Voronoi diagrams and Simulated Annealing for airspace block optimization

Andréas Guitart[1)†], Julien Lavandier[1)], and Daniel Delahaye[1)]

[1)]*OPTIM, Ecole Nationale de l'Aviation Civile, France*
[†]*email: andreas.guitart@enac.fr*

This paper investigates the design of airspace blocks using Voronoi diagrams and simulated annealing. This approach aims to optimize the layout of airspace blocks by minimizing the complexity gap between them. The algorithm is tested with different complexity metrics. By using Voronoi diagrams, which partition the airspace into regions around specified points, and simulated annealing, which iteratively refines solutions to find near-optimal configurations, the algorithm provides a systematic method for airspace design. The study focuses specifically on the French airspace, providing a real-world application of the proposed methodology. Through experimentation and evaluation, the algorithm demonstrates its ability to generate airspace block configurations that balance complexity. This research contributes to ongoing efforts in airspace management and optimization by providing insights and techniques for designing airspace structures that meet the evolving needs of air traffic control systems.

Key Words : Complexity metrics, airspace block, Simulated Annealing, Voronoï diagram

1. Introduction

As air traffic continues to grow and resources such as air traffic controllers remain limited, there is an strong need to reduce airspace congestion. The existing airspace layout must be adapted to the changing traffic demands to mitigate airspace congestion. Currently, airspace is segmented into 3D volumes called sectors to ensure safe and efficient air traffic management. The Dynamic Airspace Configuration (DAC) concept is a key element in both the current and future European air traffic systems, enabling the increase of airspace capacity. The SESAR Concept of Operations [18] emphasizes DAC as the primary solution for balancing demand and capacity, positioning it as a central pillar in the envisaged architecture of European airspace outlined in the Airspace Architecture Study. Increasing airspace capacity is essential to address the significant capacity constraints experienced in recent years and to accommodate the expected growth in air traffic while maintaining safety standards, improving flight efficiency, and mitigating environmental impacts. The design of basic volumes is at the core of the DAC process. As such, the design of basic volumes is key for defining sectors and sector configuration plans that are both efficient and resilient. Those basic volumes can be classified into the following classes or types: Elementary Sectors (ES), Airspace Blocks (AB), and Shareable Airspace Blocks (SAB). Elementary Sectors (ES) are sectors that are controllable on their own. To form a controllable sector, Airspace Blocks (AB) need to be attached to another AB or ES, while Shareable Airspace Blocks (SAB) are non-workable volumes that need to be dynamically attached to any ES or AB to build an operational sector. This work is part of the European project SMARTS (Smart Sectors) [19], which aims to provide the right amount of capacity at the right time with maximum efficiency to better meet air traffic demand. This paper focuses on the design of AB by comparing different complexity metrics.

This paper is organized as follows: Section 2. presents an overview of related works. In Section 3., a mathematical model of airspace blocks is proposed. In Section 4., the solution algorithm based on the Simulated Annealing and Voronoi diagram is described. Finally, results on French airspace are presented in Section 5..

2. Previous related works

This section presents some previous related works on the Dynamic Airspace Configuration problem and complexity metrics.

2.1. Dynamic Airspace Configuration

To date, only a handful of studies have addressed DAC. In fact, DAC represents a relatively recent paradigm shift within airspace systems. This concept revolves around the allocation of airspace as a resource to meet the evolving needs of airspace users. For a deeper understanding of the DAC concept, interested readers may refer to the works of Kopardekar et al. [13], Zelinski and Lai [21]. The DAC concept should not be confused with the current operational airspace configuration. While such an airspace configuration focuses on adapting the airspace to the changing needs of airspace users by creating new sector configurations for different time periods throughout the day [3, 7, 15], DAC is concerned with allocating airspace as a resource to meet these changing needs. Existing approaches to DAC adopt a model where the airspace is initially partitioned into functional 2D or 3D airspace blocks [6, 12, 21]. This setup transforms the DAC problem into a combinatorial problem. Configurations are assembled using controlled sectors constructed from predefined airspace blocks. However, some studies utilize already established and operationally functional ATC sectors to create configurations [10], or even entire configurations [20], to

devise an opening scheme. The design of airspace blocks is an important part of DAC. Bichot and Durand [2] compare different methods for generating functional airspace blocks. They show that fusion-fission metaheuristics seem to be better at solving this problem. Sergeeva et al. [17] propose to use the Voronoï diagram to build the airspace blocks, guided by a k-means that computes the position of the centers. Schultz et al. [16] also propose to build a Voronoï diagram, but this time guided by fuzzy clustering. As in the two previously presented works, we propose to use a Voronoi diagram, guided by Simulated Annealing in order to minimize the complexity gap between the blocks. In the literature, several complexity metrics have already been defined.

2.2. Complexity metrics

In the context of operational control, we target to find a workload metric that precisely measures the difficulty of air traffic controllers in managing a set of aircraft in a region of interest. The complexity metric only depends on the geometry of trajectories, Γ, in a region of interest and uncertainty of positions, speeds, and future positions. It approximates the difficulty that controllers have in managing aircraft. The complexity, w_p, of the discretized spatiotemporal positions, $p \in \mathbb{R}^4$, along the trajectory, $\gamma \in \Gamma$, is the complexity of the region of interest around this position, p.

Pairwise convergence metric (PWC) The pairwise convergence metric [5] characterizes the variation of the pairwise relative distance, $r_{ij} = \frac{\partial}{\partial t} d_{ij}$, weighted by the spatial distribution of aircraft in a region of interest. Grouped situations with converging aircraft require more attention to manage, as opposed to more evenly distributed or diverging aircraft:

$$w_p = \sum_{\substack{(i,j)\in\{1,\dots,N_A\}^2 \\ i\neq j,\, r_{ij}\leq 0}} -r_{ij} e^{-\alpha d_{ij}^2}. \tag{1}$$

Dynamical system metrics (LDS/NLDS) The dynamical system metrics [4] interpolate the vector field of the trajectories of a region of interest, $\frac{d\vec{p}}{dt} = \vec{f}(\vec{p})$, with the space coordinates, $\vec{p} \in \mathbb{R}^3$, minimizing the interpolation criterion: $\sum_{i=1}^{N} \left\| \vec{v}_i - \vec{f}(\vec{p}_i) \right\|_2^2$, where $\vec{v}_i$ is the speed of the aircraft at position $\vec{p}_i$, and $\vec{f}(\vec{p}_i)$ represents the speed of the vector field at point $\vec{p}_i$ given by the dynamical system. Two possible types of functions have been used in [4]: an affine function, $\vec{f}(\vec{p}) = \mathbf{A}\vec{p} + \vec{B}$, or a non-linear function minimizing a regularity criterion, $\vec{f}(\vec{p}) = \sum_{i\in\{1,\dots,N_A\}} 12\|\vec{p}_i - \vec{p}\|_2 \vec{a}_i + \mathbf{A}\vec{p} + \vec{B}$.

In the case of a linear interpolation, complexity is linked to the stability of the system characterized by the matrix's eigenvalues:

$$w_p = \sum_{\lambda\in Sp(\mathbf{A})\,|\,\mathrm{Re}(\lambda)<0} -\mathrm{Re}(\lambda). \tag{2}$$

In the case of a non-linear interpolation, complexity is linked to the stability of the system of the affine approximation at different grid coordinates, X, characterized by the Jacobian matrix's eigenvalues:

$$w_p = \sum_{\vec{x}\in X} \sum_{\lambda\in Sp(\mathbf{J}_f(\vec{x}))\,|\,\mathrm{Re}(\lambda)<0} -\mathrm{Re}(\lambda), \tag{3}$$

where $\mathbf{J}_f(\vec{x})$ is the Jacobian matrix of the vector field at point x.

Speed disorder (SD) The speed disorder [5] is the speed vectors' disorder and can be expressed as:

$$w_p = \sqrt{\sum_{i\in\{1,\dots,N_A\}} \left\| \vec{v}_i - \overline{\vec{v}} \right\|^2}, \tag{4}$$

where $\overline{\vec{v}}$ is the mean speed vector and equals: $\sum_{i\in\{1,\dots,N_A\}} \vec{v}_i$.

Conflict duration for heading maneuvers (CDH) The conflict duration sensitivity [14] to heading maneuvers is the pairwise conflict duration for heading changes between the minimal, $-\Psi$, and maximal heading change, Ψ:

$$w_p = \int_{-\Psi}^{\Psi} \left| [t_-^{xy}, t_+^{xy}] \cap [t_-^{z}, t_+^{z}] \right| d\psi, \tag{5}$$

where $[t_-^{xy}, t_+^{xy}]$ (resp. $[t_-^{z}, t_+^{z}]$) is the interval of time when aircraft are horizontally (resp. vertically) in conflict.

Conflict duration for heading maneuvers with continuous time uncertainty (CDHU) The conflict duration sensitivity [14] to heading maneuvers with continuous time uncertainty is the total conflict duration for heading maneuvers between $-\Psi$ and Ψ, integrating all possible positions of aircraft i and j around the reference with a constant speed and a continuous time uncertainty:

$$w_p = \int_{-\Psi}^{\Psi} \mathcal{T}_{ij}(\psi)\, d\psi, \tag{6}$$

where $\mathcal{T}_{ij}(\psi)$ is the integral for a specific heading change and equals:

$$\mathcal{T}_{ij}(\psi) = kT(z(\overline{u})) - kT\left(z(\underline{u})\right),$$

where k is the conflict duration's ellipse area divided by π, $z(\underline{u})$ (resp. $z(\overline{u})$) is linked to the minimal (resp. maximal) time uncertainty and:

$$T(z) = \begin{cases} 0 & \text{if } z \leq -1 \\ \frac{2+z^2}{3}\sqrt{1-z^2} + z\frac{\pi}{2} + z\arcsin(z) & \text{if } z \in (-1,1) \\ \pi z & \text{if } z \geq 1 \end{cases}$$

More details about this metric may be found in [14]. All these metrics extract different features of air traffic situations, such as traffic flow and crossing. The described metrics will be used in the dynamic airspace configuration.

3. Mathematical model

This section presents the mathematical model: the space representation, the decision variables, the objective function and the way to build the blocks.

3.1. Grid discretization

In this study, we propose to discretize the space with a grid to define the center's position of the blocks. Moreover, this grid is used to store the complexity values. The grid is composed of n rows and m columns. Each grid cell is defined by a unique number $n_{i,j}$, which is computed as follows:

$$n_{i,j} = i \cdot n + j. \tag{7}$$

After creating the grid, we project all the trajectory 4D points on this grid and assign to each cell (i, j) a complexity $w_{i,j}$ equal to the sum of the complexities of the points contained in the cell. This complexity is therefore calculated as follows:

$$w_{i,j} = \sum_{p \in \chi_{i,j}} w_p, \tag{8}$$

where $\chi_{i,j}$ is the space of the cell (i, j) defined as follows:

$$\chi_{i,j} = \{(x, y) \,|\, x_i \leq x < x_{i+1}, y_i \leq y < y_{i+1}\}, \tag{9}$$

where (x_i, y_i) is the position of the bottom left corner of the cell (i, j). Fig. 1 shows an example with three trajectories crossing a grid. These trajectories are discretized in time. Where the trajectories cross, the complexity will likely be high if they cross at close times.

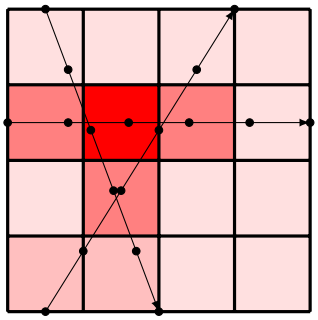

Fig. 1. Example of a complexity grid: the redder the cell, the higher its complexity.

3.2. Decision variables

The first decision variable is the number of centers n^b *i.e.* the number of blocks. It is defined as follows:

$$n^b \in [n^{b,min}, n^{b,max}], \tag{10}$$

where $n^{b,min}$ and $n^{b,max}$ are the minimum and maximum numbers of cells respectively. With this number of blocks, the set of positions can be defined as follows:

$$P = \{p_b = (x_b, y_b) | (x_b, y_b) \in \mathbb{N}^2, b \in \{1, ..., n^b\}\}, \tag{11}$$

where p_b is the center position of the block b. As presented in the previous subsection, the Voronoi cell center positions are some of the centers of the grid cells. The number of possible state n^s is therefore:

$$n^s = \sum_{n^b = n^{b,min}}^{n^{b,max}} \frac{(n \cdot m)!}{n^b!(n \cdot m - n^b)!}. \tag{12}$$

This value can be extremely high if the grid is large. Therefore, we decide to use a metaheuristic: Simulated Annealing.

3.3. Objective function

The optimization process aims to balance the complexity of air traffic between airspace blocks, while limiting the highest complexity among them. Let w_b be the complexity of the block b. This complexity is defined as the sum of the complexity of all grid cells in the block as follows:

$$w_b = \sum_{(i,j) \in \chi_b} w_{i,j}, \tag{13}$$

where $w_{i,j}$ is the complexity of the grid cell (i, j) and χ_b is the space defined by the block b.

The expected balanced complexity is then defined by:

$$W_e = \frac{W}{n_b}, \tag{14}$$

where W is the total complexity of the space computed as follows:

$$W = \sum_{i=0}^{n} \sum_{j=0}^{m} w_{i,j}. \tag{15}$$

The objective function is composed of two different functions. The first one represents the imbalance of complexity between blocks and is defined as follows:

$$f^b = \sum_{b=1}^{n^b} \frac{|w_b - W_e|}{W_e}. \tag{16}$$

The second objective is to minimize the maximum complexity of blocks. This function is defined as follows:

$$f^c = \frac{\max_{b \in \{1,...,n^b\}} w_b \cdot n^{b,min} \cdot n^{b,min}}{W} \tag{17}$$

The constant values W and $n^{b,min}$ have been integrated into this function to have the same unit and order of magnitude as the imbalance function at the beginning of the optimization process. Finally, the bi-objective function f is defined by:

$$f = \lambda f^b + (1 - \lambda) f^c \tag{18}$$

where $\lambda \in [0, 1]$ is a compromise coefficient between the two criteria.

3.4. Airspace blocks design

To construct the blocks, we use the Voronoi diagram. A Voronoi diagram [1] is a way of dividing a plane into regions that are close to each of a given set of objects. It is often categorized as a mosaic. In its simplest form, these objects are a finite number of points on the plane called seeds, sites, or generators. Each seed corresponds to a region called a Voronoi cell, which contains all points on the plane closer to that seed than any other. The Voronoi diagram of a set of points is complementary to the Delaunay triangulation [9] of the same set. The process begins by constructing a Voronoi diagram in two dimensions, then expanded into the third dimension (see Fig. 2). This expansion results in a series of airspace units collectively covering the entire designated airspace. While these units maintain a consistent shape across layers, their complexity varies at different altitudes.

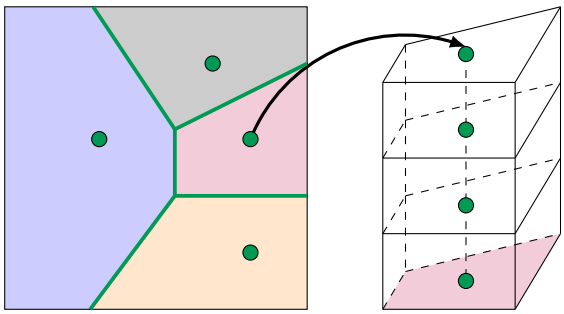

Fig. 2. Construction of the Voronoi diagram and 3-dimensional projection.

4. Resolution algorithm

This section presents the proposed algorithm based on Voronoi diagram and Simulated Annealing.

4.1. Simulated Annealing

This work proposes the use of the Simulated Annealing (SA) algorithm, originally introduced by Kirkpatrick *et al.* [11], inspired by the physical annealing process of materials. The SA algorithm consists of two main phases: a heating phase that brings the solid into a high temperature, followed by a gradual cooling phase to achieve a solid state with minimal energy [8].

A notable feature of Simulated Annealing is its ability to accommodate transitions that may degrade the objective function. Initially, the algorithm operates at a high temperature T, which allows it to accept transitions with significant degradation in the criterion, thus allowing a thorough exploration of the state space. As the temperature T decreases, only transitions that improve the criterion or those with minimal degradation are accepted. Finally, as T approaches zero, no deterioration of the criterion is tolerated, and the SA algorithm behaves much like a Monte Carlo algorithm.

4.2. Solution generation

The proposed algorithm generates a solution by selecting the cell centers.

At the beginning of the algorithm, a list of all grid positions (all $n_{i,j}$) is generated. This list is then shuffled to get a random order of positions. Finally, a number n^b corresponding to the selected centers is chosen (See Fig. 3). The order of the positions and the number n^b are then modified during optimization.

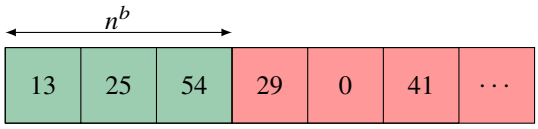

Fig. 3. Example of solution: the centers 13, 25, and 54 are selected to build the Voronoi diagram.

Finally, the Voronoi diagram is created from these activated points (see Fig. 4). The example in this figure will be the common thread of the method explanation. This solution is evaluated in the optimization process. This solution is then modified by generating a neighbor to try to find a better solution.

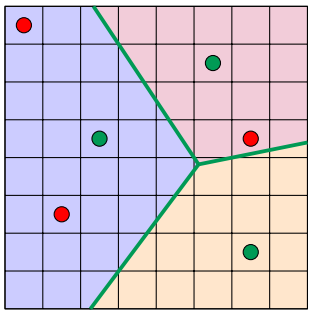

Fig. 4. Example of a Voronoï diagram built from 3 activated points in green. The next three points in the list are in red.

4.3. Neighborhood

A neighborhood operator is necessary to explore the state space efficiently. The algorithm generates a neighbor by modifying the decisions by one of the following operators:

- The first one consists in modifying the number of selected centers. This number can be decreased or increased by 1 at each iteration. In the example of Fig. 5, the number is increased by 1, and point 29 is now considered as a selected center. This change implies a modification of the Voronoi diagram (see Fig. 6).

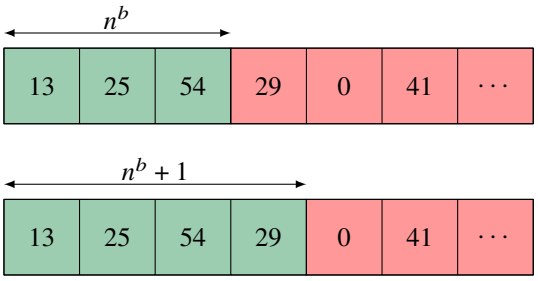

Fig. 5. Change the number of considered centers.

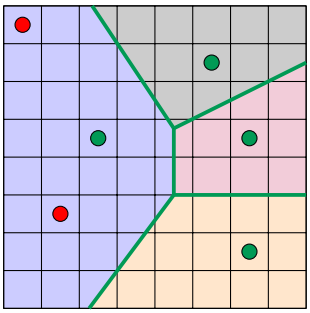

Fig. 6. New Voronoi diagram after modifying the point p_2.

- The second one consists of exchanging two centers in the decision list (see Fig. 7). First, an index lower than n^b is generated, and the corresponding point is exchanged with another one in the second part of the list *i.e.* one of the points that are not selected. This cell can be selected randomly (see Fig. 8) or it can be a neighbor of the considered cell (see Fig. 9). The random selection is mainly done at the beginning of the optimization process to visit as much of the space as possible. On the other hand, the neighbor selection is mainly done at the end of the optimization to adjust the solution.

The second neighborhood operator is performed more often than the first one, since the diversity of solutions is mainly due to the choice of centers.

5. Results

This section presents the results of the proposed method tested on the French airspace and with several complexity metrics.

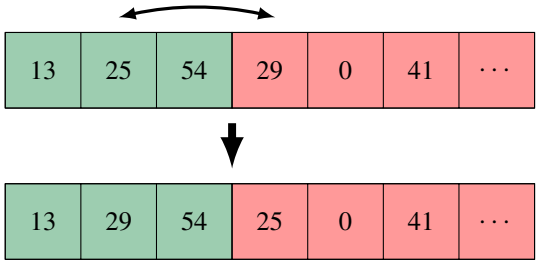

Fig. 7. The exchange of two points.

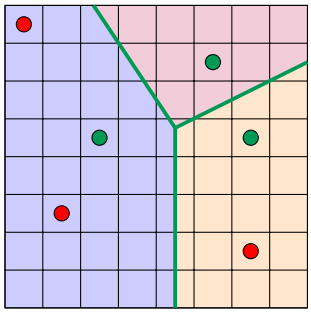

Fig. 8. New Voronoï diagram after an exchange with a random point.

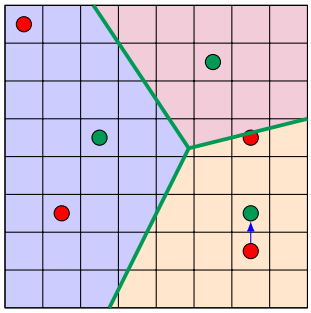

Fig. 9. New Voronoï diagram after an exchange with a neighbor point.

5.1. Case study

The proposed algorithm was tested on French airspace data containing 2,600 generated trajectories (see Fig. 10). The computer used for the experiments is equipped with an Intel Core-i9 processor, with 64 Go RAM. The input parameters of the algorithm are given in Table 1. With these parameters, the number of iterations is $46 \times 10^6 = \left(500 \times \frac{\log 10^{-4}}{\log 0.999}\right)$.

$n^{b,min}$	75
$n^{b,max}$	125
Grid size	200×200
Transition number	500
Temperature decay coefficient	0.999
Final Temperature	$10^{-4}\times$ Initial Temperature

Table 1. Algorithm input parameters.

5.2. Comparison of complexity metrics

All the results with $\lambda = 0.4$ for the different metrics are shown in Figs. 12 to 17 . In all figures, the green polygon corresponds to the French airspace boundary. A red gradient represents the associated complexity. For all metrics,

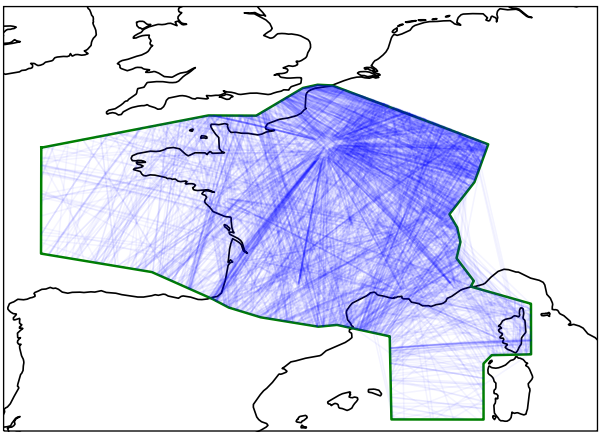

Fig. 10. French airspace trajectories.

the computation time is around 2 hours. Table 2 shows the values of the complexity imbalance, maximum complexity function and the number of airspace blocks after the optimization process for each of the complexity metrics. This table shows that the evolution of the various functions is similar regardless of the metric used. However, it seems easier to find a balance with a rather low maximum with the LDS and NLDS metrics. This table shows that the higher the weight on the complexity balance, the lower the number of blocks generated. Conversely, the greater the weight on maximum complexity, the greater the number of blocks generated. Fig. 11 shows the complexity imbalance and the maximum complexity function values for different λ values. It seems that a λ value between 0.2 and 0.4 is a good compromise between complexity balance and maximum complexity. However, this value can be chosen by the user according to his preferences.

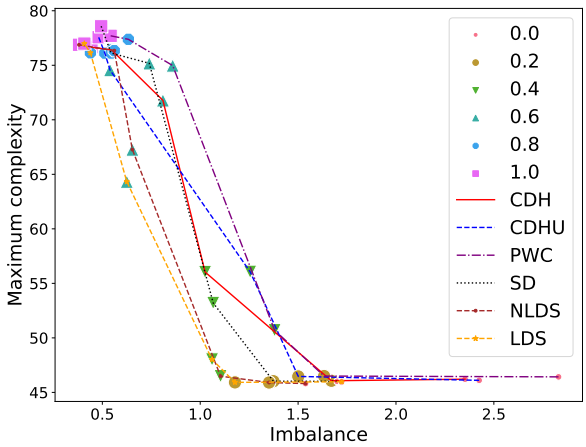

Fig. 11. Maximum complexity and complexity imbalance values depending on λ value.

Fig. 12 shows the result obtained with the pairwise convergence metric (PWC). As expected, it shows that this metric focuses on convergence points. This implies the creation of

Metric	PWC			LDS			NLDS			SD			CDH			CDHU		
Values / λ	f^b	f^c	n^b	f^b	f^c	n^b	f^b	f^c	n^b	f^b	f^c	n^b	f^b	f^c	n^b	f^b	f^c	n^b
0	2.83	46.4	125	1.72	46	125	1.54	45.8	125	1.70	46	125	2.35	46.2	125	2.43	46.1	125
0.2	1.63	46.5	125	1.18	45.9	125	1.35	45.9	125	1.37	46	125	1.67	46.1	125	1.50	46.5	125
0.4	1.38	50.7	115	1.06	48	120	1.10	46.5	124	1.07	53.2	108	1.03	56	103	1.26	56.1	103
0.6	0.87	75	77	0.62	64.3	89	0.65	67.3	85	0.74	75.2	76	0.81	71.8	80	0.54	74.6	77
0.8	0.63	77.4	75	0.44	76.1	75	0.56	76.3	75	0.54	76.1	75	0.55	76.4	76	0.51	76.1	76
1	0.54	77.7	75	0.41	77	75	0.38	76.9	75	0.49	78.6	75	0.44	76.7	75	0.48	77.6	75

Table 2. The values of the complexity imbalance, maximum complexity function and the number of airspace blocks after the optimization process for each of the complexity metrics.

small blocks around Paris, which is a place of strong convergence. In contrast, LDS and NLDS metrics (see Figs. 13 and 14) are more nuanced, as they also seem to detect flows (*e.g.* Toulouse-Paris or Madrid-Paris). For these two metrics, complexity is fairly uniform throughout French airspace. This explains why the imbalance value is lower than that of other complexities. In fact, with an "almost" uniform complexity in space, it is easier for the algorithm to find a balanced decomposition. This implies the creation of blocks of similar size. Speed disorder and CDH metrics (see Figs. 15 and 16 seem to focus more on flows. In fact, complexity peaks in the west and southeast of France. On the other hand, near Paris or Reims, the complexity is a bit lower (especially for CDH), resulting in the creation of larger blocks. This does not seem adapted for areas with very heavy traffic and different flow directions. Finally, the CDHU metric appears to detect both high traffic flows (Madrid-Paris and Madrid-London mainly) and areas of strong convergence with significant differences in complexity from other areas (see Fig. 17). This implies the creation of truly diverse block sizes. This metric seems most appropriate for the next step: sector design. However, due to the very high complexity values, the resulting decomposition is less balanced than the other metrics. These conclusions remain to be confirmed during future work.

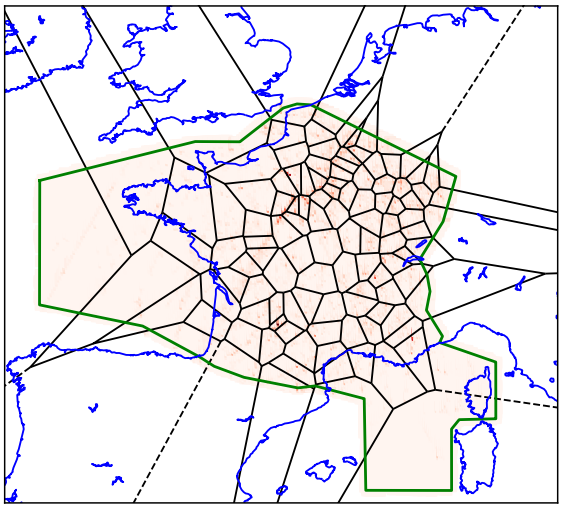

Fig. 12. Results with the pairwise convergence metric (PWC) and $\lambda = 0.4$.

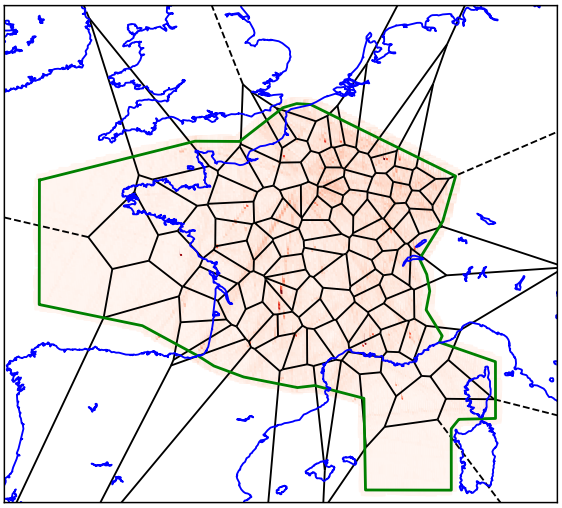

Fig. 13. Results with the linear dynamical system metric (LDS) and $\lambda = 0.4$.

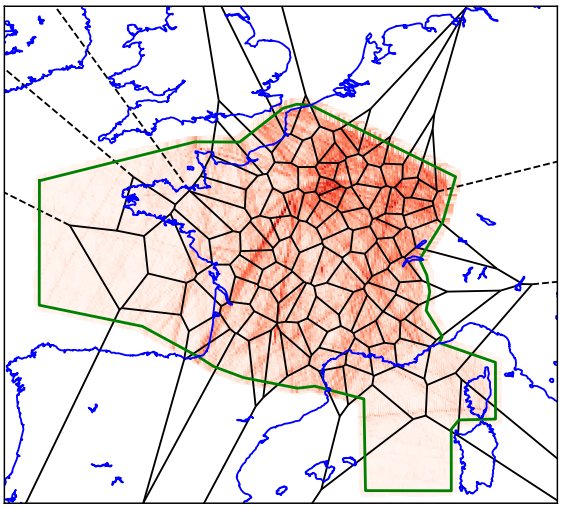

Fig. 14. Results with the non-linear dynamical system metric (NLDS) and $\lambda = 0.4$.

6. Conclusion

This paper deals with the automatic design of airspace blocks. The proposed solution algorithm is based on the Voronoi diagram and Simulated Annealing. The aim was to limit the complexity gap between the airspace blocks. Several complexity metrics have been compared to show their impact on the design. The algorithm has been tested on the

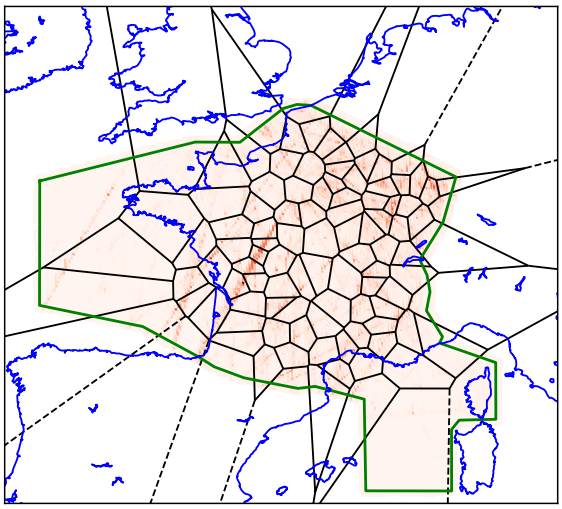

Fig. 15. Results with the speed disorder metric (SD) and $\lambda = 0.4$.

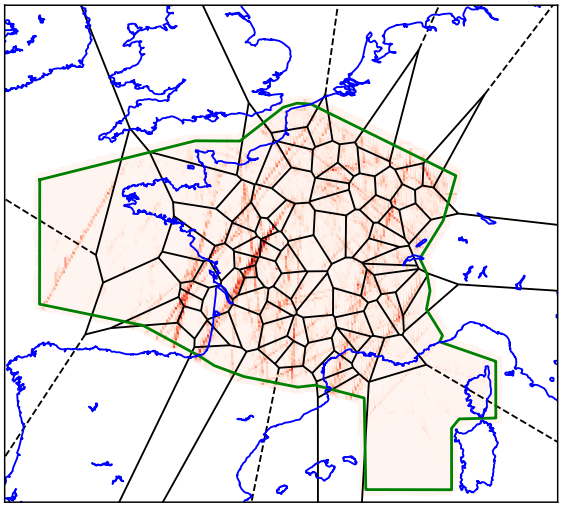

Fig. 16. Results with the CDH metric and $\lambda = 0.4$.

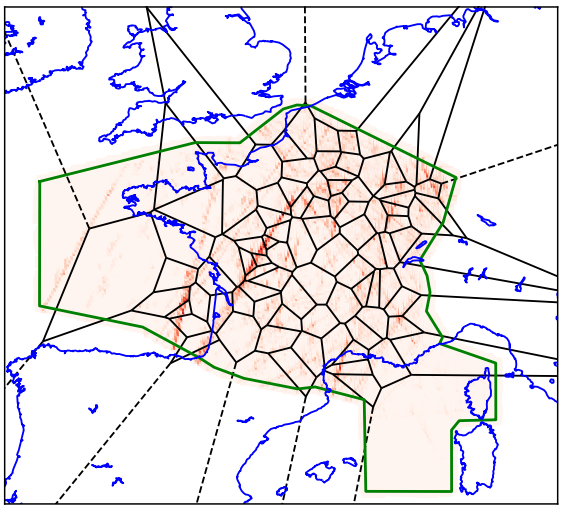

Fig. 17. Results with the CDHU metric and $\lambda = 0.4$.

French airspace, and the results have shown that the proposed algorithm seems efficient in generating complexity-balanced blocks. A comparison of the metrics showed that the CDHU metric seems to be the most effective as it identifies strong flows as well as areas of convergence. This remains to be confirmed in future works using generated airspace blocks. In the future, it could be interesting to combine the metrics studied in this article in the objective function. Future projects will include the integration of the Shareable Airspace Block concept into the optimization process. This study paves the way for the development of a complete tool for dynamic airspace configuration.

Acknowledgment

This project has received funding from the SESAR 3 Joint Undertaking (JU) under grant agreement No 101114686. The JU receives support from the European Union's Horizon Europe research and innovation programme and the SESAR 3 JU members other than the Union. UK participants in SMARTS receive funding from UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant numbers 10086651 (Lancaster University) and 10091277 (NATS). Opinions expressed in this work reflect the authors views only, and the SESAR 3 JU and UKRI are not responsible for any use that may be made of the information contained herein.

References

1) F. Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.

2) C.-E. Bichot and N. Durand. A tool to design functional airspace blocks. In *ATM 2007, 7th USA/Europe Air Traffic Management Research and Development Seminar*, pages pp–xxxx, 2007.

3) Y. Chen, H. Bi, D. Zhang, and Z. Song. Dynamic airspace sectorization via improved genetic algorithm. *Journal of Modern Transportation*, 21:117–124, 2013.

4) D. Delahaye and S. Puechmorel. Air traffic complexity based on dynamical systems. In *49th IEEE Conference on Decision and Control (CDC)*, pages 2069–2074. IEEE. ISBN 978-1-4244-7745-6. doi: 10.1109/CDC.2010.5718004.

5) D. Delahaye and S. Puechmorel. Air traffic complexity : Towards intrinsic metrics. 2000. URL `https://api.semanticscholar.org/CorpusID:53954624`.

6) D. Delahaye, J.-M. Alliot, M. Schoenauer, and J.-L. Farges. Genetic Algorithms for Automatic Regrouping of Air Traffic Control Sectors. In *Evolutionary Programming IV: Proceedings of the Fourth Annual Conference on Evolutionary Programming*. The

MIT Press, 08 1995. ISBN 9780262290920. doi: 10.7551/mitpress/2887.003.0057.

7) D. Delahaye, M. Schoenauer, and J.-M. Alliot. Airspace sectoring by evolutionary computation. In *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360)*, pages 218–223. IEEE, 1998.

8) D. Delahaye, S. Chaimatanan, and M. Mongeau. Simulated annealing: From basics to applications. In *Handbook of metaheuristics*, pages 1–35. Springer, 2019. doi: 10.1007/978-3-319-91086-4_1.

9) B. Delaunay et al. Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7(793-800):1–2, 1934.

10) D. Gianazza. Forecasting workload and airspace configuration with neural networks and tree search methods. *Artificial intelligence*, 174(7-8):530–549, 2010.

11) S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598): 671–680, 1983. doi: 10.1126/science.220.4598.671.

12) A. Klein, P. Lucic, M. D. Rodgers, K. Leiden, and C. Brinton. Exploring tactical interaction between dynamic airspace configuration and traffic flow management (dac-tfm). In *2012 IEEE/AIAA 31st Digital Avionics Systems Conference (DASC)*, pages 4A1–1. IEEE, 2012.

13) P. Kopardekar, K. Bilimoria, and B. Sridhar. Initial concepts for dynamic airspace configuration. In *7th AIAA ATIO Conf, 2nd CEIAT Int'l Conf on Innov and Integr in Aero Sciences, 17th LTA Systems Tech Conf; followed by 2nd TEOS Forum*, page 7763, 2007.

14) J. Lavandier, M. Mongeau, and D. Delahaye. Conflict duration for heading maneuvers with time uncertainty: a complexity metric for air traffic planning. *Preprint*, 2024 ?

15) S. Martinez, G. Chatterji, D. Sun, and A. Bayen. A weighted-graph approach for dynamic airspace configuration. In *Aiaa guidance, navigation and control conference and exhibit*, page 6448, 2007.

16) M. Schultz, I. Gerdes, T. Standfuß, and A. Temme. Future airspace design by dynamic sectorization. In *Air Traffic Management and Systems III: Selected Papers of the 5th ENRI International Workshop on ATM/CNS (EIWAC2017) 5*, pages 19–34. Springer, 2019.

17) M. Sergeeva, D. Delahaye, and C. Mancel. 3d airspace sector design by genetic algorithm. In *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pages 499–506, 2015. doi: 10.1109/MTITS.2015.7223300.

18) SESAR. Sesar concept of operations (conops 2019). URL `https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5c91e877d&appId=PPGMS`.

19) SESAR. Smarts- smart sectors, 2023. URL `https://sesar.eu/projects/SMARTS`.

20) C. Verlhac and S. Manchon. Improved configuration optimiser. *Continuity*, 3:318, 2005.

21) S. Zelinski and C. F. Lai. Comparing methods for dynamic airspace configuration. In *2011 IEEE/AIAA 30th Digital Avionics Systems Conference*, pages 3A1–1. IEEE, 2011.