold', ...
'Color','black' , ...
'FontName','Garamond')



%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PLOT ROW AND AISLE LINES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%

y = [0.5, 25.5];

x=[1.5,1.5];
line(x,y,'color','black','linewidth',1.5)

x=[2.5,2.5];
line(x,y,'color','black','linewidth',1.5)

x=[5.5,5.5];
line(x,y,'color','black','linewidth',1.5)

x=[6.5,6.5];
line(x,y,'color','black','linewidth',1.5)


x = [0,3.49];
x2 = [4.507,7.495];
ctr = 1.5;

while ctr < numofrows+1
y = [ctr, ctr];
line(x,y,'color','black','linewidth',1.5)
line(x2,y,'color','black','linewidth',1.5)
ctr = ctr + 1;
end

clear -x numofcolumns numofrows aisles mat major
```

```
%saveas(gcf(),strcat('backgroundfigure.png')); %produce each image



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INITIALIZE PEOPLE AND ATTRIBUTES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

numofpassengers = (numofcolumns - aisles)*(numofrows);
numofattributes = 4;

%4th attribute: time taken to put away luggage based on log normal
    ↪ distribution

%Boarding Testing
%perm = 1:numofpassengers;
%perm = [[3,2,1,4],[5:numofpassengers] ];
%perm = numofpassengers:-1:1;


%Random Boarding
%perm = randperm(numofpassengers);

%Back-to-Front Boarding
%{
section1 = [1:30];
n=numel(section1);
ii=randperm(n);
[~,previous_order]=sort(ii);
section1perm=section1(ii);

section2 = [31:60];
n=numel(section2);
ii=randperm(n);
[~,previous_order]=sort(ii);
section2perm=section2(ii);

section3 = [61:90];
n=numel(section3);
ii=randperm(n);
[~,previous_order]=sort(ii);
section3perm=section3(ii);
```

```
section4 = [91:120];
n=numel(section4);
ii=randperm(n);
[~,previous_order]=sort(ii);
section4perm=section4(ii);

section5 = [121:150];
n=numel(section5);
ii=randperm(n);
[~,previous_order]=sort(ii);
section5perm=section5(ii);
%}
%front to Back
%perm = [section1perm, section2perm, section3perm, section4perm,
    ↪ section5perm];

%back to front
%perm = [section5perm, section4perm, section3perm, section2perm,
    ↪ section1perm];

%rotating zone
%perm = [section5perm, section1perm, section4perm, section2perm,
    ↪ section3perm];

%rotating zone 2
%perm = [section5perm, section3perm, section4perm, section2perm,
    ↪ section1perm];

%Random Outside-In
%{
section1 = [[1:6:145],[6:6:150]];
n=numel(section1);
ii=randperm(n);
[~,previous_order]=sort(ii);
section1perm=section1(ii);

section2 = [[2:6:146],[5:6:149]];
n=numel(section2);
ii=randperm(n);
[~,previous_order]=sort(ii);
section2perm=section2(ii);
```

```
section3 = [[3:6:147],[4:6:148]];
n=numel(section3);
ii=randperm(n);
[~,previous_order]=sort(ii);
section3perm=section3(ii);

perm = [section1perm, section2perm, section3perm];
%}

%Block Boarding Strategy
%{
section1 = [[103:6:145],[108:6:150]];
n=numel(section1);
ii=randperm(n);
[~,previous_order]=sort(ii);
section1perm=section1(ii);

section2 = [[104:6:146],[107:6:149],[105:6:147],[106:6:148]];
n=numel(section2);
ii=randperm(n);
[~,previous_order]=sort(ii);
section2perm=section2(ii);

section3 = [[55:6:97],[60:6:102]];
n=numel(section3);
ii=randperm(n);
[~,previous_order]=sort(ii);
section3perm=section3(ii);

section4 = [[56:6:98],[59:6:101],[58:6:100],[57:6:99]];
n=numel(section4);
ii=randperm(n);
[~,previous_order]=sort(ii);
section4perm=section4(ii);

section5 = [[1:6:49],[6:6:54]];
n=numel(section5);
ii=randperm(n);
[~,previous_order]=sort(ii);
section5perm=section5(ii);
```

```
section6 = [[2:6:50],[3:6:51],[4:6:52],[5:6:53]];
n=numel(section6);
ii=randperm(n);
[~,previous_order]=sort(ii);
section6perm=section6(ii);

perm = [section1perm, section2perm, section3perm,
section4perm, section5perm, section6perm];
%}

%Pyramid Boarding Strategy
%{
section1 = [[67:6:145],[72:6:150]];
n=numel(section1);
ii=randperm(n);
[~,previous_order]=sort(ii);
section1perm=section1(ii);

section2 = [[37:6:61],[42:6:66],[98:6:146],[101:6:149]];
n=numel(section2);
ii=randperm(n);
[~,previous_order]=sort(ii);
section2perm=section2(ii);

section3 = [[41:6:95],[38:6:92],[6:6:36],[1:6:31]];
n=numel(section3);
ii=randperm(n);
[~,previous_order]=sort(ii);
section3perm=section3(ii);

section4 = [[2:6:32],[5:6:35],[93:6:147],[94:6:148]];
n=numel(section4);
ii=randperm(n);
[~,previous_order]=sort(ii);
section4perm=section4(ii);

section5 = [[3:6:87],[4:6:88]];
n=numel(section5);
ii=randperm(n);
[~,previous_order]=sort(ii);
section5perm=section5(ii);
```

```
perm = [section1perm, section2perm, section3perm, section4perm,
   ↪ section5perm];
%}

%Steffen Boarding Strategy
%{
section1 = [145:-12:1];
section2 = [150:-12:6];
section3 = [139:-12:7];
section4 = [144:-12:12];
section5 = [146:-12:2];
section6 = [149:-12:5];
section7 = [140:-12:8];
section8 = [143:-12:11];
section9 = [147:-12:3];
section10 = [148:-12:4];
section11 = [141:-12:9];
section12 = [142:-12:10];


perm = [section1, section2, section3, section4,
section5, section6, section7, section8,
section9, section10,section11,section12];
%}

%Ordered Back-to-Front Outside-In Zigzag

section1 = [[145:-12:1],[144:-12:12]];
section1 = fliplr(sort(section1));

section2 = [[150:-12:6],[139:-12:7]];
section2 = fliplr(sort(section2));

section3 = [[146:-12:2],[143:-12:11]];
section3 = fliplr(sort(section3));

section4 = [[149:-12:5],[140:-12:8]];
section4 = fliplr(sort(section4));

section5 = [[147:-12:3],[142:-12:10]];
section5 = fliplr(sort(section5));
```

```
section6 = [[148:-12:4],[141:-12:9]];
section6 = fliplr(sort(section6));

perm = [section1,section2,section3,section4,section5,section6];



person=zeros(numofattributes, numofpassengers,'single');

for i=1:numofpassengers

person(1,i) = perm(i);
person(2,i) = ceil(person(1,i)/(numofcolumns - aisles));
person(3,i) = rem(person(1,i),(numofcolumns - aisles));
person(4,i) = min(round(lognrnd(1.675,1.2)),90);

%Last row is the time it takes to put away suitcase.
%Numbers are generated bewteen 1 and 90. Numbers generated are skewed
    ↪ towards 0.

if person(3,i) >= 4 %center aisle is column 4
person(3,i) = person(3,i) + 1;
end

if person(3,i) == 0
person(3,i) = numofcolumns;
end
end

%person
clear i perm numofpassengers aisles emergencyrows



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INITIALIZE POINTS FOR PLOTTING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hold on

p=zeros(numofrows+2,numofcolumns,3);
```

```
draw_ctr = 1;

for iii = 1:3
for i = 1:numofrows+2
for ii = 1:numofcolumns
if iii == 1
if ii == 4

%nothing

else

%plot the ordered seat number of the seat

p(i,ii,iii) = text(ii-0.01, i-0.015 , ...
[num2str(draw_ctr)] , ...
'color','black' , ...
'fontsize',8 , ...
'horizontalalignment','center' , ...
'fontweight','bold');

draw_ctr = draw_ctr + 1;
end

elseif iii == 2

%plot the green point to represent the person

p(i,ii,iii) = plot(ii,i , ...
'marker','o' , ...
'markersize',12 , ...
'markeredgecolor','green' , ...
'markerfacecolor','green');
else

%plot the ordered seat number of the person

p(i,ii,iii) = text(ii-0.01, i-0.015 , ...
['X'] , ...
'color','black' , ...
'fontsize',7 , ...
'horizontalalignment','center' , ...
'fontweight','bold');
end
end
end
end

%turn off all plotted points
set(p(:,:,2:3),'visible','off');

%turn off all plotted points
set(p(numofrows+1:numofrows+2,:,1),'visible','off');

%produce each image
%saveas(gcf(),strcat('AirplaneLayout3-3.png'));

clear i ii iii draw_ctr



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INITIALIZE MATRIX LANDSCAPE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

totalsteps=3000;

steps=zeros(numofrows+4,numofcolumns,totalsteps,numofattributes,'uint8
    ↪ ');

steps(1,4,1,:) = person(:,1);
set(p(1,4,2),'visible','on');
set(p(1,4,3),'visible','on','string',num2str(steps(1,4,1,1)));
set(p(1,4,1),'visible','off');


%saveas(gcf(),strcat('3-3OrderedSeatNumBoarding',num2str(1),'.png'));
%capture first image

person_ctr = 1;
person(:,person_ctr) = 0;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% BEGIN BOARDING AND LOGIC %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%
% Row Movement
%%%%%%%%%%%%%%%%

for time = 2:totalsteps;

for back = numofrows+2:-1:1

% Initialize column variables
a1 = 1;
a3 = 3;
b1 = 7;
b3 = 5;

acol = 4;


for col=a1:a3

if (steps(back,col,time-1,1) ~= 0) && (steps(back,col,time-1,2) ==
    ↪ back)

if (steps(back,col,time-1,3) < col)
if steps(back,col-1,time,1) == 0

if steps(back,acol,time-1,2) ~= back

steps(back,col-1,time,:) = steps(back,col,time-1,:);
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

elseif (steps(back,acol,time-1,3) > max(steps(back,a1:a3,time-1,3)))
```

```
steps(back,col-1,time,:) = steps(back,col,time-1,:);
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

else %when steps(back,col,time-1,3) = col
steps(back,col,time,:) = steps(back,col,time-1,:); % stay fixed in
    ↪ correct seat

end
end%if

end%for col

clear col

for col=b1:-1:b3

if (steps(back,col,time-1,1) ~= 0) && (steps(back,col,time-1,2) ==
    ↪ back)

if (steps(back,col,time-1,3) > col)
if steps(back,col+1,time,1) == 0

if sum(steps(back,b3:b1,time-1,1)) > 0
min_var = min(steps(back,b3:b1,time-1,3)(steps(back,b3:b1,time-1,3)>0)
    ↪ );
else
min_var = 0;
end
```

```
if steps(back,acol,time-1,2) ~= back

steps(back,col+1,time,:) = steps(back,col,time-1,:);
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

elseif min_var ~= 0 && steps(back,acol,time-1,3) < min_var

steps(back,col+1,time,:) = steps(back,col,time-1,:);
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

else %when steps(back,col,time-1,3) = col
steps(back,col,time,:) = steps(back,col,time-1,:); % stay fixed in
    ↪ correct seat
end


end %if

end %for col

clear a1 a3 b1 b3 min_var
```

```
%%%%%%%%%%%%%%%%%%%
% Aisle Movement
%%%%%%%%%%%%%%%%%%%

for col = acol

slow = rand;
if (slow < 0.10 && steps(back,col,time-1,2) ~= back)
% 0.nm = nm% of the time person in aisle will not move

steps(back,col,time,:) = steps(back,col,time-1,:); %stay in position

else
if (steps(back,col,time-1,1) ~= 0)

if steps(back,col,time-1,2) > back
if steps(back+1,col,time,1) == 0
steps(back+1,col,time,:) = steps(back,col,time-1,:);
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

elseif steps(back,col,time-1,2) == back


%Time it takes to put away suitcase.
%If the 4th parameter for each person is not zero yet, then
%count down this number by 1 per step until zero is reached
%and then the person will sit in their seat on the next step.

if steps(back,col,time-1,4) ~= 0
steps(back,col,time,:) = steps(back,col,time-1,:);
steps(back,col,time,4) = steps(back,col,time-1,4) - 1;
```

```
else

if (steps(back,col,time-1,3) < col)

if (steps(back,col,time-1,3) > max(steps(back,col-3:col-1,time,3)))
if steps(back,col-1,time,1) == 0
steps(back,col-1,time,:) = steps(back,col,time-1,:);
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
end
else %shuffling

if steps(back,col-2,time,1) ~= 0
if steps(back,col-1,time,1) == 0

steps(back,col-1,time,:) = steps(back,col-2,time,:);
steps(back,col-2,time,:) = 0;
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col-2,2:3),'visible','off');
set(p(back,col-2,1),'visible','on');

elseif steps(back+1,col,time,1) == 0 %if both seats occcupied then
    ↪ move both

steps(back+1,col,time,:) = steps(back,col-1,time,:);
steps(back,col-1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col-1,2:3),'visible','off');
set(p(back,col-1,1),'visible','on');
```

```
steps(back,col-1,time,:) = steps(back,col-2,time,:);
steps(back,col-2,time,:) = 0;
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col-2,2:3),'visible','off');
set(p(back,col-2,1),'visible','on');

end

elseif steps(back,col-1,time,1) ~= 0

if steps(back+1,col,time,1) == 0

steps(back+1,col,time,:) = steps(back,col-1,time,:);
steps(back,col-1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col-1,2:3),'visible','off');
set(p(back,col-1,1),'visible','on');

elseif steps(back+1,col,time,2) == back
%is back+1 from the previous row?

if steps(back+2,col,time,1) == 0
steps(back+2,col,time,:) = steps(back+1,col,time,:);
steps(back+1,col,time,:) = 0;
set(p(back+2,col,2),'visible','on');
set(p(back+2,col,1),'visible','off');

set(p(back+2,col,3),'visible','on','string',num2str(steps(back+2,col,
    ↪ time,1)));
set(p(back+1,col,2:3),'visible','off');
set(p(back+1,col,1),'visible','on');

steps(back+1,col,time,:) = steps(back,col-1,time,:);
steps(back,col-1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col-1,2:3),'visible','off');
set(p(back,col-1,1),'visible','on');
end
end
end

steps(back,col,time,:) = steps(back,col,time-1,:);

end %shuffling


else %column is greater than aisle and do the following


if sum(steps(back,col+1:col+2,time,1)) > 0
min_var = min(steps(back,col+1:col+2,time,3)(steps(back,col+1:col+2,
    ↪ time,3)>0));
else
min_var = 0;
end


if min_var ~= 0 && steps(back,col,time-1,3) < min_var

if steps(back,col+1,time,1) == 0
steps(back,col+1,time,:) = steps(back,col,time-1,:);
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

end
```

```
elseif min_var == 0

if steps(back,col+1,time,1) == 0

steps(back,col+1,time,:) = steps(back,col,time-1,:);
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

end

elseif steps(back,col+2,time,1) ~= 0 %shuffling

if steps(back,col+1,time,1) == 0

steps(back,col+1,time,:) = steps(back,col+2,time,:);
steps(back,col+2,time,:) = 0;
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col+2,2:3),'visible','off');
set(p(back,col+2,1),'visible','on');

elseif steps(back+1,col,time,1) == 0

%if both seats occcupied then move both

steps(back+1,col,time,:) = steps(back,col+1,time,:);
steps(back,col+1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col+1,2:3),'visible','off');
set(p(back,col+1,1),'visible','on');
```

```
steps(back,col+1,time,:) = steps(back,col+2,time,:);
steps(back,col+2,time,:) = 0;
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
   ↪ time,1)));
set(p(back,col+2,2:3),'visible','off');
set(p(back,col+2,1),'visible','on');

end

steps(back,col,time,:) = steps(back,col,time-1,:);

else %if steps(back,col+1,time,1) ~= 0

if steps(back+1,col,time,1) == 0

steps(back+1,col,time,:) = steps(back,col+1,time,:);
steps(back,col+1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
   ↪ time,1)));
set(p(back,col+1,2:3),'visible','off');
set(p(back,col+1,1),'visible','on');

elseif steps(back+1,col,time,2) == back
%is back+1 from the previous row?

if steps(back+2,col,time,1) == 0

steps(back+2,col,time,:) = steps(back+1,col,time,:);
steps(back+1,col,time,:) = 0;
set(p(back+2,col,2),'visible','on');
set(p(back+2,col,1),'visible','off');

set(p(back+2,col,3),'visible','on','string',num2str(steps(back+2,col,
   ↪ time,1)));
set(p(back+1,col,2:3),'visible','off');
set(p(back+1,col,1),'visible','on');

steps(back+1,col,time,:) = steps(back,col+1,time,:);
steps(back,col+1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col+1,2:3),'visible','off');
set(p(back,col+1,1),'visible','on');
end
end

steps(back,col,time,:) = steps(back,col,time-1,:);

end %shuffling

end
end %storing luggage overhead bin

else % < back

steps(back,col,time,:) = steps(back,col,time-1,:);

end %elseif relay

if steps(back+1,col,time,2) == back
if steps(back,col,time,1) == 0

steps(back,col,time,:) = steps(back+1,col,time,:);
steps(back+1,col,time,:) = 0;
set(p(back,col,2),'visible','on');
set(p(back,col,1),'visible','off');

set(p(back,col,3),'visible','on','string',num2str(steps(back,col,time
    ↪ ,1)));
set(p(back+1,col,2:3),'visible','off');
set(p(back+1,col,1),'visible','on');

end
end
```

```
if steps(back+2,col,time,2) == back
if steps(back+1,col,time,1) == 0

steps(back+1,col,time,:) = steps(back+2,col,time,:);
steps(back+2,col,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back+2,col,2:3),'visible','off');
set(p(back+2,col,1),'visible','on');

end
end

end %if steps ~= 0

end %slow
end %for col = acol
end %for back



%%%%%%%%%%%%%%
% New Person
%%%%%%%%%%%%%%
if mod(time,12) == 1 %passenger boarding frequency

if (person_ctr < length(person)) && (steps(1,4,time,1) == 0)
person_ctr = person_ctr + 1;
steps(1,4,time,:) = person(:,person_ctr);
person(:,person_ctr) = 0;
set(p(1,4,2),'visible','on');
set(p(1,4,3),'visible','on','string',num2str(steps(1,4,time,1)));
set(p(1,4,1),'visible','off');
end

end
```

```
%%%%%%%%%%%%%%%%%
% Produce Image
%%%%%%%%%%%%%%%%%

%saveas(gcf(),strcat('3-3OrderedSeatNumBoarding',num2str(time),'.png')
   ↪ );



%%%%%%%%%%%%%%
% Stop Logic
%%%%%%%%%%%%%%


if (sum(person(1,:)) == 0) && ((sum(steps(:,4,time,1)) == 0
&& sum(steps(numofrows+2,:,time,1)) == 0))
break;
end



%%%%%%%%%%%%%%%
% Break Test
%%%%%%%%%%%%%%%


%{
if time == 15
break;
end
%}


end %for time



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% END BOARDING AND LOGIC %%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%produce final image
```

```
%saveas(gcf(),strcat('3-3RandomBoardingFinalIm',num2str(time),'.png'))
    ↪ ;

printf("Simulation required %i steps.\n",time);

mat(major) = time;

end %major

mat
```

9 Appendix B: Airbus A380-800/ Boeing 777-300ER Passenger Boarding

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% AIRBUS A380-800/ BOEING 777-300ER PASSENGER BOARDING %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% construct matrix to hold averages for each run
mat = zeros(1,5);

% for loop for each of the 5 runs
for major = 1:5

close()
clear -x mat major
clc
hold off
more off



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INITIALIZE GLOBAL VARIABLES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% 8 then 14 then 13 then 5 rows, 3-4-3
numofrows = 46;

% num of columns of seats (including aisles)
numofcolumns = 12;

% walking aisles to find seat on plane
aisles = 2;
emergencyrows = 6;

left_aisle = 4;
right_aisle = 9;
low_emerg_row = 10;
mid_emerg_row = 25;
high_emerg_row = 39;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CONSTRUCT IMAGE MATRICES OF AIRPLANE SEATING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

matrix0 = [0,0,0,0,0,0,0,0,0,0,0,0];

matrix8 = repmat([1,1,1,0,1,1,1,1,0,1,1,1;
1.4,1.4,1.4,0,1.4,1.4,1.4,1.4,0,1.4,1.4,1.4],4,1);

matrix14 = repmat([1,1,1,0,1,1,1,1,0,1,1,1;
1.4,1.4,1.4,0,1.4,1.4,1.4,1.4,0,1.4,1.4,1.4],7,1);

matrix12 = repmat([1,1,1,0,1,1,1,1,0,1,1,1;
1.4,1.4,1.4,0,1.4,1.4,1.4,1.4,0,1.4,1.4,1.4],6,1);

matrix4 = repmat([1,1,1,0,1,1,1,1,0,1,1,1;
1.4,1.4,1.4,0,1.4,1.4,1.4,1.4,0,1.4,1.4,1.4],2,1);

matrixadd = [1,1,1,0,1,1,1,1,0,1,1,1];

matrix = [matrix0; matrix8; matrix0; matrix14; matrix0;
matrix12; matrixadd; matrix0; matrix4; matrixadd];



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PRODUCING THE IMAGE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

colourmin = min(min(matrix));
colourmax = max(max(matrix));
colourrange = [ colourmin 1.8*colourmax ];

% colourbar scaled image
imagesc(matrix, colourrange);
colormap(flipud(gray))
axis xy

% ranges of graph
```

```
xlim([0.5,numofcolumns + 0.5])
ylim([0.5,numofrows+0.5])


set(gcf(),'paperunits','normalized', ...
'paperposition',[0,0,0.4,0.7]);

set(gca(), ...
'Box' , 'off' , ...
'TickDir' , 'out' , ...
'TickLength' , [.02 .02] , ...
'XMinorTick' , 'off' , ...
'YMinorTick' , 'on' , ...
'YGrid' , 'off' , ... % dotted grid system
'XGrid' , 'off' , ...
'XColor' , [.1 .1 .1], ... % faded axes grayscale
'YColor' , [.1 .1 .1], ...
'LineWidth' , 1 , ...
'Fontsize' , 11)
title('Passenger Boarding' , ...
'FontSize',18 , ...
'FontWeight','bold' , ...
'Color','black' , ...
'FontName' , ...
'Garamond')
xlabel('Airplane Width' , ...
'FontSize',15 , ...
'FontWeight','bold', ...
'Color','black' , ...
'FontName','Garamond')
ylabel('Airplane Length' , ...
'FontSize',15 , ...
'FontWeight','bold', ...
'Color','black' , ...
'FontName','Garamond')



%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% PLOT ROW AND AISLE LINES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
for section = 1:4
if section == 1
y = [1.5, low_emerg_row-0.5];
elseif section == 2
y = [low_emerg_row+0.5, mid_emerg_row-0.5];
elseif section == 3
y = [mid_emerg_row+0.5, high_emerg_row-0.5];
else
y = [high_emerg_row+0.5, numofrows-1.5];
end


x=[1.5,1.5];
line(x,y,'color','black','linewidth',1.5)

x=[2.5,2.5];
line(x,y,'color','black','linewidth',1.5)

x=[5.5,5.5];
line(x,y,'color','black','linewidth',1.5)

x=[6.5,6.5];
line(x,y,'color','black','linewidth',1.5)

x=[7.5,7.5];
line(x,y,'color','black','linewidth',1.5)

x=[10.5,10.5];
line(x,y,'color','black','linewidth',1.5)

x=[11.5,11.5];
line(x,y,'color','black','linewidth',1.5)

end

x = [0,3.495];
x2 = [4.505,8.495];
x3 = [9.505,12.495];
ctr = 1.5;

while ctr < numofrows-1
y = [ctr, ctr];
```

```
line(x,y,'color','black','linewidth',1.5)
line(x2,y,'color','black','linewidth',1.5)
line(x3,y,'color','black','linewidth',1.5)
ctr = ctr + 1;
end

clear -x numofcolumns numofrows aisles emergencyrows low_emerg_row
    ↪ mid_emerg_row high_emerg_row left_aisle right_aisle mat major

%saveas(gcf(),strcat('backgroundfigurePVMTEST.png')); %produce each
    ↪ image



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INITIALIZE PEOPLE AND ATTRIBUTES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

numofpassengers = (numofcolumns - aisles)*(numofrows - emergencyrows);
numofattributes = 4;

%4th attribute: time taken to put away luggage based on log normal
    ↪ distribution

%Boarding Testing
%perm = 1:numofpassengers;
%perm = [[3,2,1,4],[5:numofpassengers] ];
%perm = numofpassengers:-1:1;

%Random Boarding
%perm = randperm(numofpassengers);

%Random Outside-In
%{
section1 = [[1:10:391],[10:10:400]];
n=numel(section1);
ii=randperm(n);
[~,previous_order]=sort(ii);
section1perm=section1(ii);

section2 = [[2:10:392],[9:10:399],[5:10:395],[6:10:396]];
n=numel(section2);
```

```
ii=randperm(n);
[~,previous_order]=sort(ii);
section2perm=section2(ii);

section3 = [[3:10:393],[4:10:394],[7:10:397],[8:10:398]];
n=numel(section3);
ii=randperm(n);
[~,previous_order]=sort(ii);
section3perm=section3(ii);

perm = [section1perm, section2perm, section3perm];
%}


%Steffen Boarding Strategy
%{
section1 = [391:-20:11];
section2 = [400:-20:20];
section3 = [381:-20:1];
section4 = [390:-20:10];

section5 = [392:-20:12];
section6 = [395:-20:15];
section7 = [382:-20:2];
section8 = [385:-20:5];

section9 = [396:-20:16];
section10 = [399:-20:19];
section11 = [386:-20:6];
section12 = [389:-20:9];

section13 = [393:-20:13];
section14 = [394:-20:14];
section15 = [383:-20:3];
section16 = [384:-20:4];

section17 = [397:-20:17];
section18 = [398:-20:18];
section19 = [387:-20:7];
section20 = [388:-20:8];

perm = [section1, section2, section3, section4, section5,
section6, section7, section8, section9, section10,
section11,section12,section13,section14,section15,
section16,section17,section18,section19,section20];
%}

%Ordered Back-to-Front Outside-In Zigzag
%{
section1 = [[391:-20:11],[390:-20:10]];
section1 = fliplr(sort(section1));
section2 = [[400:-20:20],[381:-20:1]];
section2 = fliplr(sort(section2));

section3 = [[392:-20:12],[385:-20:5]];
section3 = fliplr(sort(section3));
section4 = [[395:-20:15],[382:-20:2]];
section4 = fliplr(sort(section4));

section5 = [[396:-20:16],[389:-20:9]];
section5 = fliplr(sort(section5));
section6 = [[399:-20:19],[386:-20:6]];
section6 = fliplr(sort(section6));

section7 = [[393:-20:13],[384:-20:4]];
section7 = fliplr(sort(section7));
section8 = [[394:-20:14],[383:-20:3]];
section8 = fliplr(sort(section8));

section9 = [[397:-20:17],[388:-20:8]];
section9 = fliplr(sort(section9));
section10 = [[398:-20:18],[387:-20:7]];
section10 = fliplr(sort(section10));


perm = [section1,section2,section3,section4, section5,
section6,section7,section8,section9, section10];
%}

%Ordered Back-to-Front Outside-In Switching Aisles

section1 = [391:-10:1];
section2 = [400:-10:10];
```

```
section3 = [392:-10:2];
section4 = [399:-10:9];

section5 = [395:-10:5];
section6 = [396:-10:6];

section7 = [393:-10:3];
section8 = [397:-10:7];

section9 = [394:-10:4];
section10 = [398:-10:8];

perm = [section1,section2,section3,section4,section5,
section6,section7,section8,section9,section10];



person=zeros(numofattributes, numofpassengers,'single');

for i=1:numofpassengers

person(1,i) = perm(i);
person(2,i) = ceil(person(1,i)/(numofcolumns - aisles))+1;
person(3,i) = rem(person(1,i),(numofcolumns - aisles));
person(4,i) = min(round(lognrnd(1.675,1.2)),90);

%Last row is the time it takes to put away suitcase.
%Numbers are generated bewteen 1 and 90. Numbers generated are skewed
   ↪ towards 0.

if person(2,i) >= low_emerg_row
person(2,i) = person(2,i) + 1;
end
if person(2,i) >= mid_emerg_row
person(2,i) = person(2,i) + 1;
end
if person(2,i) >= high_emerg_row
person(2,i) = person(2,i) + 1;
end

if person(3,i) >= left_aisle
person(3,i) = person(3,i) + 1;
```

```
end

if person(3,i) >= right_aisle
person(3,i) = person(3,i) + 1;
end

if person(3,i) == 0
person(3,i) = numofcolumns;
end
end

person;
clear i perm numofpassengers aisles emergencyrows



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INITIALIZE POINTS FOR PLOTTING
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hold on

p=zeros(numofrows+2,numofcolumns,3);

draw_ctr = 1;

for iii = 1:3
for i = 1:numofrows+2
for ii = 1:numofcolumns
if iii == 1
if i == 1 || i == low_emerg_row || i == mid_emerg_row ||
i == high_emerg_row || i > numofrows-2 ||
ii == left_aisle || ii == right_aisle

%nothing

else

%plot the ordered seat number of the seat

p(i,ii,iii) = text(ii-0.01, i-0.015 , ...
[num2str(draw_ctr)] , ...
'color','black' , ...
'fontsize',7 , ...
'horizontalalignment','center' , ...
'fontweight','bold');

draw_ctr = draw_ctr + 1;
end

elseif iii == 2

%plot the green point to represent the person

p(i,ii,iii) = plot(ii,i , ...
'marker','o' , ...
'markersize',8 , ...
'markeredgecolor','green' , ...
'markerfacecolor','green');
else

%plot the ordered seat number of the person

p(i,ii,iii) = text(ii-0.01, i-0.015 , ...
['X'] , ...
'color','black' , ...
'fontsize',5 , ...
'horizontalalignment','center' , ...
'fontweight','bold');
end
end
end
end

%turn off all plotted points
set(p(:,:,2:3),'visible','off');

%turn off all plotted points
set(p(numofrows+1:numofrows+2,:,1),'visible','off');

%produce each image
%saveas(gcf(),strcat('AirplaneLayout3-4-3.png'));

clear i ii iii draw_ctr
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INITIALIZE MATRIX LANDSCAPE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

totalsteps=3500;

steps=zeros(numofrows+2,numofcolumns,totalsteps,numofattributes,'
    ↪ uint16');

steps(1,numofcolumns,1,:) = person(:,1);
set(p(1,numofcolumns,2),'visible','on');
set(p(1,numofcolumns,3),'visible','on','string',num2str(steps(1,
    ↪ numofcolumns,1,1)));
set(p(1,numofcolumns,1),'visible','off');


%saveas(gcf(),strcat('3-4-3RandomBoarding',num2str(1),'.png')); %
    ↪ capture first image

person_ctr = 1;
person(:,person_ctr) = 0;



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% BEGIN BOARDING AND LOGIC %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%
% Row Movement
%%%%%%%%%%%%%%%%

for time = 2:totalsteps;

for back = numofrows:-1:2

% Initialize column variables
```

```
for twice = 1:2
if twice == 1
a1 = 1;
a3 = 3;
b1 = 6;
b3 = 5;
acol = left_aisle;
else
a1 = 7;
a3 = 8;
b1 = 12;
b3 = 10;
acol = right_aisle;
end

for col=a1:a3

if (steps(back,col,time-1,1) ~= 0) && (steps(back,col,time-1,2) ==
   ↪ back)

if (steps(back,col,time-1,3) < col)
if steps(back,col-1,time,1) == 0

if steps(back,acol,time-1,2) ~= back

steps(back,col-1,time,:) = steps(back,col,time-1,:);
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
   ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

elseif (steps(back,acol,time-1,3) > max(steps(back,a1:a3,time-1,3)))

steps(back,col-1,time,:) = steps(back,col,time-1,:);
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
   ↪ time,1)));
```

```
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

else %when steps(back,col,time-1,3) = col
steps(back,col,time,:) = steps(back,col,time-1,:); % stay fixed in
    ↪ correct seat

end
end%if

end%for col

clear col

for col=b1:-1:b3

if (steps(back,col,time-1,1) ~= 0) && (steps(back,col,time-1,2) ==
    ↪ back)

if (steps(back,col,time-1,3) > col)
if steps(back,col+1,time,1) == 0

if sum(steps(back,b3:b1,time-1,1)) > 0
min_var = min(steps(back,b3:b1,time-1,3)(steps(back,b3:b1,time-1,3)>0)
    ↪ );
else
min_var = 0;
end

if steps(back,acol,time-1,2) ~= back

steps(back,col+1,time,:) = steps(back,col,time-1,:);
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');
set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

elseif min_var ~= 0 && steps(back,acol,time-1,3) < min_var

steps(back,col+1,time,:) = steps(back,col,time-1,:);
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

else %when steps(back,col,time-1,3) = col
steps(back,col,time,:) = steps(back,col,time-1,:); % stay fixed in
    ↪ correct seat
end


end %if

end %for col

clear min_var twice



%%%%%%%%%%%%%%%%%%%
% Aisle Movement
%%%%%%%%%%%%%%%%%%%

for col = acol
```

```
slow = rand;
if (slow < 0.10 && steps(back,col,time-1,2) ~= back)
% 0.nm = nm% of the time person in aisle will not move

steps(back,col,time,:) = steps(back,col,time-1,:); %stay in position

else
if (steps(back,col,time-1,1) ~= 0)

if steps(back,col,time-1,2) > back
if steps(back+1,col,time,1) == 0
steps(back+1,col,time,:) = steps(back,col,time-1,:);
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
   ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
else
steps(back,col,time,:) = steps(back,col,time-1,:);
end

elseif steps(back,col,time-1,2) == back


%Time it takes to put away suitcase.
%If the 4th parameter for each person is not zero yet, then
%count down this number by 1 per step until zero is reached
%and then the person will sit in their seat on the next step.

if steps(back,col,time-1,4) ~= 0
steps(back,col,time,:) = steps(back,col,time-1,:);
steps(back,col,time,4) = steps(back,col,time-1,4) - 1;
else

if (steps(back,col,time-1,3) < col)

if (steps(back,col,time-1,3) > max(steps(back,col-3:col-1,time,3)))
if steps(back,col-1,time,1) == 0
steps(back,col-1,time,:) = steps(back,col,time-1,:);
```

```
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
end
else %shuffling

if steps(back,col-2,time,1) ~= 0
if steps(back,col-1,time,1) == 0

steps(back,col-1,time,:) = steps(back,col-2,time,:);
steps(back,col-2,time,:) = 0;
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col-2,2:3),'visible','off');
set(p(back,col-2,1),'visible','on');

elseif steps(back+1,col,time,1) == 0 %if both seats occcupied then
    ↪ move both

steps(back+1,col,time,:) = steps(back,col-1,time,:);
steps(back,col-1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col-1,2:3),'visible','off');
set(p(back,col-1,1),'visible','on');

steps(back,col-1,time,:) = steps(back,col-2,time,:);
steps(back,col-2,time,:) = 0;
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');
set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col-2,2:3),'visible','off');
set(p(back,col-2,1),'visible','on');

end

elseif steps(back,col-1,time,1) ~= 0

if steps(back+1,col,time,1) == 0

steps(back+1,col,time,:) = steps(back,col-1,time,:);
steps(back,col-1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col-1,2:3),'visible','off');
set(p(back,col-1,1),'visible','on');

elseif steps(back+1,col,time,2) == back
%is back+1 from the previous row?

if steps(back+2,col,time,1) == 0
steps(back+2,col,time,:) = steps(back+1,col,time,:);
steps(back+1,col,time,:) = 0;
set(p(back+2,col,2),'visible','on');
set(p(back+2,col,1),'visible','off');

set(p(back+2,col,3),'visible','on','string',num2str(steps(back+2,col,
    ↪ time,1)));
set(p(back+1,col,2:3),'visible','off');
set(p(back+1,col,1),'visible','on');

steps(back+1,col,time,:) = steps(back,col-1,time,:);
steps(back,col-1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
```

```
set(p(back,col-1,2:3),'visible','off');
set(p(back,col-1,1),'visible','on');
end
end
end

steps(back,col,time,:) = steps(back,col,time-1,:);

end %shuffling


else %column is greater than aisle and do the following


if sum(steps(back,col+1:col+2,time,1)) > 0
min_var = min(steps(back,col+1:col+2,time,3)(steps(back,col+1:col+2,
    ↪ time,3)>0));
else
min_var = 0;
end


if min_var ~= 0 && steps(back,col,time-1,3) < min_var

if steps(back,col+1,time,1) == 0
steps(back,col+1,time,:) = steps(back,col,time-1,:);
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

end

elseif min_var == 0

if steps(back,col+1,time,1) == 0

steps(back,col+1,time,:) = steps(back,col,time-1,:);
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

end

elseif steps(back,col+2,time,1) ~= 0 %shuffling

if steps(back,col+1,time,1) == 0

steps(back,col+1,time,:) = steps(back,col+2,time,:);
steps(back,col+2,time,:) = 0;
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');

set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
    ↪ time,1)));
set(p(back,col+2,2:3),'visible','off');
set(p(back,col+2,1),'visible','on');

elseif steps(back+1,col,time,1) == 0

%if both seats occcupied then move both

steps(back+1,col,time,:) = steps(back,col+1,time,:);
steps(back,col+1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col+1,2:3),'visible','off');
set(p(back,col+1,1),'visible','on');

steps(back,col+1,time,:) = steps(back,col+2,time,:);
steps(back,col+2,time,:) = 0;
set(p(back,col+1,2),'visible','on');
set(p(back,col+1,1),'visible','off');
set(p(back,col+1,3),'visible','on','string',num2str(steps(back,col+1,
   ↪ time,1)));
set(p(back,col+2,2:3),'visible','off');
set(p(back,col+2,1),'visible','on');

end

steps(back,col,time,:) = steps(back,col,time-1,:);

else %if steps(back,col+1,time,1) ~= 0

if steps(back+1,col,time,1) == 0

steps(back+1,col,time,:) = steps(back,col+1,time,:);
steps(back,col+1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
   ↪ time,1)));
set(p(back,col+1,2:3),'visible','off');
set(p(back,col+1,1),'visible','on');

elseif steps(back+1,col,time,2) == back
%is back+1 from the previous row?

if steps(back+2,col,time,1) == 0

steps(back+2,col,time,:) = steps(back+1,col,time,:);
steps(back+1,col,time,:) = 0;
set(p(back+2,col,2),'visible','on');
set(p(back+2,col,1),'visible','off');

set(p(back+2,col,3),'visible','on','string',num2str(steps(back+2,col,
   ↪ time,1)));
set(p(back+1,col,2:3),'visible','off');
set(p(back+1,col,1),'visible','on');

steps(back+1,col,time,:) = steps(back,col+1,time,:);
steps(back,col+1,time,:) = 0;
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');
set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col+1,2:3),'visible','off');
set(p(back,col+1,1),'visible','on');
end
end

steps(back,col,time,:) = steps(back,col,time-1,:);

end %shuffling

end
end %storing luggage overhead bin

else % < back

steps(back,col,time,:) = steps(back,col,time-1,:);

end %elseif relay

if steps(back+1,col,time,2) == back
if steps(back,col,time,1) == 0

steps(back,col,time,:) = steps(back+1,col,time,:);
steps(back+1,col,time,:) = 0;
set(p(back,col,2),'visible','on');
set(p(back,col,1),'visible','off');

set(p(back,col,3),'visible','on','string',num2str(steps(back,col,time
    ↪ ,1)));
set(p(back+1,col,2:3),'visible','off');
set(p(back+1,col,1),'visible','on');

end
end

if steps(back+2,col,time,2) == back
if steps(back+1,col,time,1) == 0

steps(back+1,col,time,:) = steps(back+2,col,time,:);
steps(back+2,col,time,:) = 0;
```

```
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
   ↪ time,1)));
set(p(back+2,col,2:3),'visible','off');
set(p(back+2,col,1),'visible','on');

end
end

end %if steps ~= 0

end %slow
end %for col = acol

end %twice

end %for back



%%%%%%%%%%%%%%%%%%
% Entering Plane
%%%%%%%%%%%%%%%%%%

for back = 1

for col = left_aisle
if steps(back,col,time-1,1) ~= 0
if steps(back+1,col,time,1) == 0
steps(back+1,col,time,:) = steps(back,col,time-1,:);
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
   ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

else
steps(back,col,time,:) = steps(back,col,time-1,:);
```

```
end
end
end

for col = left_aisle+1:right_aisle-1

if steps(back,col,time-1,1) ~= 0
if steps(back,col-1,time,1) == 0
steps(back,col-1,time,:) = steps(back,col,time-1,:);
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

else
steps(back,col,time,:) = steps(back,col,time-1,:);
end
end%if

end %col

for col = right_aisle
temp = 2; %define temp
if steps(back,col,time-1,1) ~= 0
if steps(back,col,time-1,3) == 5
%temp = round(rand(1)); %random integer 0 or 1
end

if (steps(back,col,time-1,3) > 6 || temp == 1)
if steps(back+1,col,time,1) == 0
steps(back+1,col,time,:) = steps(back,col,time-1,:);
set(p(back+1,col,2),'visible','on');
set(p(back+1,col,1),'visible','off');

set(p(back+1,col,3),'visible','on','string',num2str(steps(back+1,col,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');
```

```
else
steps(back,col,time,:) = steps(back,col,time-1,:);
end
else
if steps(back,col-1,time,1) == 0
steps(back,col-1,time,:) = steps(back,col,time-1,:);
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

else
steps(back,col,time,:) = steps(back,col,time-1,:);
end
end
end
end

for col = right_aisle+1:numofcolumns

if steps(back,col,time-1,1) ~= 0
if steps(back,col-1,time,1) == 0
steps(back,col-1,time,:) = steps(back,col,time-1,:);
set(p(back,col-1,2),'visible','on');
set(p(back,col-1,1),'visible','off');

set(p(back,col-1,3),'visible','on','string',num2str(steps(back,col-1,
    ↪ time,1)));
set(p(back,col,2:3),'visible','off');
set(p(back,col,1),'visible','on');

else
steps(back,col,time,:) = steps(back,col,time-1,:);
end
end%if

end %col

end %back = 1
```

```
%%%%%%%%%%%%%%
% New Person
%%%%%%%%%%%%%%
if mod(time,6) == 1 %passenger boarding frequency

if (person_ctr < length(person)) && (steps(1,numofcolumns,time,1) ==
    ↪ 0)
person_ctr = person_ctr + 1;
steps(1,numofcolumns,time,:) = person(:,person_ctr);
person(:,person_ctr) = 0;
set(p(1,numofcolumns,2),'visible','on');
set(p(1,numofcolumns,3),'visible','on','string',num2str(steps(1,
    ↪ numofcolumns,time,1)));
set(p(1,numofcolumns,1),'visible','off');
end
end


%%%%%%%%%%%%%%%%%
% Produce Image
%%%%%%%%%%%%%%%%%

%saveas(gcf(),strcat('3-4-3RandomBoarding',num2str(time),'.png'));



%%%%%%%%%%%%%%
% Stop Logic
%%%%%%%%%%%%%%

if (sum(steps(:,left_aisle,time,1)) == 0 && sum(steps(:,right_aisle,
    ↪ time,1)) == 0
&& sum(steps(low_emerg_row,:,time,1)) == 0 && sum(steps(mid_emerg_row
    ↪ ,:,time,1)) == 0
&& sum(steps(high_emerg_row,:,time,1)) == 0 && sum(steps(1,:,time,1))
    ↪ == 0
&& sum(steps(numofrows,:,time,1)) == 0) && (sum(person(1,:)) == 0)
break;
end
```

```
%%%%%%%%%%%%%%%
% Break Test
%%%%%%%%%%%%%%%


%{
if time == 15
break;
end
%}


end %for time



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% END BOARDING AND LOGIC %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%produce final image
%saveas(gcf(),strcat('figure',num2str(1),'.png'));

printf("Simulation required %i steps.\n",time);

mat(major) = time;

end %major

mat
```