Flight Efficiency in European Airspace

Problem presented by

Martin Hawley and Karol Gotz

Winsland Consulting

ESGI107 was jointly hosted by
The University of Manchester
Smith Institute for Industrial Mathematics and System Engineering

with additional financial support from
KTN Ltd
Natural Environment Research Council
Manchester Institute for Mathematical Sciences

Report author

Cristina Sargent (Smith Institute)

Executive Summary

The problem posed by Winsland Consulting was concerned with finding an efficient correlation between a 4D flight path (the civilian requirement) and a 4D volume of restricted airspace (due to weather or military use). It further asked how this could be used to explore the best shape of airspace reservations in respect of: sensitivity to civil demand (daily and seasonal); best configuration (shape and time); and the best times to release military sectors to civil traffic (network and local level). The ESGI mathematicians developed a number of approaches that gave confidence that an airspace optimisation tool is feasible.

Version 1.0
September 11, 2015
iv+30 pages

Contributors

Chris Cawthorn (University of Cambridge)
Gemma Cupples (University of Birmingham)
Eoin Devane (University of Cambridge)
Mel Devine (University of Limerick)
Craig Holloway (University of Birmingham)
Janis Klaise University of Warwick)
James Mathews (University of Cambridge)
Faizal Nazar (University of Warwick)
Cezary Olszowiec (Imperial College, London)
Clarice Poon (University of Cambridge)
Gunnar Peng (University of Cambridge)
Jan Van Lent (University of the West of England, Bristol)
Cristina Sargent (Smith Institute)
Emily Walsh (University of the West of England, Bristol)

Contents

1 Problem statement

(1.0.1) There are around 30,000 flights per day in European airspace. This flow of traffic is controlled through 70 air traffic control centres and coordinated by the Network Manager, part of Eurocontrol. Air traffic is controlled at the level of a sector, which is a volume of airspace typically managed by a team of two air traffic controllers. Coordination across Europe ensures that the number of flights planned to fly through a sector at any one time is limited to a safe level (typically $20 - 35$ per hour).

(1.0.2) Aircraft fly along pre-planned routes according to their flight plan, which must be filed before the flight (typically 10 hours before). These routes further define how air traffic flows through the network. Flight plans are based on the fixed route structure and are used to calculate $4D$ trajectories: latitude, longitude, flight level (height) and time. These trajectories are used to predict where network problems may occur and the circumstances can change minute by minute.

(1.0.3) Usable airspace and the routes through it are limited by military areas, known as reservations, and also by bad weather. There are mechanisms to release the airspace for civil aviation when it is not used by the military, under the concept of the flexible use of airspace. Winsland is interested in exploring the value of non-available airspace and the mutual interactions that change this picture. This has the potential to inform operational planning, practices and airspace design that may improve the flexible use of airspace. This is of interest in the context of future systems where, for example, more direct flight routes will be possible. Currently there is mostly a fixed route structure, with flight plans based on fixed waypoints and optimised by Dijkstra's algorithm, to minimise flight time or cost.

(1.0.4) Our study question is whether there is an efficient way to correlate a $4D$ flight path (the civilian requirement) with a $4D$ volume of restricted airspace (due to weather or military use). This could be used to explore the network level impact of any particular airspace volume constraint. It could help explore whether the best shape of airspace reservations is being achieved, for example in respect of: sensitivity to civil demand (daily and seasonal); best configuration (shape and time); and the best times to release military sectors to civil traffic (network and local level).

(1.0.5) Answers to these questions would feed into European efforts to improve efficiency in the route network and airspace utilisation. If all parties understand the impact of their operational decisions, then there is the potential to finesse operations. Given the high number of flights per year, substantial savings ($10^7 - 10^8$) could accrue to EU air operators even with small improvements in flight efficiency.

Figure 1: One of 30,000 aircraft per day flying in European airspace....

2 Introduction

(2.0.1) Currently aircraft routing uses discrete waypoints, a fixed set of points connected by path segments, along which aircraft routes are planned, much like a road network. However in the future there may be a shift towards free routing, where aircraft can take any path through the open airspace.

(2.0.2) We will consider both the continuous and the discrete problem. The continuous problem approach works best when an approximation to the cheapest path is already known. Hence, the optimal method might be a combination of discrete and continuous approaches, where an estimate of the cheapest path found using a coarse discrete network is used as an initial guess for the continuous optimization algorithm.

(2.0.3) The main drawback with a purely discrete problem approach is that aircraft are constrained to fly along the segments in the network rather than taking the true cheapest paths. Adding more waypoints and segments to the network allows for more direct routes to be flown but increases the complexity of the system, both in terms of computational cost for path-finding as well as more practical issues such as reading maps and communication with ATC (Air Traffic Control) and hence a trade-off is necessary.

3 Discrete problem optimisation methods

3.1 Civil flights path intersection with Military Airspace (MA) analysis

(3.1.4) Currently military airspace has zones where civilian aircraft can't fly at certain times. We want to investigate the effect of military zones on flight cost. Initially we use MATLAB to plot the great circle routes, for each flight, between the departure and arrival airports and predict where the plane will be along this route at any specified time.

(3.1.5) We also made the following assumptions:

- Constant speed, determined by time of flight (difference between departure times and arrival times) and the distance between the departure and arrival airports.
- Constant height, determined by the flight level (FL).

(3.1.6) We plot the military zones in three dimensions as specified by the coordinates of the military zone and the flight levels it is between. We want to find the number of intersections between the flight paths and the military airbases. For convenience, we split the day into 48 thirty minute periods, and calculate the number of intersections in each period.

3.2 Program overview

(3.2.7) The key stages of the program are:

- Plot great circle flight routes, assuming uniform speed and height,
- Plot paths of great circle for each 30 minute period,
- For each path, work out how many airbases it intersects in 3 dimensions (so we include height of the airbase and height of the flight),
- Plot the data.

3.3 Input data

(3.3.8) We use flight data provided for a single day, in the form shown in table 1. The first column is the name of the flight, and the second and third columns are the codes for the departure and arrival airports. The fourth and fifth columns are the departure and arrival times, and the sixth column is the flight level of the flight. The seventh column is the actual distance

flown by the plane. The 8th column is the type of aircraft. The next two columns are the latitude and longitude of the departure airport, while the last two columns are the latitude and longitude of the arrival airport.

AA32719391	EGGW	LFLC	01:15:00	02:38:39	350	449.79	E145	51.87	-0.37	45.79	3.16
AA32725152	EGJJ	EGBB	09:30:00	10:31:26	290	279.53	C56X	49.21	-2.20	52.45	-1.75
AA32726897	USSS	EDDN	22:45:00	03:20:39	400	1853.61	F900	56.74	60.81	49.50	11.08
AA32727021	EGBB	EGJJ	10:45:00	11:36:16	280	217.69	C56X	52.45	-1.75	49.21	-2.20
AA32727184	EGNV	EGJJ	10:00:00	11:21:23	280	385.24	C510	54.51	-1.43	49.21	-2.20
AA32727510	EGJJ	LEIB	13:25:00	15:18:31	410	703.71	F2TH	49.21	-2.20	38.87	1.37
AA32730029	ESOW	EDHK	08:00:00	10:19:29	160	390.87	DA42	59.59	16.64	54.38	10.15
AA32732166	LIPZ	EDDS	08:55:00	10:00:33	320	344.79	GLF5	45.51	12.35	48.69	9.22
AA32732265	EDDS	ETAR	10:39:00	11:12:58	90	96.94	GLF5	48.69	9.22	49.43	7.60
AA32734378	EDDS	EDDS	15:15:00	16:56:04	70	82.28	P28A	48.69	9.22	48.69	9.22
AA32748315	LIRA	LIPZ	06:00:00	06:55:12	240	283.46	GLF4	41.80	12.60	45.51	12.35
AA32749220	LFBP	LFKS	09:00:00	11:17:28	180	495.47	CN35	43.38	-0.42	41.93	9.41
AA32749478	LFMI	LFPC	14:30:00	16:39:39	180	408.10	CN35	43.52	4.92	49.25	2.52
AA32755839	EGPD	EBBR	11:30:00	12:56:14	410	519.71	F2TH	57.20	-2.20	50.90	4.48
AA32756082	EGGP	LOAN	11:15:00	17:12:17	110	866.54	P68	53.33	-2.85	47.84	16.26

Table 1: Sample of flight data

(3.3.9) The military zone data is a sample of real world data. In practice there are about $75,000$ blocks of military airspace, each defined by its own coordinates we only consider 38 bases. Each airbase is defined by a block name, the flight levels the base is between and a list of latitude and longitude coordinates of the vertices which specify the polygon shape of the airbase. In Table 2 we give an example of data used to define airbases.

201LS	250	260	47.17	7.15	47.08	7.29	46.77	6.73	46.82	6.58	46.85	6.56	47.17	7.15
933LF	195	999	44.38	4.51	43.97	4.64	43.51	4.90	43.66	4.03	43.89	3.67	44.38	4.51
025LS	125	285	46.49	7.89	46.32	7.52	46.60	7.19	46.69	7.35	46.82	7.60	46.49	7.89
027LS	125	285	46.32	7.52	45.99	7.91	45.95	7.88	46.06	7.24	46.16	7.15	46.32	7.52
033LS	295	660	46.82	7.60	46.49	7.89	46.32	7.52	46.60	7.19	46.69	7.35	46.82	7.60

Table 2: Air base data format. First column is the block name, second and third columns are the minimum and maximum flight level of the base, and all subsequent columns are the latitude and longitude coordinates of the vertices.

3.4 Results of program

(3.4.10) Figure 2 shows the paths of each flight in a certain thirty minute period and the military airbases we are considering.

(3.4.11) We consider block 930LF which is part of the French airbase near the German border. We plot the number of flights going through this airbase at different times during the day, as shown in Figure 3.

(3.4.12) We can see that the best time to close this airbase would be between 9am - 11am and between 4pm - 6pm.

(3.4.13) We can also look at the other 38 air blocks and determine the number of the intersections in each thirty minute period. The colour of each airbase corresponds to the number of intersections, as shown in Figure 4. If an airbase is yellow this corresponds to zero intersections; to the other end of the scale red corresponds to the maximum amount of intersections.

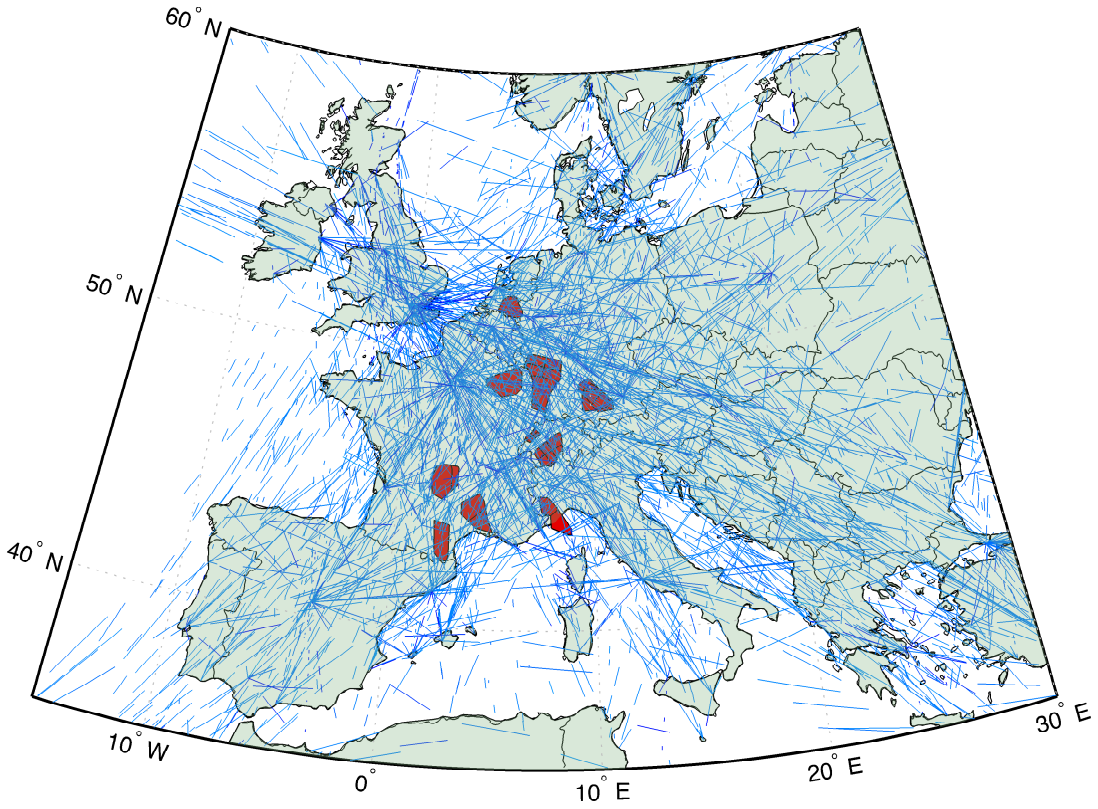


Figure 2: Plot of flights between 17.30 and 18.00 (blue) and 38 military airbases (red). The corresponding video shows how the flights change throughout the day.

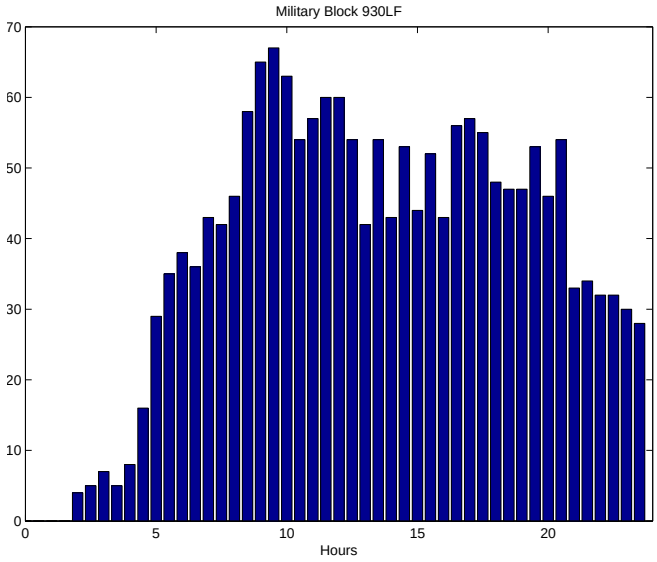


Figure 3: Plot of intersections over time for military block 930LF.

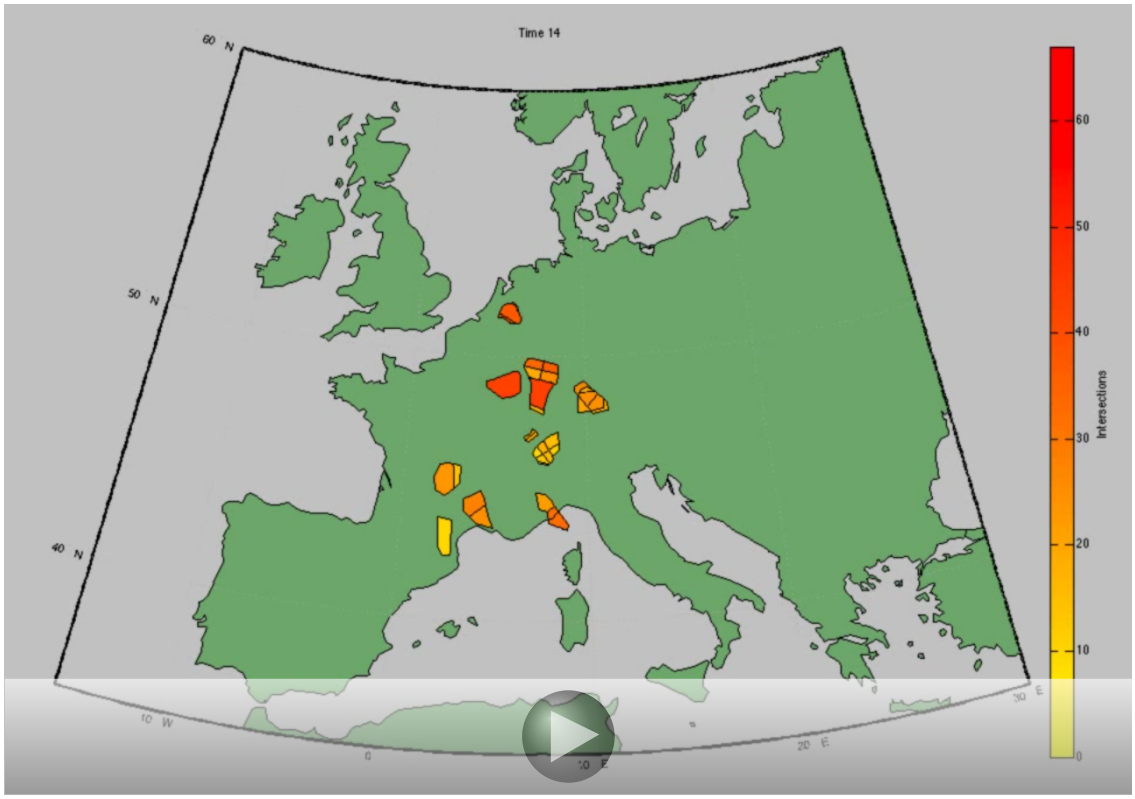


Figure 4: Colour scale representation of the number of flights crossing each airbase

3.4.1 Effect of moving a military airspace

(3.4.14) We next consider the effect of moving a military airspace by a small amount of latitude or longitude. We shift the airspace by half a degree of latitude or longitude, which corresponds to moving military airspace by about 30 miles along the compass points. In Figure 5 we can see the effect of moving this airspace.

(3.4.15) We then consider how many intersections each shifted airspace has accrued compared to the original. The idea would then be to shift the airspace in the direction with fewest intersections.

(3.4.16) In Figure 6 we can see a colour coded plot of the number of intersections for the original base and the shifted version. We would want to shift the airspace in the direction of most yellow in the figure, since this corresponds to fewest intersections.

(3.4.17) One small improvement that needs to be made is that currently if a 30 minute path lies entirely within a military airspace then this won't count as an intersection.

(3.4.18) In the future some of the optimal path algorithms described later on could be used, so that if a certain airspace is closed then a great circle route

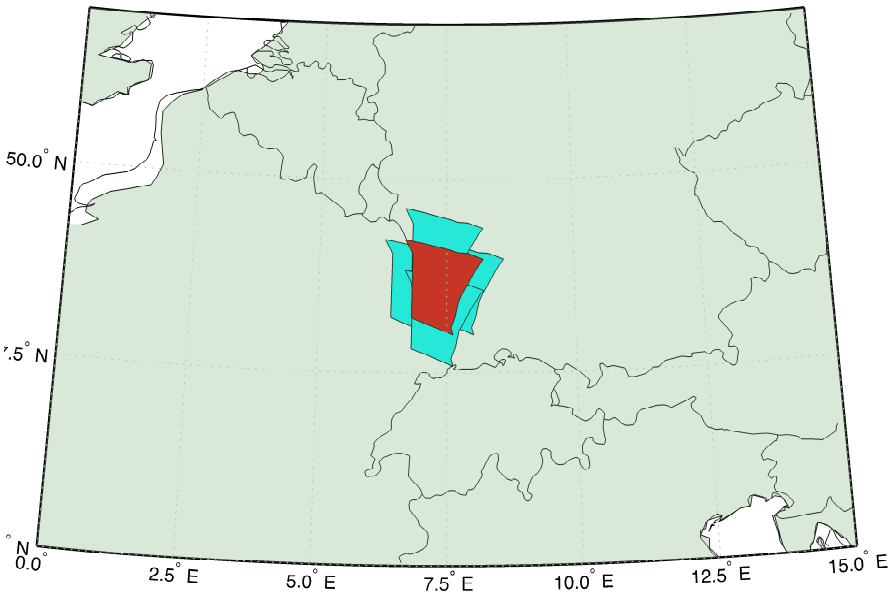


Figure 5: Moving an airbase by half a degree of latitude and longitude.

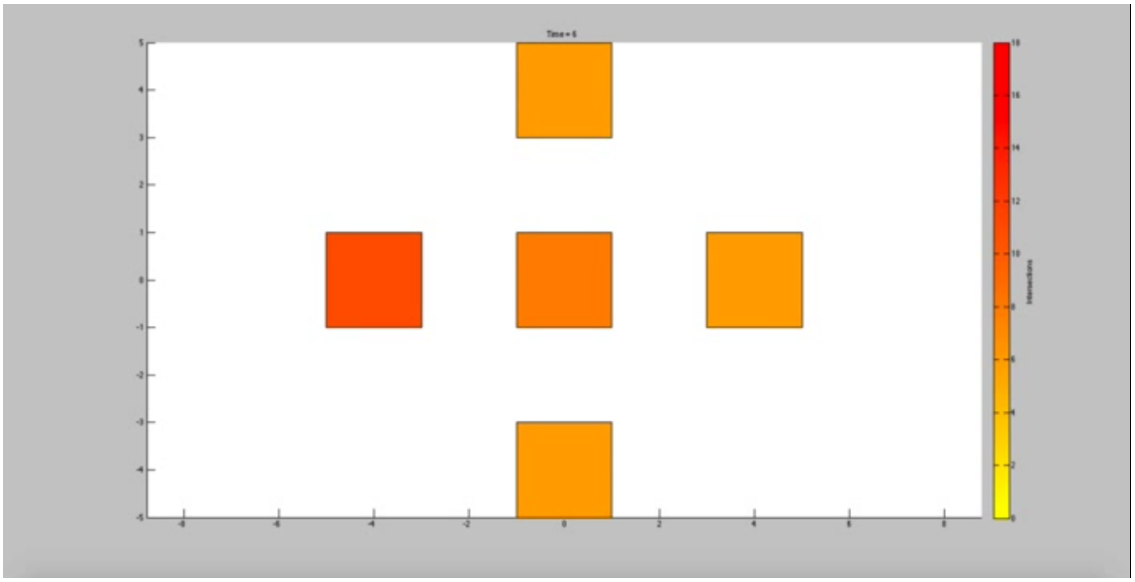


Figure 6: Number of intersections if the military airspace is shifted in different directions. The centre square corresponds to the number of intersections in the initial position.

should not be used but a new route around that airspace. We could then calculate the number of intersections when we have a certain number of military airspace reservations closed.

3.5 Main approximations

(3.5.19) In order to quantify how altering military airspace closures reduces costs for civil aviation, we need to be able to calculate these costs for any given airspace configuration. This involves predicting which route each aircraft will take, i.e.calculating the cheapest route from point A to point B based on the costs discussed in section 6. In this section, we investigate how to calculate both the least-cost routes for aircraft and the reduction in costs when military airspace is open for civil air traffic.

(3.5.20) The purpose of our modelling is to understand how the cost varies with different route choices. As the main aim is to quantify how varying military airspace closures affect civil air traffic costs, we can allow approximations which simplify the problem without having too large an impact on the end result.

(3.5.21) One such approximation is to ignore aspects which do not change appreciably with the choice of route, such as the non-cruise phases of flight (take-off, ascent, descent and landing), which typically add a constant cost on top of the cruise phase costs. For our purposes, taking each flight to be from the source airport to the destination airport at a constant speed (and flight level) is thus a reasonable approximation.

(3.5.22) As was discussed in section 6, the ATC zone cost paid by airlines is not proportional to the actual distance flown, but rather the straight-line (great-circle) distance between the planned entry and exit points of each zone. For simplicity, we can use the following approximations:

- for short flights (between airports in the same ATC zone) we assume that the flight does not leave the zone and hence the cost does not vary with the route choice.
- for long flights (between airports in different zones) we assume that the path through each zone is relatively direct, so that the great-circle distance can be replaced by the actual route length in the formula for cost, yielding simply an additional cost proportional to the distance travelled, with different proportionality constants in different zones.

(3.5.23) For unpredictable issues, such as sudden weather events or delays on the ground, we assume that they can be replaced by their average effect e.g. adding a cost penalty for routes through regions which have unstable weather and may require taking a detour.

(3.5.24) A final, more crude approximation, is to ignore conflicts between aircraft i.e. congestion issues due to network capacity constraints. This allows us to treat each flight route independently from all the others, greatly simplifying the problem in two ways: firstly, the problem requires less computational power and secondly, the computation for different flights

can be done in parallel on different processor cores.

(3.5.25) Employing this last approximation means that we cannot investigate the effects of bottleneck regions between nearby airspace closures, where large savings could be made if more space were opened up so that aircraft would not need to wait or choose a different path (figure 7(a)). However, the model does capture approximately the savings of congested aircraft obstructed by single military zones (figure 7(b)).

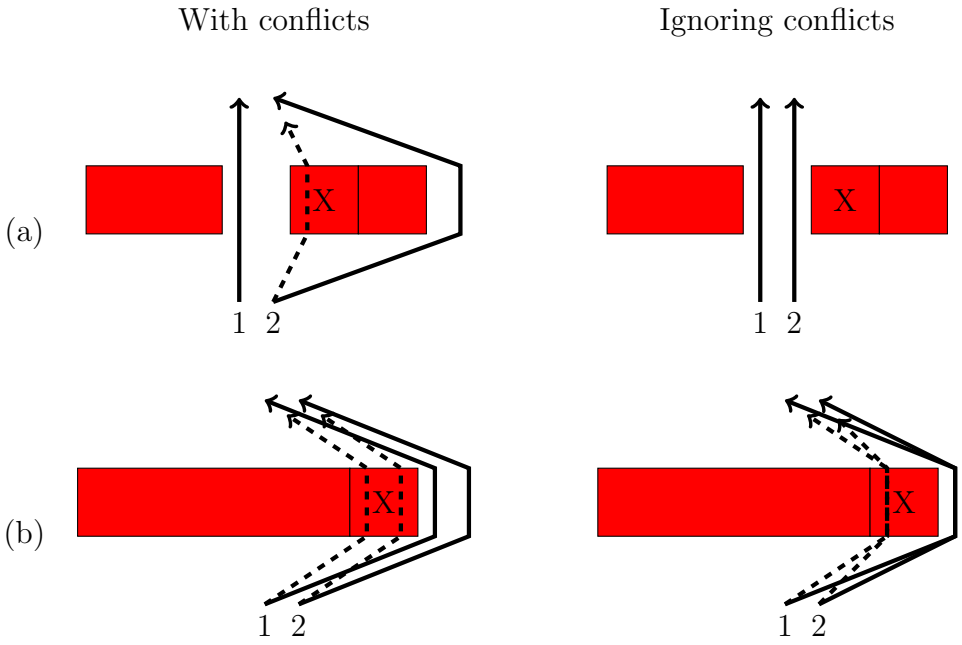


Figure 7: Effects of ignoring conflicts between aircraft. Due to capacity constraints, flight 1 is preventing flight 2 from taking the most optimal route avoiding the military closures (red), but treating each flight independently does not capture this. The estimates for the benefit of opening up block X (resulting in the shorter dashed routes) are incorrect in case (a), but decent in case (b).

3.6 Waypoint-based routing

(3.6.26) Air traffic over Europe currently employs waypoint-based routing: a fixed set of waypoints connected by path segments, along which aircraft routes are planned, much like a road network. We can represent the network mathematically as a graph, where the nodes are the waypoints and the edges are the segments (see figure 8). In this section, we analyse how to solve our problem in this framework.

3.6.1 Two-dimensional model

(3.6.27) A full model would have to include the aspects of time and flight level, but a

simplified two-dimensional model, which only takes latitude and longitude into account, can nevertheless give useful insight into how military airspace closures affect civil air traffic. A discussion of how to incorporate time and flight level into this model is given in section 3.6.6.

(3.6.28) Throughout the rest of this section, we use the term "distance" to refer to the costs of travelling along the network. This is just for convenience of language, and should not be interpreted as a restriction of the analysis to minimizing physical distances only. For the example results, due to lack of better data, we have taken the costs to be the actual shortest-path (great-circle) distances, but this again is not a restriction on the generality of the analysis.

(3.6.29) Another simplification we have made is to assume that distances are symmetric.

3.6.2 Basic path-finding

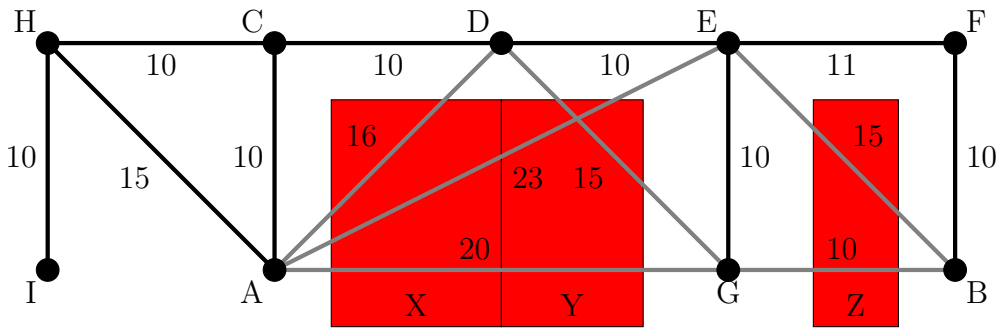


Figure 8: Example network. (Costs are approximate distances with some adjustment.)

(3.6.30) We are given a graph, like in figure 8, and wish to find the shortest (i.e. cheapest) path from e.g. A to B (without using the segments passing through the red military zones). The search can be accomplished efficiently using the well-known Dijkstra's algorithm[1]. The algorithm calculates the shortest path from the starting node A to any node in the network by processing each node in turn. At each step, the next node to process is chosen to be the node closest to A which has not yet been processed. This guarantees that every node needs to be processed at most once, and the process can be terminated as soon as B has been processed.

(3.6.31) The resulting information obtained by the algorithm is shown in figure 9. Each node has been labelled by its shortest distance to the starting node, as well as which direction the shortest path is in. For example, the

[1] `http://en.wikipedia.org/wiki/Dijkstra's_algorithm`

shortest path from A to B is found by following the arrows from B back to A.

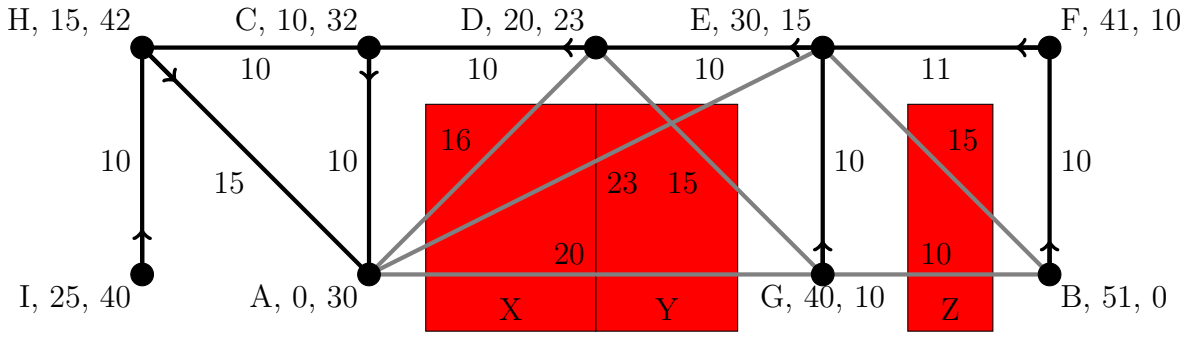


Figure 9: Basic results for the example network. Each node is labelled with its shortest distance to A, found by Dijkstra's algorithm, and the heuristic (straight-line distance to B) used by the A* algorithm. The arrows from each node indicate which node precedes it in the shortest path from A.

(3.6.32) One drawback of Dijkstra's algorithm is that it processes all nodes in order of increasing distance from A until the goal node B is found (A, C, H, D, I, E, G, F, B in the example). Thus, the algorithm processes all nodes which are closer to A than B is before getting to B, including nodes which are "obviously" in the wrong direction (H and I in the example). Pictorially, Dijkstra's algorithm processes nodes in an expanding circle until B is reached.

(3.6.33) An improvement on Dijkstra's algorithm is called the A* (A-star) algorithm[2], and makes use of an extra piece of information we know about each node M: a lower bound on its distance to B. This information is called a "heuristic". If the "distances" used here are physical distances, then a suitable heuristic would be the shortest-path (great-circle) distance from M to B. For a more sophisticated model including costs and wind, this heuristic could for example be the cost of flying the shortest path from M to B, with the maximum tailwind and minimum ATC path costs possible in the whole network.

(3.6.34) The A* algorithm works like Dijkstra's algorithm, except that it processes nodes M in order of increasing estimate of total route length (distance from A to M plus heuristic for distance from M to B) rather than just increasing distance from A to M. Using the heuristic values shown in figure 9, we obtain the same result as for Dijkstra's algorithm, but the nodes are processed in order of increasing distance plus heuristic (A(30), C(42), D(43), E(45), F(50), B(50) in the example), and the search terminates before H(57) or I(65) are processed.

[2] `http://en.wikipedia.org/wiki/A*_search_algorithm`

(3.6.35) Pictorially, the A* algorithm searches in a "beam" from A towards B. If there are no obstacles, then the beam is very narrow and the search is extremely efficient. If obstacles are encountered, the beam progressively widens until a path is found around the obstacle. The beam is always contained inside the circle searched by Dijkstra's algorithm, so A* never processes more nodes than Dijkstra.

3.6.3 Example results

(3.6.36) Here, we show results for flights in one day over Europe, using a real dataset of 29,638 waypoints and 37,316 segments, of which 3,953 pass through military airspace. The optimal paths are computed using Dijkstra's algorithm.

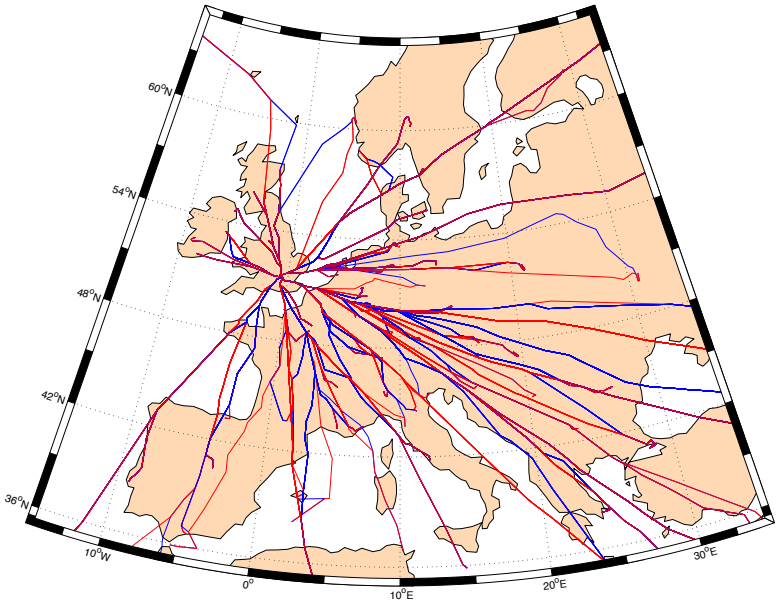


Figure 10: Optimal flight routes using actual waypoints and segments for scheduled flights departing from London Heathrow with all military airspace closed (blue) and open (red).

(3.6.37) We first quantify the monetary value in our model of all of Europe's military airspace. To do this, we calculate the optimal routes avoiding military airspace, and compare with the optimal routes passing through military airspace if necessary. Assuming constant cruising speed and considering fuel costs alone, this gives an approximate average cost saving of $36.05 per flight. We plot in Fig. 10 the different routes determined in the two cases for all of the day's flights departing from London Heathrow Airport, illustrating how the opening of the military airspace can allow the civil airlines to fly shorter routes.

(3.6.38) This approach can also be used to estimate the cost savings that could be attained by opening separately any of the military airspaces. For example, when the German airspaces 852–855ED are opened, this method shows that 2,169 of the day's flights can then be rerouted (the optimal routes of the remaining 30,358 flights are unchanged as they cannot be improved by routing through the newly-opened airspace) at an average cost saving of \$22.22 per flight. Analogously, we could also obtain explicit estimates for the cost savings due to opening particular combinations of segments passing through military airspace. This could allow deductions to be made about which conditional segments it would be most desirable to open, giving indications of how the military airspaces might be reshaped so as to yield maximum savings.

3.6.4 Quantifying savings from opening up military airspace

(3.6.39) As for the question of how releasing parts of military airspace would shorten a given route, a brute-force approach would be to consider every release alternative and perform the path-finding calculation above on each case. However, when considering the European network as a whole, there are many alternatives to consider – there are several different reservations, and one might be interested in options to release parts of them for different parts of the day and at different flight levels. Hence, recalculating the optimal path for each alternative individually would be a very costly operation.

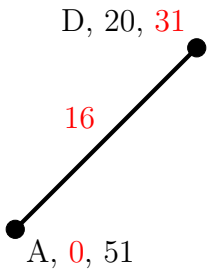
(3.6.40) A first obvious optimization to make is to note during the first path-finding for the route, which reservations were reached by the search algorithm. Releasing any other reservations would have no benefit for the route in question. Again, the A* algorithm does a more focused search than Dijkstra, and hence leaves fewer alternatives to consider here.

(3.6.41) The second, and main optimization, is to perform two searches (avoiding all reservations): One from A to B and one from B to A. This yields, for every potentially interesting node, its shortest distance from A and its shortest distance to B. After this information is obtained, we use the following method to determine the savings from opening up any collection of routes through military airspace:

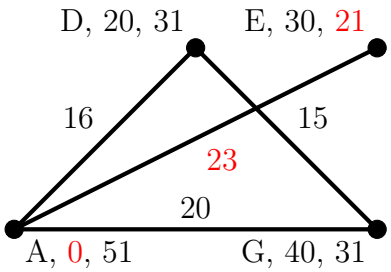
(3.6.42) We look at the subgraph containing only the segments that change from closed to open. The task is to find the two nodes M and N which yield the lowest value of

$$\begin{array}{ccccc} \text{distance from} & & \text{distance from} & & \text{distance from} \\ \text{A to M} & + & \text{M to N} & + & \text{N to B} \\ \text{(pre-calculated)} & & \text{in subgraph} & & \text{(pre-calculated).} \end{array} \tag{1}$$

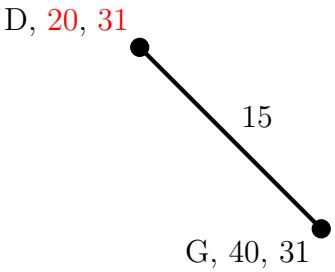
(3.6.43) We do this by re-initializing a Dijkstra or A* search on the subgraph using the pre-calculated distances from A. Each node in the subgraph is then processed (in order of increasing distance or distance plus heuristic) and once the least distances from A to nodes N in the subgraph have been found (i.e. the first two terms in equation (1)) we add the pre-calculated distances from N to B to obtain the overall distance. The minimum of these is the least distance from A to B which makes use of the subgraph at most once.



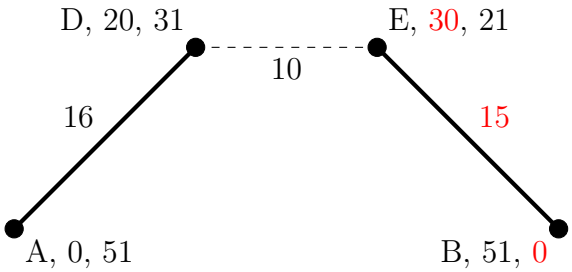
(a) X: $0 + 16 + 31 = 47$



(b) X, Y: $0 + 23 + 21 = 44$



(c) Y: $20 + 0 + 31 = 51$



(d) X, Z: $30 + 15 + 0 = 45 \neq 16 + 10 + 15$

Figure 11: Path-finding on subgraphs of opened segments. Each node is labelled by the pre-calculated shortest distance from A and to B using the already open segments. The algorithm finds the shortest path using the numbers in red.

(3.6.44) Some results for the example network from before are shown in figure 11. The method correctly predicts the savings that can be made by cutting the corner if (a) X or (b) both X and Y are opened up. Importantly, it also detects that opening up (c) only Y has no benefit when X and Z remain closed (whereas the naive method which only looks at shortest paths ignoring closures would indicate that Y is obstructing the route from A to B and imply that opening up Y would yield some saving). However, if (d) X and Z are opened, then the new shortest path needs to pass through the subgraph two separate times. The algorithm can not find such paths, and hence it underestimates the savings for this option. This can be remedied by using a more sophisticated algorithm that re-evaluates not only the subgraph but also other nearby edges.

(3.6.45) To obtain the savings for multiple flights, e.g. all flights on a given day, we would simply apply this algorithm on each flight and add up the savings

for each alternative.

3.6.5 Example results

(3.6.46) Instead of using the actual military airspace shapes and waypoint locations, we divided the airspace into grid squares which can be either closed or open for civil air traffic, and placed artificial waypoints on the midpoint of each edge. The four waypoints around each grid square were then joined by segments. This was done simply to avoid having to choose how to divide the airspace, and is not exploited by the path-finding algorithms in any way.

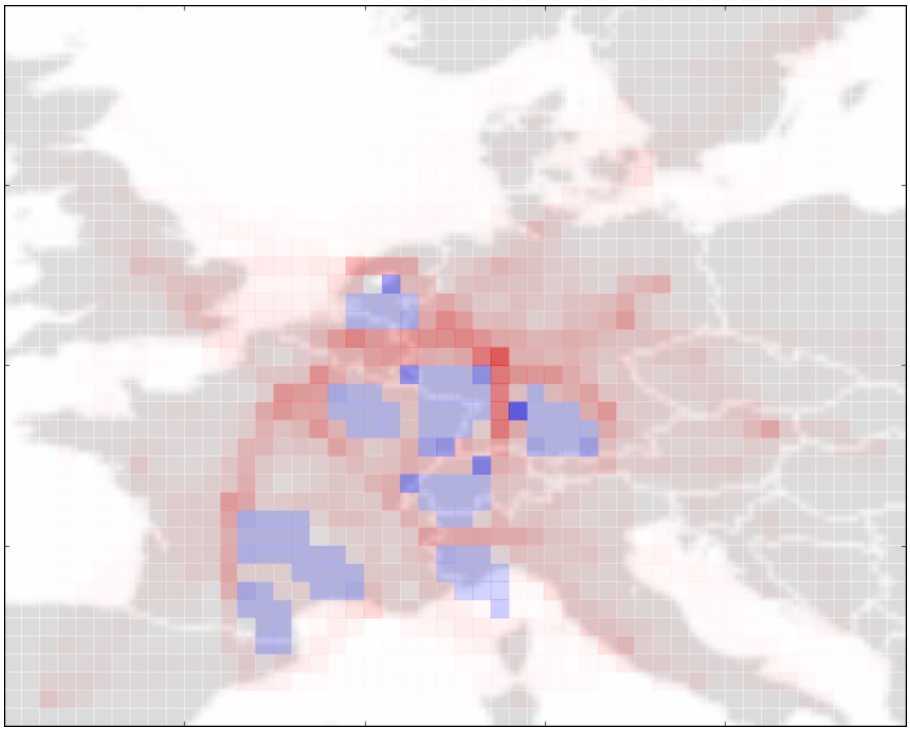


Figure 12: Results of shortest-path calculations with quantification of savings for a sample of $1,000$ flights. The blue regions are military airspace, with darker shading indicating larger potential savings (up to 0.25% of the total cost of all flights in the network) from opening up the airspace for civil traffic. The red shading in non-military airspace indicates high density of flights (up to 125).

(3.6.47) An example result is shown in figure 12. The grid spacing is $0.5°$, corresponding to around 50 km vertically and slightly less horizontally. The calculation for $1,000$ flights on the grid with approximately $4,000$ waypoints and $12,000$ segments took 10 seconds to complete on an average laptop.

3.6.6 Effects of altitude and time

(3.6.48) Introducing time in the model brings two implications: firstly, the route conditions (e.g. wind field and airspace closures) may vary with time. Secondly, we may want to know about potential savings from opening up some airspace only part of the time.

(3.6.49) For the first aspect, the simplest approach is to take the conditions at a distance L from the airport to be the predicted conditions for the fixed time L/v_c (or perhaps slightly later) after the scheduled departure time, regardless of when the aircraft actually arrives at that location. This removes the time aspect of the problem, which can then be solved using the algorithms above. The approximation is reasonable for aircraft taking relatively direct paths in slowly changing conditions (so that the mismatch between the actual conditions when the aircraft passes and the assumed conditions is small). The approximation would fail e.g for cases when an aircraft could fly slow on purpose in order to wait for a direct route through military airspace to open up, but taking such cases into account would come at a very large cost for not very much benefit.

(3.6.50) To deal with the second aspect, the algorithms would be modified slightly to also keep track of travel times (along the shortest paths found). The Dijkstra or A* calculation then yields both the least distance from A to each relevant node M and when the aircraft is scheduled to arrive at M. When evaluating savings from opening up military airspace, we readily obtain which times the airspace needs to be open for the aircraft to pass.

(3.6.51) Finally, in order to deal with different altitudes, a full model would duplicate each waypoint into multiple waypoints representing different altitudes. Segments would join neighbouring waypoints with similar altitudes (limited by the maximum climb or descent rate) and have different costs associated with them depending on whether they represent a climb or a descent and also depending on the fuel efficiency at each altitude.

(3.6.52) A simpler model would ignore maximum climb or descent rates so that no duplication of waypoints is necessary. The segments connecting the waypoints, however, remain duplicated, and represent flying at various altitudes with various costs. Duplication of segments has a very small effect on the path-finding algorithms, and they should execute in almost the same time as for the case without duplication. In order to take into account take-off and landing, high-altitude segments near the departure and arrival airport can be removed from consideration. Good qualitative results can be obtained from considering only two altitudes : "low" and "high". Closing low-altitude segments forces aircraft connecting to airports nearby to take detours, while closing high-altitude segments forces aircraft passing by to either take a detour or fly at the costlier lower altitude.

3.6.7 Optimal choice of waypoint locations - visibility graph

(3.6.53) When aircraft are constrained to follow a network of waypoints and segments, the locations of the waypoints has a large effect on how far the aircraft must deviate from the cheapest possible route (ignoring the network) and hence on the overall cost efficiency. Since the computational cost of discrete path-finding algorithms increases with the number of waypoints, we would like waypoints to be located so that we obtain the best results (i.e. the cheapest total cost for flights following the network) as possible using as few waypoints in total as possible.

(3.6.54) In the idealized case when the costs of flights are simply given by the distance flown and all airspace is open, there would only need to be waypoints at every airport and all aircraft would fly directly from their departure waypoint to their arrival waypoint. With military airspace closures (which are assumed to have polygonal shapes) the optimal waypoint and segment distribution is given by the visibility graph in Fig. 13. Applying the path-finding algorithms to this graph always yields the cheapest route (even if not following the network).

(3.6.55) Some open questions are: (a) If we wish to reduce the number of waypoints further, which waypoints should be removed to yield the least impact on the cost? (b) How should the waypoints be located if more effects are taken into account, such as wind or opening up of military airspace?

3.6.8 Airspace congestion – linear programming approach

(3.6.56) The shortest-path problem for a single aircraft can be expressed as a linear-programming problem as follows:

$$\text{Find } x_{i,j} \geq 0 \text{ which minimize } \sum_{i,j} C_{i,j} x_{i,j}, \tag{2}$$

subject to the constraint that for each waypoint i:

$$\sum_j x_{i,j,f} - x_{j,i,f} = \begin{cases} 1, & \text{if } i \text{ is the departure airport,} \\ -1, & \text{if } i \text{ is the arrival airport,} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

(3.6.57) Here, we interpret the airway network as a network of tubes, through which a fluid (representing the aircraft) flows. The decision variable $x_{i,j}$ is the flux of fluid along the segment $i \to j$, and $C_{i,j}$ is the cost per unit of fluid of flowing along this segment (equal to the cost of flying along this segment for the aircraft). The constraint (3) expresses that one unit of fluid is pumped into the system at the departure airport and extracted at the

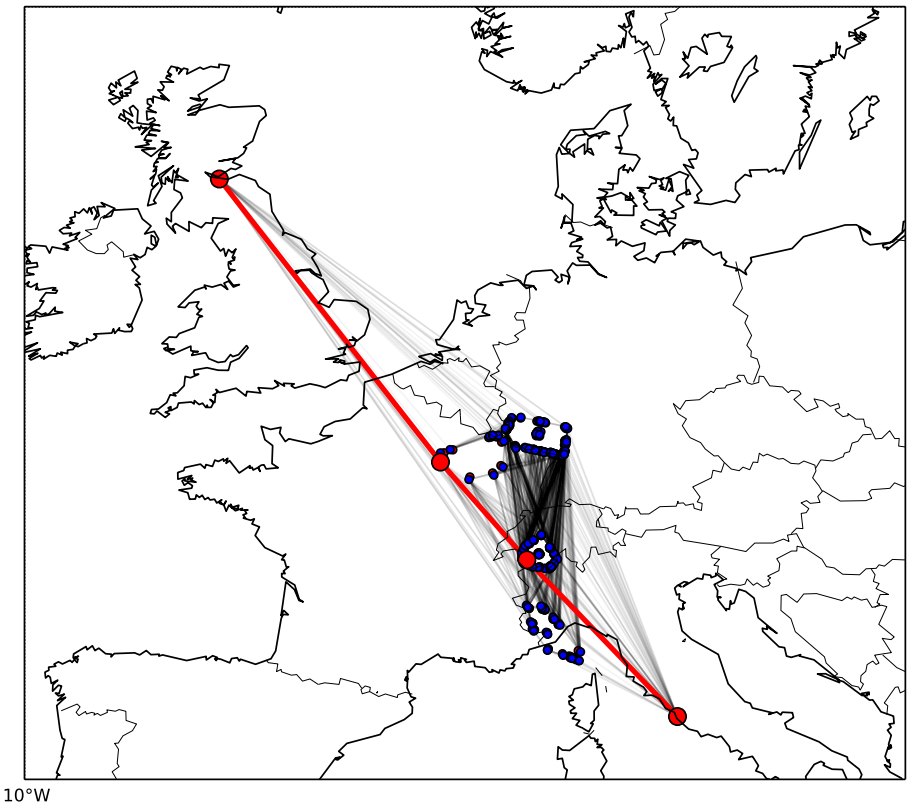


Figure 13: An example of a visibility graph approach to path-finding. An aircraft travelling from Edinburgh to Rome cannot go in a straight line (great circle) as military bases are blocking the route. To find the optimal flight path we construct a graph whose nodes are the origin and destination and corners of the military bases and whose edges are all links that do not intersect any military base. After running a path-finding algorithm (e.g. Dijkstra's algorithm) on the graph we find that the shortest path consists of piecewise straight lines avoiding the military bases at the corners.

arrival airport, while, at any other waypoint, the amount of fluid flowing in must equal the amount of fluid flowing out. Although the variables $x_{i,j}$ are allowed to be real numbers, Ahuja et al.[3] have shown that for (certain) optimal solutions, each $x_{i,j}$ is either 1 or 0, corresponding to whether the optimal aircraft route runs along the segment $i \to j$ or not, respectively.

(3.6.58) The linear-programming formulation is equivalent to the shortest-path formulation, and hence represents no improvement on its own. However, it can be generalised to tackle the issue of capacity constraints when multiple flights require use of the same segments:

$$\text{Find } x_{i,j,f} \geq 0 \text{ which minimize } \sum_{i,j,f} C_{i,j,f} x_{i,j,f}, \tag{4}$$

[3]Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin (1993). Network Flows: Theory, Algorithms and Applications. Prentice Hall. ISBN 0-13-617549-X.

subject to the constraint that for each waypoint i and flight f:

$$\sum_j x_{i,j,f} - x_{j,i,f} = \begin{cases} 1, & \text{if } i \text{ is the departure airport of flight } f, \\ -1, & \text{if } i \text{ is the arrival airport of flight } f, \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

and for each segment $i \to j$:

$$\sum_f x_{i,j,f} \leq M_{i,j}. \tag{6}$$

Here, $M_{i,j}$ is the maximum number of flights the segment $i \to j$ can handle. Capacity constraints on the total number of flights across a collection of segments (e.g. all segments in a single ATC sector) can also be implemented.

(3.6.59) Constraint (4) then ensures that flights that arrive at node j from node i must also leave that node unless that node presents an airport. If node i represents an arrival airport then flight f can only leave that node while if node j is an arrival airport then flight f can only arrive at that node.

(3.6.60) One crucial issue is that when modelling airspace congestion, the passage of time can no longer be ignored – it is important to know whether the flights passing through a sector in one day are spread out in time or if they intend to all be in the same place at the same time. Time can be incorporated in the model by, like in the case with altitudes, duplicating each waypoint into multiple waypoints representing different instances in time. In the end, the optimal solution may turn out to be very computationally expensive to calculate.

4 Continuous problem approach

4.1 Model

(4.1.1) Suppose that we can represent a flight path in the form $y = y(x)$ and aim to find the path that minimises the cost functional

$$C[y] = \int_{x_A}^{x_B} \phi(x, y)\sqrt{1 + y'^2}\,\mathrm{d}x. \tag{7}$$

The cost function ϕ represents the cost per unit distance at the location (x, y). If we are just interested in total distance as a cost metric, then $\phi = 1$ in open airspace. Military airspace can be given an artificially high ϕ, possibly continuously decreasing to 1 around the edges. The radial functions including a piecewise-linear top-hat, a Gaussian and an error function were tested. All functions used were adequate.

(4.1.2) One approach is to minimise the cost functional using the Euler-Lagrange equation

$$\frac{\partial L}{\partial y} = \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial L}{\partial y'}\right) \tag{8}$$

where $L = \phi\sqrt{1 + y'^2}$ is the Lagrangian. The resulting ordinary differential equation (ODE) is

$$\phi y'' = (1 + y'^2)\left(\frac{\partial \phi}{\partial y} - y'\frac{\partial \phi}{\partial x}\right) \tag{9}$$

with boundary conditions $y(x_A) = y_A$, $y(x_B) = y_B$.

(4.1.3) A program (`potentialRoute`) was developed in MATLAB, incorporating the `bvp4c` routine, to solve this equation for a given cost function ϕ and endpoints $\mathbf{x}_A$ and $\mathbf{x}_B$.

(4.1.4) As a small improvement to working in fixed Cartesian coordinates, the coordinates y and x are defined as normal and parallel to the straight line joining the start and end points of the flight. The `potentialRoute_normal` code rotates into and out of this frame for each flight path, returning a solution path in fixed Cartesian space. Using the code we computed a few illustrative examples.

4.2 Assumptions

(4.2.5) Cartesian coordinates are used.

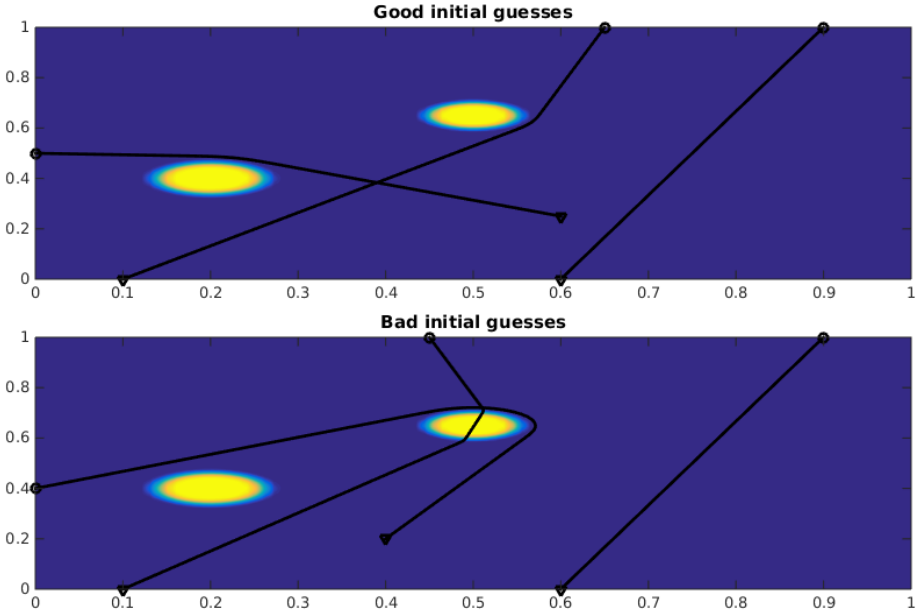
(4.2.6) The path is represented as a single-valued function. An improvement would be to use a parametric form $(x(t), y(t))$. We then get the Euler-Lagrange equations

$$\frac{\partial L}{\partial x} = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{x}}\right), \qquad \frac{\partial L}{\partial y} = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{y}}\right). \tag{10}$$

(4.2.7) The problem is that the Euler-Lagrange equations will give a result that if the problem is time-invariant, doesn't depend on the parameterisation. That is, we could let $t = t(s)$ and still have a solution.

(4.2.8) Time-independence: if we can parameterise the flight path using real time (rather than use t as an arbitrary parameter) then it would be simple to make the cost function ϕ time-dependent and modify the Euler-Lagrange equations accordingly.

(4.2.9) Maximisation vs minimisation: the Euler-Lagrange equations don't distinguish between maxima and minima. In the formulation given, this can occasionally lead to finding a stationary point of the cost function which is not a desired minimum (i.e. going right through military space). This could be fixed by reformulating the problem to minimise travel time (suitably penalised according to winds etc).



4.3 Model includes wind

(4.3.10) A different approach considers the effect of wind. Suppose that we aim to

minimise the flight time

$$T = \int_0^\ell \frac{\mathrm{d}s}{v} = \int_{x_A}^{x_B} \frac{\sqrt{1+y'^2}}{v(x,y,y')}\,\mathrm{d}x \tag{11}$$

where v is the speed of the aircraft.

(4.3.11) If we assume, crudely, that the aircraft flies at fixed unit speed relative to the wind, then we could write

$$v(x,y,y') = 1 + \frac{w_x + y' w_y}{\sqrt{1+y'^2}}, \tag{12}$$

where $\mathbf{w}(x,y) = (w_x, w_y)$ is the wind velocity scaled by the aircraft's cruising speed. The total time is then

$$T[y] = \int_{x_A}^{x_B} \frac{1+y'^2}{\sqrt{1+y'^2} + w_x + y' w_y}\,\mathrm{d}x. \tag{13}$$

(4.3.12) This should be minimised with respect to $y(x)$ satisfying the endpoint conditions $y(x_A) = y_A$ and $y(x_B) = y_B$.

(4.3.13) Obstacles (military zones) could be included in this formulation via multiplication by a penalty function ϕ as before. However, unless the cost function represents obstacle edges as sharp steps, the functional will no longer represent the true flight time.

4.3.1 Effects of wind during cruise

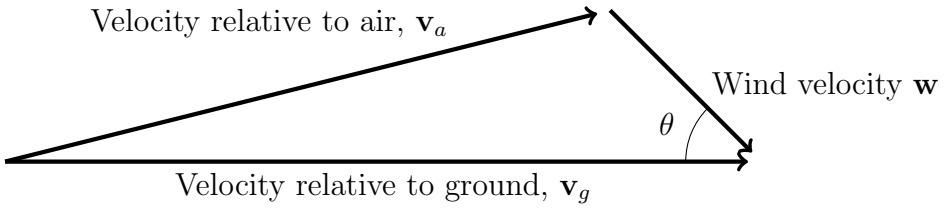


Figure 14: The wind triangle.

(4.3.14) When an aircraft flies in a horizontal wind $\mathbf{w}$, its ground velocity $\mathbf{v}_g = \dot{\mathbf{x}}$ and air velocity $\mathbf{v}_a$ are related by the wind triangle (figure 14):

$$\mathbf{v}_g = \mathbf{v}_a + \mathbf{w} \qquad \Rightarrow \qquad \begin{cases} v_a &= \sqrt{v_g^2 - 2v_g w\cos\theta + w^2}, \\ v_g &= \sqrt{v_a^2 - w^2\sin^2\theta} + w\cos\theta, \end{cases} \tag{14}$$

(4.3.15) Here θ is the angle between the wind and the ground velocity, with $\theta = 0$ corresponding to a tailwind. We begin by investigating this expression

for flight in a straight line with constant wind, to help us understand the effects of the wind. Also, the straight path is the optimal path between very nearby (infinitesimally close) points, so the results can be used for our general analysis later.

4.3.2 Optimum air speed

(4.3.16) Aircraft usually fly at a cruise speed v_c, which we presume is chosen to minimise the cost of operating the aircraft (e.g. fuel costs and staff costs). Here we investigate whether the presence of wind may change the optimal flying speed.

(4.3.17) We let the unknown function $C(v)$ denote the cost per unit time of flying the aircraft at air speed $v_a = v$. The cost per unit distance flown (without wind) is thus $C(v)/v$, and the cruise speed $v = v_c$ minimises this expression. With wind, the cost per unit distance flown along the ground is then $C(v_a)/v_g$ (with v_g related to v_a by equation (14)), and we immediately observe that this might attain a minimum at a value of v_a different from the cruise speed v_c.

(4.3.18) To make more progress, we use a perturbation expansion in the limit of small wind, $w \ll v_c$, with the aircraft flying at near-cruise speed. That is, we let $\epsilon = w/v_c \ll 1$ be a small parameter, and expand the air speed as $v_a = v_c[1 + \epsilon\hat{v}_{a1} + \epsilon^2\hat{v}_{a2} + O(\epsilon^3)]$. This yields the ground speed

$$v_g = v_c \left[1 + \epsilon(\hat{v}_{a1} + \cos\theta) + \epsilon^2(\hat{v}_{a2} - \tfrac{1}{2}\sin^2\theta) + O(\epsilon^3)\right]. \tag{15}$$

(4.3.19) Since the cruise speed v_c minimizes $C(v)/v$, we can write down the Taylor expansion

$$\frac{C(v)}{v} = A\left[1 + \frac{B}{v_c^2}(v - v_c)^2 + O((v - v_c)^3)\right], \tag{16}$$

where

$$A = \frac{C(v_c)}{v_c} \qquad \text{and} \qquad B = \frac{v_c^2 C''(v_c)}{C(v_c)} - \frac{2v_c C'(v_c)}{C(v_c)} + 2 \tag{17}$$

are positive constants depending on the aircraft operating cost characteristics. A is the cost per unit distance at cruise speed, and B measures how sensitive the cost is to variations in speed. The cost per actual distance flown is thus

$$\frac{C(v_a)}{v_g} = \frac{C(v_a)}{v_a}\frac{v_a}{v_g} = \tag{18}$$

$$= A\left[1 - \epsilon\cos\theta + \epsilon^2\left(B\hat{v}_{a1}^2 + \hat{v}_{a1}\cos\theta + \tfrac{1}{2}\cos^2\theta + \tfrac{1}{2}\right)\right]. \tag{19}$$

(4.3.20) We immediately observe that the leading-order effect of the wind (i.e. the first two terms in equation (19)) is not affected by the choice of $\hat{v}_{a1}$:

$$\frac{C(v_a)}{v_g} = A\left[1 - \frac{w}{v_c}\cos\theta + O\left(\frac{w^2}{v_c^2}\right)\right]. \tag{20}$$

(4.3.21) For the purposes of calculating costs we can thus assume that all aircraft are flying at cruise airspeed. We can understand these results intuitively as follows: let's first suppose that the aircraft flies exactly at cruise speed. The presence of wind then alters the length of the path the aircraft has to fly according to the wind triangle and this change in distance corresponds directly to a change in travel costs (via the proportionality factor $C(v_c)/v_c$). Changing the speed of the aircraft slightly has a small effect on the proportionality factor $C(v)/v$ since its derivative is zero near v_c and a small effect on the length change and so a small effect on the overall cost.

(4.3.22) For completeness, we can still minimize the third term in equation (18) to find the optimal choice

$$\hat{v}_{a1} = -\frac{\cos\theta}{2B} \qquad \Rightarrow \qquad v_a = v_c - \frac{w\cos\theta}{2B} + O\left(\frac{w^2}{v_c^2}\right). \tag{21}$$

(4.3.23) Hence, it is very slightly cheaper to fly slightly slower than cruise speed in tailwind, and slightly faster in headwind.

4.3.3 Cost for flying at cruise speed with wind

(4.3.24) We have argued in the previous section that for wind speeds much slower than v_c, the effect of wind is to alter the distance that the aircraft flies relative to the moving air and this does not depend on whether the aircraft is flying at the optimal speed or at cruise speed v_c. Also, aircraft in reality probably do not adjust their air speed greatly to take wind into account but rather keep flying near cruise speed. Hence, for the rest of this report, we will assume that the air speed of each aircraft is the (aircraft-dependent) cruise speed v_c (which converts to a ground speed via equation (14)). The path of an aircraft in 4D spacetime is then determined by its 3D path through space (together with its departure time) and the costs captured by the function $C(v)$ are given by the (aircraft-dependent) constant $C(v_c)/v_c$ multiplied by the distance flown relative to the air.

(4.3.25) For simplicity, we think of the cost as being given by the constant $C(v_c)$ multiplied by the time taken (flying at cruise speed v_c) instead. However, it is important to remember that the cost does not decrease (as easily) with a speed increase like the travel time does, as the cost is actually determined by the distance flown through the air.

(4.3.26) From equation (14), we see that the time required to fly along a line segment $\mathbf{dx}$ is

$$dt = \frac{|\mathbf{dx}|}{v_g} = \frac{dx}{\sqrt{v_c^2 - w^2 \sin^2\theta} + w\cos\theta}. \tag{22}$$

(4.3.27) The time it takes to fly a path $\mathbf{x}(s)$, depending on an arbitrary parameter $s \in [s_0, s_1]$, through space is then given by

$$T = \int_{s_0}^{s_1} \frac{|\mathbf{x}'|\,\mathrm{d}s}{\sqrt{v_c^2 - |\mathbf{w}|^2 + (\mathbf{w}\cdot\hat{\mathbf{x}}')^2} + \mathbf{w}\cdot\hat{\mathbf{x}}'}, \tag{23}$$

where $\mathbf{w}$ may depend on $\mathbf{x}(s)$ and $\hat{\mathbf{x}}' = \mathbf{x}'/|\mathbf{x}'|$ is the unit tangential vector.

(4.3.28) If we again assume that the wind speed is small, i.e. $w \ll v_c$, then an approximate result is

$$T = \int_{s_0}^{s_1} \frac{|\mathbf{x}'|}{v_c} - \frac{\mathbf{w}\cdot\mathbf{x}'}{v_c^2} + O\left(\frac{w^2|\mathbf{x}'|}{v_c^3}\right)\,\mathrm{d}s. \tag{24}$$

(4.3.29) Here we recognise the first term as yielding the total arc length L of the curve (divided by v_c).

(4.3.30) In the particular case that the wind field is constant and uniform (and the curvature of the Earth ignored), the second term in equation (24) yields a constant contribution $\mathbf{w}\cdot(\mathbf{x}(s_1)-\mathbf{x}(s_0))/v_c^2$. Thus the integral is minimized when the arc-length is minimised, i.e. by a straight line and the time taken is

$$T = \frac{|\mathbf{\Delta x}|}{v_c} - \frac{\mathbf{w}\cdot\mathbf{\Delta x}}{v_c^2}, \qquad \text{where} \quad \mathbf{\Delta x} = \mathbf{x}(s_1) - \mathbf{x}(s_0). \tag{25}$$

(4.3.31) For general wind fields minimization of the integrals in equations (23) and (24) requires making use of the Euler–Lagrange equations, as discussed in section 4.

5 Queue optimisation algorithm for a network of flights

5.1 Current procedure

(5.1.32) The pilot of a civil aircraft sends a (4D) route proposition to Eurocontrol a minimum three hours ahead of the flight. This route is either accepted or Eurocontrol proposes other one. The pilot can either accept it or repeat the procedure.

5.2 Example

(5.2.33) Eurocontrol assigns routes to aircraft on 'first come first served' basis therefore each pilot will propose an option which is optimal for them, but only at that particular moment.

(5.2.34) Consider the situation where two pilots P_1, P_2 would like to fly through a corridor between two military areas, but only one of them is allowed due to safety reasons. Since P_1 is served first by Eurocontrol, they are offered an optimal route P_{1R1} which runs through this corridor and the other one is left with the more expensive route P_{1R2}. If P_1 chooses P_{1R1}, the pilot P_2 agrees on route P_{2R2} which is less optimal for them than the one they propose P_{2R1}. This rule of assigning aircraft to routes turns out to be non-effective, when the added cost of P_{1R1} and P_{2R2} is strictly bigger than of P_{1R2} and P_{2R1}.

5.3 Problem

(5.3.35) Find an algorithm optimising total cost of routes for the whole network of flight connections, with certain constraints.

5.4 Theoretical model

(5.4.36) Let us denote by $V := \{A_i\}_{i\in I}$ - the set of considered civil airports in Europe and the set of all used air corridors by $\{X_k\}_{k\in K}$ (especially those spaces between two military areas). Consider discretized model of flight connections, i.e. symbolically identify each entrance and exit to/from corridor $\{X_k\}$ with two points $\{x_k\}$, $\{x^k\}$ respectively (assume for example that entrance point lies more to the north or east with respect to exit

point). For every $i \in I$ let $T_i := (t_n^i)_{n \in N_i}$ be the sequence of times during the day, for which there exist flight starting from airport A_i. Define $T := \bigcup T_i$.

(5.4.37) For every $i, j \in I$, if there is a flight connection $A_i \to A_j$, starting at t_n^i, then denote by $x_k(t_n^i)$, $x^k(t_n^i)$, the times when the corresponding aircraft enters and exits corridor X_k. Note that the aircraft can enter the corridor through its exit point, and then exits through enter point. This is only to be consistent with the notation - enter and exit points are fixed points on the map of Europe. So it is possible that $x_k(t_n^i) > x^k(t_n^i)$.

(5.4.38) For each time interval $[s, s+h]_{s,h \in \mathbb{R}_+}$ and corridor X_k, consider the set: $C_{X_k}^{[s,s+h]} := \{(i,j,t) \in I \times I \times T \mid \exists \text{ flight } A_i \to A_j \text{ starting at time } t \in T_i \text{ so that } x_k(t) \text{ or } x^k(t) \in (s, s+h)\}$.

5.5 Solution

(5.5.39) Assume that at time s_0 schedule for the flights starting, at any from considered aircraft, at times between $s_1 < s_2$, $s_1 > s_0$ has to be created. For each of these flights create a list of possible $4D$ paths, that is each of the path is a sequence of points in 4-dimensional space - $((x_k, x_k(t_n^i)), (x^k, x^k(t_n^i)))_k$, for only this k that corridor X_k is visited. Moreover impose that the sequence $(max(x_k(t_n^i), x^k(t_n^i)))_k$ has to be decreasing or increasing.

(5.5.40) Now calculate the costs of each of the routes for every flight and set them in pairs: (sequence of paths for all of the flights with ascending order with respect to departure, summarized cost of all of these routes). Then sort this sequence, in increasing order, of pairs only with respect to the second term in pair, that is summarized cost of these routes.

(5.5.41) Loop over this sorted sequence, each time checking condition $C_{X_k}^{[s,s+h]} <$ critical value of aircraft in one corridor, due to safety reasons, for every $k \in K$ and appropriate times $s, s+h$. Stop the loop when this condition is satisfied for every $k \in K$.

(5.5.42) The wanted schedule is given by the sequence from the last loop. Note that the stopping condition exists, since we can find routes like in the algorithm used nowadays.

(5.5.43) This procedure can be applied since we assume the knowledge of the routes chosen before time s_0.

5.6 Efficient algorithm

(5.6.44) When looking for an efficient algorithm, or implementation of the above, one has to take into account:

- For each airport we can omit certain nodes defining non-relevant corridors, e.g. when flying from Krakow to London, the route over Serbia's territory can be discarded.
- When choosing the relations between s_0, s_1, s_2, we need to consider complexity of the algorithm (number of flights and possible corridor nodes).
- Weather predictions are deterministic up to six hours ahead, so the difference $s_2 - s_0$ cannot be too big comparing to the duration of the flights.

6 "Cost" metrics

(6.0.1) In this section we look at some considerations in terms of the cost of travel between two given airports A and B, assuming there are no obstacles between them. Given a route γ from A to B the cost of the flight through this route is given by

$$C(\gamma) := \int_\gamma F(\text{weight of aircraft, type of aircraft, altitude, zones crossed, speed, etc. }) \, d\gamma. \tag{26}$$

(6.0.2) The problem of finding the best possible route for a given aircraft is equivalent to finding the curve γ connecting A with B that minimizes $C(\gamma)$. The "cost function F" can be very complex and we do not write it here explicitly however, any model should consider the following parameters on which $C(\gamma)$ depends

$$\text{zone cost} := \text{max. weight} \times \text{zone dist} \times \text{zone charge}, \tag{27}$$

where **max weight** is the maximum weight of aircraft, fixed for a given aircraft and **zone dist** is the distance through great circle from entry to exit point in the zone.

(6.0.3) The cost function must also consider the fuel consumption and delay costs.

$$\begin{aligned} &\text{fuel cost} := \text{fps} \times \text{travelled distance} \times \text{number of seats} \\ &\text{delay cost} := €50 \times \text{delay minutes}. \end{aligned} \tag{28}$$

(6.0.4) Here **fps** is an averaged measure of fuel usage per kilometer per seat of a given type of aircraft.

(6.0.5) For a numerical investigation we could begin with a simple model and assume that we can divide the part of the route during which the aircraft stays above some certain altitude, into small segments creating a sequence of parts of the great circles. Moreover since above this critical altitude the wind might be considered as a piece-wise smooth vector field, we could assume that on each of these parts the wind is a constant vector. Hence we can write:

$$\begin{aligned} &\text{travel duration} = \text{duration of ascent phase} + \\ &\text{duration of being above the critical altitude (cruise flight)} + \\ &\text{duration of descent phase}. \end{aligned} \tag{29}$$

(6.0.6) Here the duration of being above the critical altitude can be written as the sum $\sum_{\text{segments}} \frac{\Delta dist}{V_g}$ where $\Delta dist$ is the length of the path segment described above and V_g is the speed of aircraft with respect to the ground. These can be computed using application E6B Flight Computer available at `http://www.vandeenensupport.com/projects-en/rs/index.html`

(6.0.7) This model shows that when comparing costs between different routes we should take into account not only the distance travelled but weather conditions and zones visited as well.

7 Future ideas

(7.0.8) All these investigations could be taken further, adding layers of complexity and refinement to each model. For example one could modify the discrete algorithm to find the path of several flights simultaneously. The model could include sector capacity and scheduling constraints. These last two options require the inclusion of time in the optimisation model. The whole network efficiency optimisation algorithm could also be developed.

8 Program files

(8.0.9) RunFile.m calculates the number of intersections of the paths with airbases. With 33,000 flights, 38 airbases and 48 time periods the program took approximately 6 hours to run.

(8.0.10) PlotMaps.m allows us to plot different types of data as seen in the results. We can plot graphs of the intersection for each base, or colour coded maps which can be turned into movies. This is much quicker than the previous file.