INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Modelling and optimizing the boarding time of a Boeing 777

Hugo Ferrer[1], João Hemptinne[1], Hippolyte Labarrière[1]

Tutored by Marc Orfila[2]

[1]Student at INSA Toulouse
[2]Operations Performance Analyst at Air France

1 Abstract

Air France carries tens of thousands of passengers every day. Every company tries to satisfy its customers so flights need to be punctual. Flight companies want to reduce the time spent on the ground and especially the boarding phase. Our aim is to improve boarding efficiency with mathematical tools. We confirmed more efficient methods of boarding. To do so, we modelled and optimized boarding time using simulated annealing. Our results demonstrate that flight companies could save a lot of time by using an optimized method of boarding. We also designed an algorithm to predict the boarding time of a flight with a given sequence of passengers. Our model being really simple, we could imagine that some sophisticated research could give more relevant results. We anticipate that the current boarding method will be soon replaced by more efficient boarding methods such as the Steffen method.

2 Introduction

2.1 Motivation

Since 1903 and the creation of the first plane ever (the Wright Flyer), until today, planes have evolved remarkably. They have become heavier but faster, safer, and more affordable for everyone. Their use has also changed greatly, being at the beginning an experiment, then becoming a way of transport for people and merchandise, and also sadly a deadly weapon, even today.

As a way of transport, planes have become a business item for airlines. They plan flights every day all over the world and, of course, the more flights that are planned everyday, the more money they earn. To reach this objective, companies try to reduce the time of all the processes that are essentials, from the cleaning to the preparation of the flights (refilling oil, re-stocking food, drinks, blankets), to finally proceed with the boarding.

Thus, boarding is a step that has a huge influence in every flight. Each airline has to reduce the boarding time of its plane to plan more flights. This is the aspect we developed in this paper: in which ways can a boarding be improved ?

2.2 Outline of the paper

In the first part, we identified some existing methods.

In the second part, we modelled numerically the boarding of a flight and we designed a function which calculates the boarding time of a given sequence of passengers. We optimized this function by using simulated annealing.

Finally, we applied our research to real data: we computed the boarding time of a Boeing 777 using some hypotheses. This allowed us to improve our algorithm and to obtain solid results.

3 Existing boarding methods

3.1 Random boarding

This is the most frequent boarding method adopted by airlines, alongside the method by blocks. The principle is simple: passengers enter the plane in a random order, with their seat number already set and written on their ticket (Figure 1). This means that the interaction between all the passengers is unpredictable and can vary a lot, from minimal to very strong, depending on the number of passengers and their behaviour. Thus, even if this method is used in all the airports around the world, airlines can not feel confident about this method because they can not be sure that their flights will all take off on time. But it is the most simple to organise, because airline employees do not have to arrange passengers in the order of their seats before entering the plane.

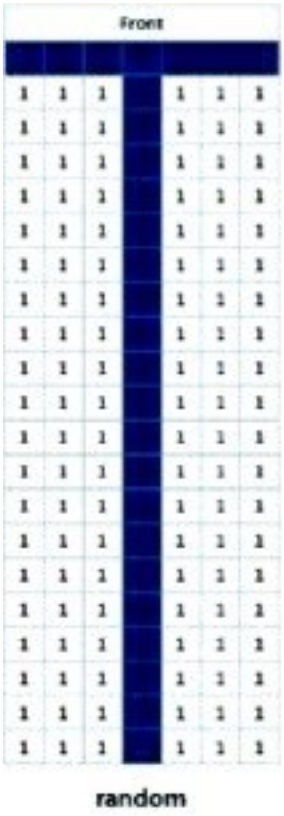

Figure 1: Random boarding scheme[3]

3.2 By blocks

This method is mainly linked to a delimitation of the plane by blocks. The passengers board block by block, with any method (for example random, outside-in or back to front). The delimitation is chosen by the company. It is used, for example, for boarding by ticket type. The premium class boards before the sky-priority class, which also boards before the economic class. It helps reducing interactions between passengers as few of them board at each step. This method gives a similar result as the random one, but with a slightly better control on the interaction.

[3] Adapted from [6]

3.3 Back-to-front boarding

This is a random boarding organised in big blocks of seats (a set number of seat rows). The first block of people sits in the back of the plane, and the last one in the front. This method is interesting since it reduces interactions between passengers that want to sit. None of the passengers in a block interact with any passengers of another block. While a block of passengers is not fully boarded, the following one have to wait to enter the plane.

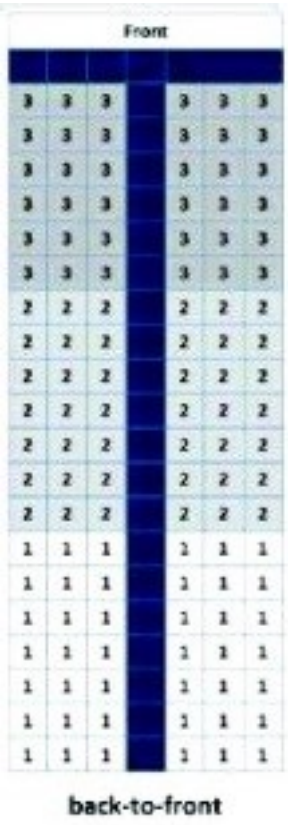

Figure 2: Back-to-front boarding scheme[4]

Figure 2 illustrates the back-to-front boarding. The lower the number is, the earlier the passenger boards. This rule is the same for all the figures following in this section.

[4]Adapted from [6]

3.4 By half-block

This method is similar to the previous one, but with smaller blocks. The width of each block is half a row instead of an entire row. The objective is again to reduce the number of people getting on board at a time (Figure 3). Smaller blocks imply less interaction between passengers, but while a block boards in, all the other blocks have to wait. Thus, having a lot of blocks increases the waiting time outside.

Figure 3: Boarding by half-blocks scheme[5]

[5] Adapted from [6]

3.5 By row, by half-row

With this method, the passengers board row by row, beginning with the back of the plane, and going back to front. Boarding by row is a kind of boarding by blocks, considering each row as a small block. The boarding by half-row follows the same principle but filling half of the plane by row, and then doing the same with the other half (Figure 4). The issues of these methods are the same as those of the previous one: there are a lot of small boarding blocks.

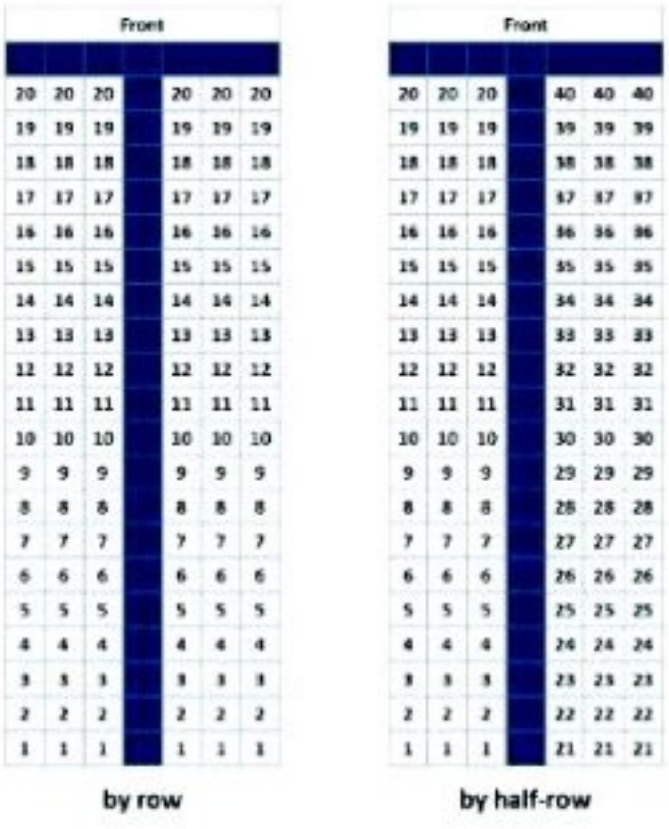

Figure 4: Boarding by row and by half-row schemes[6]

[6] Adapted from [6]

3.6 Outside-in

This method is an organised boarding where people seated at the window-side enter first. Then, people seated in the middle of the row get on board. People seated near the corridor enter last. It is a symmetric boarding, following the columns (A, B, C for example) of the plane's seats (Figure 5). This is again a boarding by blocks, considering a column as a block. Interactions between passengers are here reduced as in each block, only two passengers sit down at the same row.

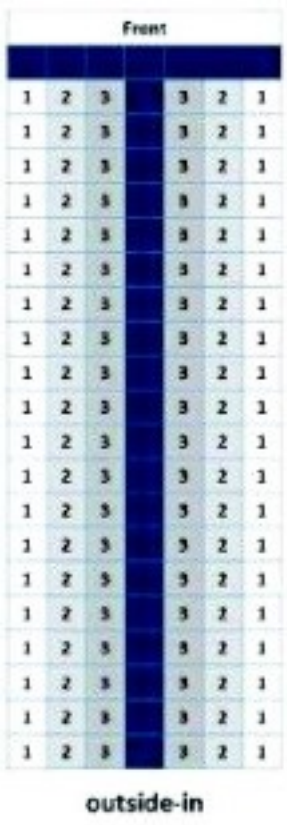

Figure 5: Outside-in boarding scheme[7]

[7] Adapted from [6]

3.7 Steffen Method

This method is a recent method created by JH Steffen [4]. He defined it using a principle seen in some other methods. He believed that people who sit near each other will take more time than distant ones, because of natural interactions (e.g. waiting while the other passenger is putting its luggage over his seat, waiting for him to sit). This method is similar to the outside-in method but "skipping" one seat over two as illustrated in Figure 6. The problem of the Steffen method is that boarding has to be very organised outside the plane as the boarding is non-random. This preliminary organisation takes time and negatively compensates the positive effects of the method.

Figure 6: Steffen boarding scheme[8]

[8] Adapted from [6]

4 Modelling and calculating the boarding time of a flight

4.1 Principle

Our main objective was to calculate the boarding time of a flight knowing the seating order of the passengers. We created an algorithm which takes a vector of the plane seats in the seating order of the passengers and returns an approximation of the boarding time. This algorithm takes in account many different factors, which were implemented using some parameters (expressed in seconds):

- $TWalk$: the time a passenger takes to move along a row of seats,
- $TFollow$: the time between the entry of two consecutive passengers,
- $TLug$: the time each passenger takes to stow its luggage in a cargo hold,
- $TWait$: the time a passenger has to wait for other passengers in the same row to stand up and let it sit at its seat.

Firstly, we set all of these parameters arbitrarily: $TWalk = 2$ seconds, $TFollow = 2$ seconds, $TLug = 12$ seconds and $TWait = 6$ seconds.

The main principle of this function is to calculate the time between two passengers who sit down by taking in account all the factors addressed above, and then to add this value to the total boarding time. Hence, this algorithm has the following structure (Algorithm 1):

Data:
$order$: A vector of seats representing each passenger in the order of the boarding
Result:
T the boarding time is set to 0;
N the number of passengers is set according to $order$;
i the iteration parameter is set to 1;
P the matrix of passengers is set to 0;
$T+=$ the time the first passenger ($order_1$) takes to go to his seat;
while $i < N$ **do**
  t the time difference between the $(i+1)$-th passenger ($order_{i+1}$) and the i-th passenger ($order_i$) sit at their seats is set to 0;
  if *the* $(i+1)$*-th passenger has to wait for the* i*-th to sit at his seat* **then**
    $t =$ the time the $(i+1)$-th passenger takes to go to his seat from the place he had to wait for the i-th passenger;
  else
    $t =$ the difference between the time taken by the the $(i+1)$-th passenger to go to his seat and the time taken by the the i-th passenger;
    $t+= TFollow$;
  end
  $T+= t$;
  The i-th passenger is added to the matrix P (we set the value of the corresponding coefficient to 1);
  $i+= 1$;
end
return T

Algorithm 1: Computation of the boarding time

4.2 Analysis of existing boarding methods

We calculated an approximation of the boarding time for each known method in different situations. Figure 7 and Figure 8 display different boarding methods described in Part 3 implemented in Python. Each square represents a seat and the color corresponds to the order of boarding: the lightest square is the first passenger to board and the darkest is the last one.

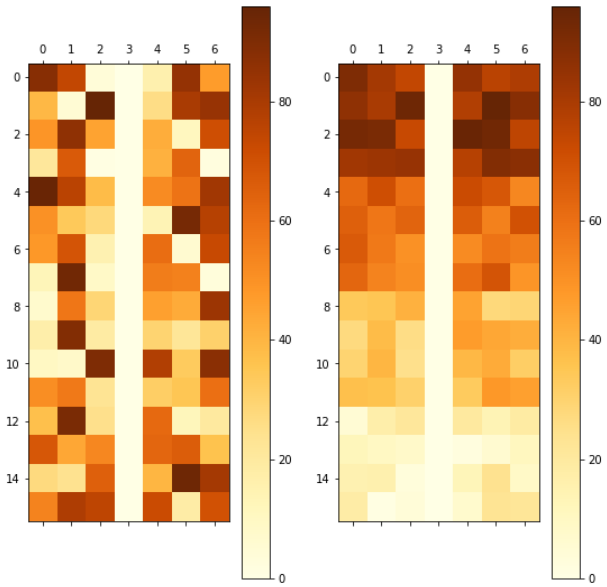

Figure 7: Random and back to front methods implemented in Python

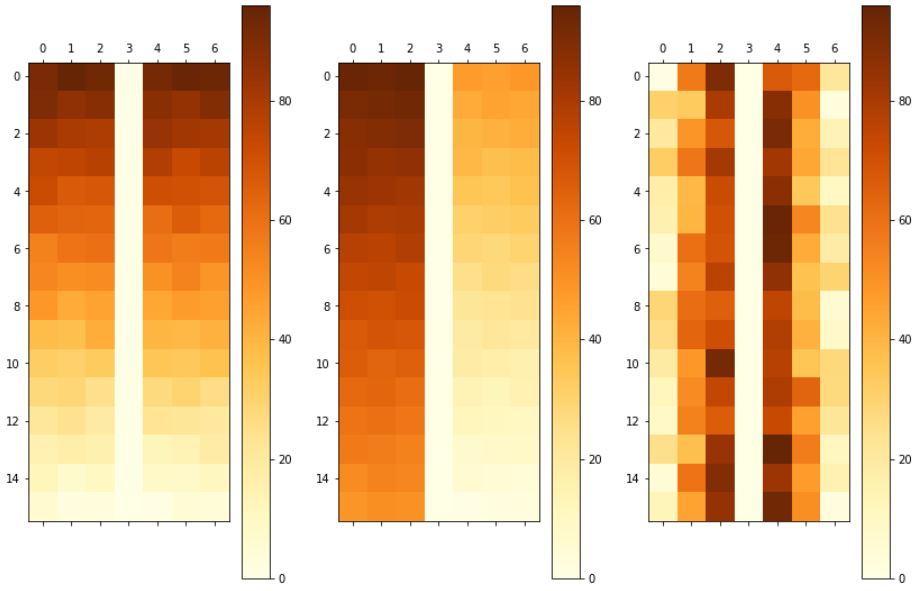

Figure 8: By row, by half row and outside-in methods implemented in Python

Even if our results were not representative of the real boarding time (because of our parameters which may be inaccurate), we verified by comparing the methods that our algorithm was consistent. We arbitrarily fix the constants of the function and it returned the following results (Figure 9):

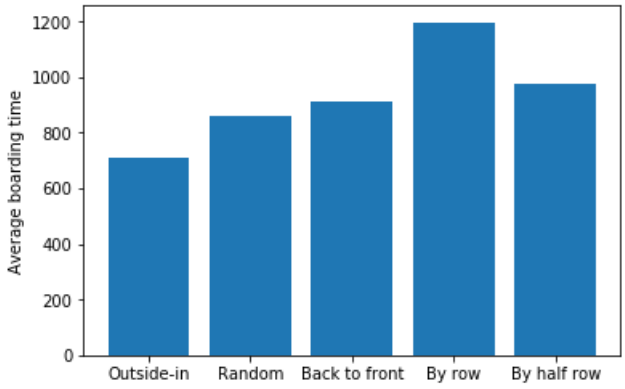

Figure 9: Calculated average boarding time of different boarding methods

This outcome is very similar to the results of other studies [2]: the outside-in method is the fastest random method of boarding in front of the random method and the back-to-front method. On the other hand, boarding by row and by half row are totally inefficient.

We varied some of the function parameters to see how the boarding time would change. First of all, we observed the variation of the average boarding time of each method according to the parameter $TWait$ (Part 4.1).

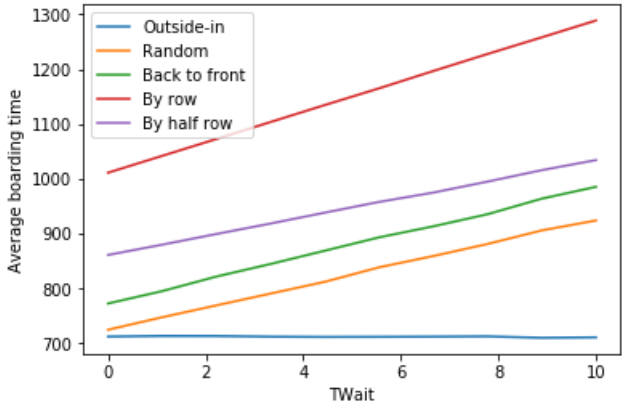

Figure 10: Evolution of the average boarding time according to $TWait$

Figure 10 shows that the only boarding method which does not depend on $TWait$ is the outside-in method. This result is logical as passengers did not have to wait in the corridor to sit at its seat: this is one of the principles of this method (Part 3.6). The other methods vary linearly according to $TWait$: the longer a passenger has to wait for others to stand up, the more the boarding time will increase.

We also varied the parameter $TLug$ to understand the behaviour of each method (Figure 11). This graph shows that the parameter $TLug$ has a big influence on the efficiency of each method. If $TLug = 0$ (the passengers do not have any luggage), the least time-efficient method is the random method. Moreover, the by row and by half row methods are slightly efficient. However, when $Tlug$ increases, the rank of efficiency of each method changes, except for the outside-in boarding. The random method then becomes the second fastest method while the average boarding time of the by row and by half row increases significantly. We also observe that the Outside-in method is still the fastest method whatever the value of $TLug$.

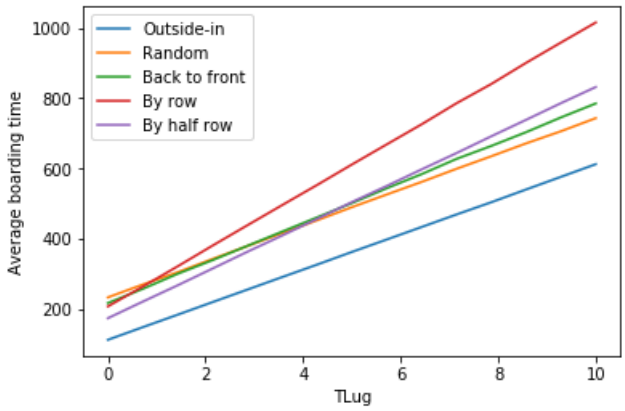

Figure 11: Evolution of the average boarding time according to $TLug$

The parameters $TWait$ and $TLug$ influence the most each method behaviour. Depending on these values, a method can be drastically improved. Comparatively, if we change the value of $TWalk$, the boarding time of every method varies similarly (Figure 12).

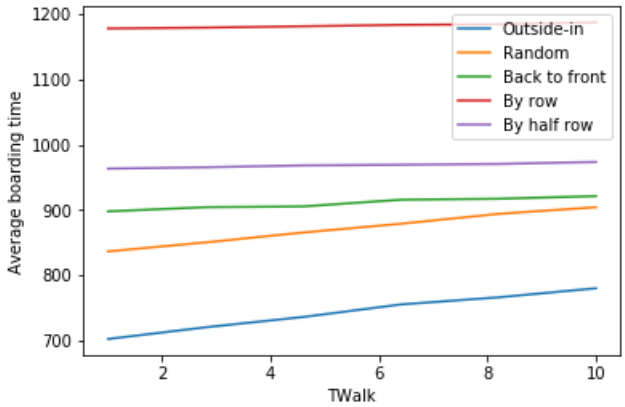

Figure 12: Evolution of the average boarding time according to $TWalk$

4.3 Applying simulated annealing to the boarding time

We applied an optimization method (simulated annealing) to our algorithm to find the boarding method which minimizes the boarding time calculated with our function. This method is often used to find an approximation of the global minimum of a function with many variables.

Simulated annealing is a succession of iterations. It requires an initial state and at each step, the algorithm considers a possible neighbour step. The next iterate is randomly set to the current iterate or to the neighbour. The probability of each possibility is connected to a function: the acceptance probability function P that depends on the energies of the current step and of the neighbour step considered, and on the temperature T (itself depending on time). The algorithm considers the difference of energy $\Delta E = e_{new} - e_{current}$: if this value is negative, i.e. the energy of the neighbour state is smaller, the next iterate is set to the neighbour state. If not, the next iterate is set to the neighbour state with a probability of $e^{-\frac{\Delta E}{T}}$. Otherwise, this value is set to the current state. The temperature can be kept constant during all the process or it can decrease at each step. It depends on the phenomenon being modelled. Thus, if $\Delta E < 0$, it means that the algorithm goes "downhill", i.e. that it is getting close to a local minimum. The case $\Delta E > 0$ is very important: the simulated annealing goes "uphill" and leaves a local minimum to reach the global minimum that would not be obtained by only going "downhill".

The travelling salesman problem is a good illustration of this algorithm. A salesman wants to find the shortest way to visit all of the cities of a defined initially set, knowing the distance between all of them. Here, the energy at each step is the total time of the salesman's trip. The only difference between two neighbour states is that two cities are switched in the sequence of the set. The algorithm keeps running even if it has found a local minimum: it is its principal interest. Figure 13 illustrates this problem, having on the left the initial travel, and on the right the optimized one.

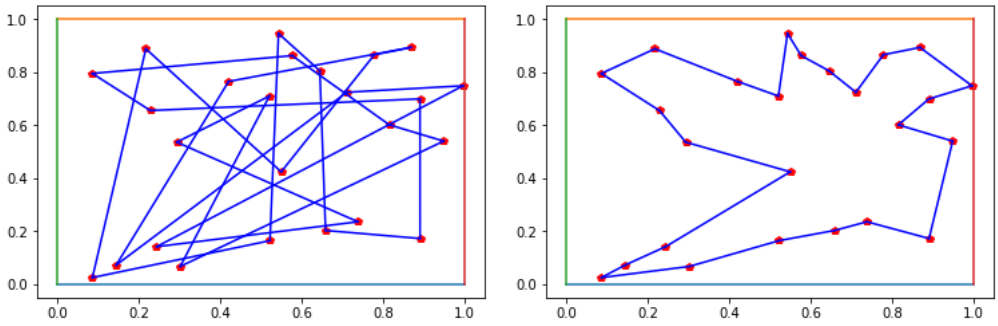

Figure 13: Travelling salesman problem with twenty-five cities, initial state on the left, final state on the right

Simulated annealing is a slow algorithm: it always converges on the global minimum but it takes a lot of time to run. Thus, we applied this method to a simple situation: we considered a plane with only twenty-four seats (four rows of six seats). This choice give convincing results that could be expanded to more complex cases (Figure 14). The energy here is the boarding time of the flight calculated with our algorithm.

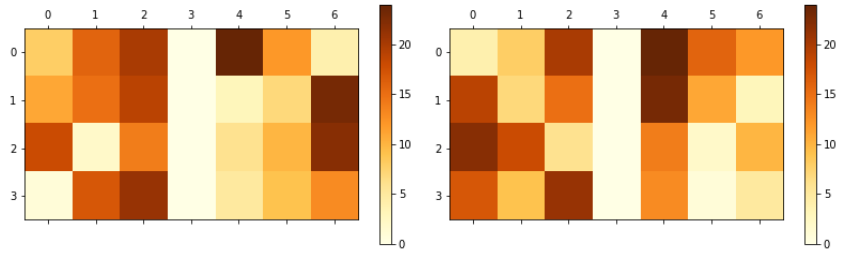

Figure 14: Two different optimal methods of boarding

Logically, this optimization problem has not an unique solution, there are different boarding methods which minimize the boarding time. We can verify this with the above results: Figure 14 represents two minimizers of our problem but boarding seems much different. However, we were able to use these solutions to identify a general scheme for the optimal solutions. The optimal methods of boarding have the same pattern: in each solution, there are six waves of passengers (because there are six seats by row in our case). In each wave, the passengers board by row: the first passenger has a seat on the first row, the second on the second row and so on. This boarding method reduces interference between passengers as the only interaction takes place between two waves of passengers. These results are interesting as we found the most optimized method of boarding but they are complex to use efficiently. There is no simple way to formulate « the optimal boarding method » as there are many possibilities. However, we found a simple method which optimizes boarding time.

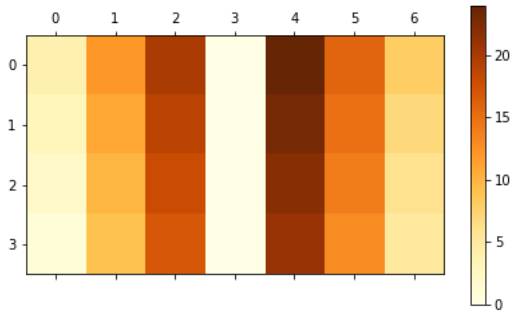

Figure 15: Optimal method of boarding, mixing outside-in and by row constraints

This method is a mix between the boarding by row and the outside-in method (Figure 15) but is non-random. Hence, it offers an impressive performance (Figure 16) but the sequence of passengers has to be set before the flight.

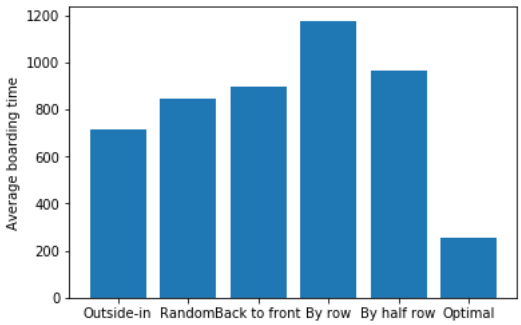

Figure 16: Average boarding time of boarding methods including the optimal method

The results show that our algorithm was consistent so we used it specifically on our initial problem: the Boeing 777.

5 Computing the boarding time of a Boeing 777

5.1 Hypothesis and simplification of the problem

The main objective of this last part was to use our algorithm in the context of a Boeing 777. This function was created by considering that all the passengers come from the same entrance, at the back of the seats. Moreover, we designed our function to calculate the boarding time for a plane with only one corridor between the seats. As illustrated on Figure 17, the Boeing 777 is more complex than the cases studied before.

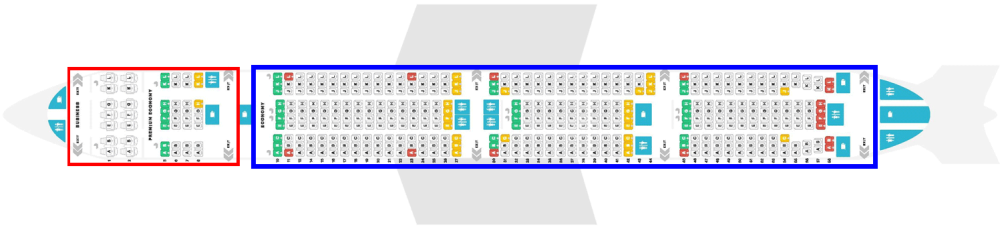

Figure 17: Boeing 777 layout[9]

The Boeing 777 is divided in two parts: the front of the plane contains premium economy and business classes (in red on the Figure 17) and the remainder of the plane contains economy class (in blue on the Figure 17). The most important factor in the boarding time of the plane is the boarding of the economy class so we did not consider the front of the plane, i.e. the business class. We divided the economy class into six blocks. Then, instead of two corridors in each part, there is only one in each block. Figure 18 represents this separation.

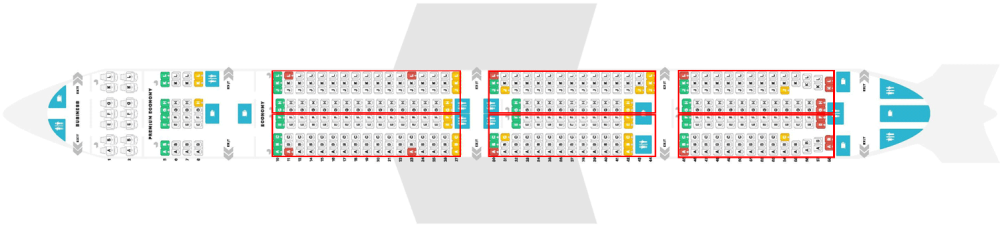

Figure 18: Boeing 777 economy class divided in six blocks[9]

However, this hypothesis simplifies the problem because this means that a passenger who has a seat on the left side of the plane will always go to its seat using the left corridor.

Lastly, we supposed that all the passengers begin the boarding at the back of the seats. It enabled us to use our function to compute the boarding time.

[9] Adapted from [1]

5.2 Data analysis

So far, we set the sequences of passengers which we worked on. Then we used real data to improve our function. We adapted our work to a different case.

To do so, we used data on real boardings from Charles de Gaulle airport in Paris which included the hour of boarding and the seat number of each passenger for fifty-four flights (Figure 19).

	A	B	C	D
1	Time pax boarded	Timestamp pax boarded (dim)	Security number	Seat number
2	04/03/2019 21:26:45	9296486	242	07G
3	04/03/2019 21:29:38	9296489	199	45F
4	04/03/2019 21:29:52	9296489	196	45G
5	04/03/2019 21:30:01	9296490	198	45H
6	04/03/2019 21:31:02	9296491	182	32G
7	04/03/2019 21:31:19	9296491	264	18J
8	04/03/2019 21:31:22	9296491	178	18K
9	04/03/2019 21:31:24	9296491	263	18H
10	04/03/2019 21:31:43	9296491	279	23B
11	04/03/2019 21:31:53	9296491	280	21B

Figure 19: Sample of data used in this part

We transformed external data which were in an Excel format into a python vector representing the sequence of passengers. Following our hypothesis, the data was already adapted to our function as we considered six independent blocks with an entry at the back. The aim was to take a given character string representing a seat/passenger in the Excel form (for example "24D"), to find the block of the plane which contains this seat and then to add the value corresponding to this seat into the vector of the block. At the end, we obtained six vectors representing the sequence of passengers in each part of the plane. Thus, we applied our function to these vectors and to calculate an approximation of the boarding time in each part of the plane.

However, a phenomenon falsified the results. In real life, passengers do not board at the same time: some of them board early or late. Our function calculates the boarding time when the passengers get on the plane one by one in a short period so this case causes an issue. Our choice was to isolate the moment of the boarding when many passengers board in a short period and then to use this data to improve our function. We called this the "rush" part of the boarding.

5.3 Calculating the boarding time of each part of the plane

We used our function on the given data, i.e. the fifty-four flights. We calculated an approximation of the boarding time in each block of each plane (Figure 20) and we compared our results with the real values.

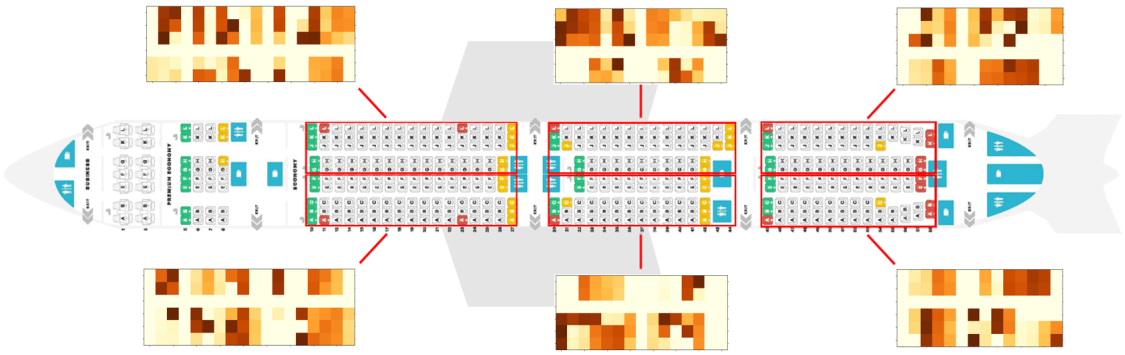

Figure 20: Representation of our code working on the problem of the Boeing 777[10]

This led us to improve the time constants of the function e.g. $TLug$, $TWait$, $TWalk$ (Part 4.1). So far, we set these parameters and as a consequence the first results obtained were not acceptable: the average gap between the calculated boarding time and the real one was more than ten minutes. The goal was to minimize this gap by finding the most accurate constants. We first tried to set these constants one by one manually before using an optimization algorithm.

We applied simulated annealing to this new problem (Part 4.3): the energy of the problem was here the average gap between the calculated boarding time and the real value. The optimal values of the time constants found with this algorithm allowed the function to reach an average gap of less than four minutes. Moreover, these values were consistent with what they were meant to represent:

- $TWalk = 1$ second
- $TFollow = 23.5$ seconds
- $TLug = 23$ seconds
- $TWait = 7$ seconds

[10] Adapted from [1]

5.4 Computing the total boarding time of the plane

Previously, we obtained six values for one boarding with our function: one value of the calculated boarding time for each block of the plane. The final step of the project was to approximate the total boarding time of the plane.

Let us note t_1, t_2, t_3, t_4, t_5 and t_6 the values of the boarding time for each block and T the total boarding time of the plane. We seek f such that:

$$T = f(t_1, t_2, t_3, t_4, t_5, t_6).$$

Firstly, we set f as the maximum function:

$$f(t_1, t_2, t_3, t_4, t_5, t_6) = \max(t_1, t_2, t_3, t_4, t_5, t_6).$$

In that case, we consider that the boarding time of the plane equals the maximum of the boarding time of each part of the plane. The following inequality is always verified:

$$T \geq \max(t_1, t_2, t_3, t_4, t_5, t_6).$$

Secondly, we tried to approximate the total boarding time with f of the following form:

$$f(t_1, t_2, t_3, t_4, t_5, t_6) = \alpha_1 t_1 + \alpha_2 t_2 + \alpha_3 t_3 + \alpha_4 t_4 + \alpha_5 t_5 + \alpha_6 t_6,$$

with $(\alpha_i)_{i=1..6} \in \mathbf{R}$.

As we had boarding data of $n = 54$ flights, we could have an expression of $(\alpha_i)_{i=1..6}$ by solving:

$$\underbrace{\begin{pmatrix} t_{11} & t_{12} & t_{13} & t_{14} & t_{15} & t_{16} \\ t_{21} & t_{22} & t_{23} & t_{24} & t_{25} & t_{26} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ t_{n1} & t_{n2} & t_{n3} & t_{n4} & t_{n5} & t_{n6} \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \end{pmatrix}}_{\alpha} = \underbrace{\begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_n \end{pmatrix}}_{b},$$

with t_{ij} the boarding time of the j-th block of the i-th flight and T_j the total boarding time of the j-th flight.

We used the least-squares method to find the vector α which optimizes the value $\|A\alpha - b\|^2$. We have then:

$$A^\top A\alpha = A^\top b$$
$$\implies \alpha = (A^\top A)^{-1} A^\top b.$$

Even if α depends on the given data, it gives a good approximation of the boarding time behaviour as n is big enough.

We applied these two expressions of f to our program. Our results showed that the second function was more effective to predict the total boarding time of a flight: we applied our algorithm to the Charles de Gaulle boarding data with every function and finally the average error on the boarding time was significantly better with the second function. The average error on the total boarding time (with our set of data) was four minutes with the first function and three minutes with the second one. Hence, we set $f(t_1, t_2, t_3, t_4, t_5, t_6) = \alpha_1 t_1 + \alpha_2 t_2 + \alpha_3 t_3 + \alpha_4 t_4 + \alpha_5 t_5 + \alpha_6 t_6$.

The final structure of our program was then :

Data:
$file$: the name of the Excel file which contains the data used
Result:
T the boarding time of the plane is set to 0;
$(t_i)_{i=1..6}$ are set to 0;
$i = 1$;
The "rush part" of the boarding is extracted from $file$;
$\left(order^i\right)_{i=1..6}$ are extracted from the "rush part" of the boarding;
while $i < 6$ **do**
    t_i is calculated with the Algorithm 1 applied on $order^i$;
    i+=1;
end
$T = f(t_1, t_2, t_3, t_4, t_5, t_6)$;
return T;

Algorithm 2: Computation of the total boarding time of a Boeing 777

6 Conclusion

We studied and analyzed existing boarding methods by using our algorithm and comparing them with each other. This led us to find the most optimized and to understand their weaknesses. We also optimized the boarding time with simulated annealing and we found an optimal boarding method. However, as the Steffen method, this boarding method is deterministic and therefore complex for airlines to use.

It finally appears that the most efficient way to reduce the boarding time is to set the order of boarding before each flight. This means that airlines would have to change their organisation. However, this change could decrease the satisfaction of customers as it would also reduce their freedom.

We designed a program which computes the time of a given boarding. Our results were accurate even if our algorithm did not take random incident in account. The average gap between the real boarding time and the one we calculated was only three minutes. This value is small in comparison with the boarding time (twenty-thirty minutes), so this was a satisfactory outcome. However, we tested our algorithm on the same data we used to optimize our parameters and the function f which shows that our results were very accurate. We can not be sure that our algorithm would be that accurate on a new data set.

Our work could nevertheless help current research as we obtained consistent results. We used different mathematical tools to understand and solve this concrete problem. We could improve our research by considering the random incidents and the influence of each passenger on the boarding time: it could lead to a statistical approach of this project. It would need more data but would give more accurate outcomes.

However, we need to consider the ethical dimension of our work. The society is currently in a context of environmental crisis, and planes contribute to atmospheric pollution. Optimizing the boarding time should be accompanied by reducing of the environmental impact of each plane. This should be one of the biggest goals of airlines for the next decades.

References

[1] Air france seat maps: Boeing 777-300er (77w) caribbean.

[2] Richard Cimler, Eva Kautzká, Kamila Olševičová, and Martin Gavalec. Agent-based model for comparison of aircraft boarding methods. In *Proceedings of 30th International Conference Mathematical Methods in Economics Agent-Based*, pages 73–78, 2012.

[3] Florian Jaehn and Simone Neumann. Airplane boarding. *European Journal of Operational Research*, 244(2):339–359, 2015.

[4] Jason H Steffen. Optimal boarding method for airline passengers. *Journal of Air Transport Management*, 14(3):146–150, 2008.

[5] Jason H Steffen and Jon Hotchkiss. Experimental test of airplane boarding methods. *Journal of Air Transport Management*, 18(1):64–67, 2012.

[6] Hendrik Van Landeghem and Annelies Beuselinck. Reducing passenger boarding time in airplanes: A simulation based approach. *European Journal of Operational Research*, 142(2):294–308, 2002.

[6] [3] [4] [5] [2] [1]