APPLICATION OF CLUSTERING TECHNIQUES IN OPTIMISATION-BASED FLIGHT CONTROL SYSTEM CLEARANCE

Patrick Piprek[1], Pedro Miguel Dias[2], Jacopo Tonti[1] & David Schwalb[1]

[1]Airbus Defence and Space GmbH, Flight Dynamics, Rechliner Straße, 85077 Manching
[2]Airbus Defence and Space GmbH, Airline Sciences Live Streams, Willy-Messerschmitt-Straße 1, 82024 Taufkirchen

Abstract

Before an aircraft is released to service, flight mechanics clearance assessments over the entire flight regime including potential failure cases, pilot inputs, measurement errors, configurations and further scenarios, such as bird strikes, are conducted to ensure its safe operation. These assessments need to be conducted in a manner that all operationally relevant scenarios are considered and no critical ones are overlooked. For this purpose, global and local optimisation strategies have proven viable as they allow to find a multitude of problematic cases and areas within the vast input parameter space. One drawback of such approaches is that the optimisation algorithm tends to converge to an optimum by design. While this is an important result for a clearance, this tendency to go towards the optimal regions may cover up further critical areas that are also operationally relevant. To mitigate this issue, a connection of optimisation methods and clustering techniques is proposed in this study. The goal of this approach is to use clustering to avoid already identified critical parameter combinations in following algorithm evaluations and rather explore other scenarios, which may also be operationally relevant. This approach increases the confidence in finding all critical regions and thus, improves the safety of the aircraft. The proposed framework is shown in an aircraft clearance scenario to highlight its benefits.

Keywords: envelope clustering, optimisation, clearance, genetic algorithm, flight control system

1. Introduction

The assessment of a Flight Control System (FCS) in flight mechanics clearance assessments is one of the most important tasks before a new or upgraded FCS software can be deployed on the hardware for further testing and integration on the Aircraft (A/C) [1]. For these clearance assessments, normal as well as failure case operation is to be considered before an A/C can be deemed safe for operation with this FCS standard. Particularly for fighter A/C, which are often naturally unstable and only artificially stabilised by the FCS, these flight mechanics assessments require significant computational and analysis resources due to the huge amount of flight conditions, manoeuvres, store configurations, and failure cases that have to be considered.

To tackle these issues, studies [2, 3] introduced an optimisation-based flight mechanics clearance philosophy: The approach is based on different types of multi-objective genetic algorithms that were combined in an island model approach. By this, a parallelization using the Open Message Passing Interface (MPI) [4] is facilitated and consequently deployment on a High Performance Computing (HPC) cluster is supported. With this setup, it is possible to evaluate millions of scenarios in a fast manner, consequently improving the confidence in the results, particularly in terms of finding the worst cases. Thus, the safety of the A/C and the FCS is improved.

One drawback of purely optimisation-based techniques is however that they tend to find regions of interest, i.e. with critical behaviour based on the selected criteria, in the initial steps and then continue to converge in those regions to find the optimum. While this is an important, desired feature, it also has drawbacks in the context of clearance assessments because it is not only desired to find the worst

case but, at best, find all critical regions with all critical scenario topologies [1]. This is necessary to ensure that the A/C is safe within the whole operational range, which is most pronounced by considering the assessed flight envelope conditions.
For this purpose, study [3] already introduced a methodology to slice the envelope in different parts to foster exploration at early stages of the optimisation algorithm execution. This method proved viable in the context of exploring more than just the most critical regions. Still, it has drawbacks in terms of properly setting up the envelope splitting, e.g. the number of splits, and specifically when and how to switch back to a common envelope again.
To mitigate these issues of purely optimisation-based approaches, the combination of clustering and optimisation has long been a research subject: In study [5], clustering techniques were used to avoid finding the same local minimum multiple times in local searches of the global optimisation algorithm, with the clusters being used in stopping criteria. This procedure also directly increases the chance of finding the global optimum as more effort can be concentrated in exploration. Studies [6, 7] then proposed different clustering methodologies in the context of stochastic optimisation. The authors defined different methods that use a minimal number of local searches in clusters and a multi-level approach to find all local optima with high reliability.
The proposed methods in [6,7] were revisited in [8]: The authors used global optimisation techniques to create the overall results and applied clustering to decide on whether or not to start a local search. They also considered dimensionality reduction strategies to mitigate computational issues in high dimensional space improving the applicability of the algorithms in practice.
Further connections of clustering and optimisation were done in [9, 10], where both transfer learning [9] and graph-based clustering [10] were used to enhance a global optimisation algorithm. It was specifically shown that the time to convergence and number of samples could be significantly reduced by applying clustering techniques.
One common denominator of all previously mentioned studies is that they use clustering to improve the probability of converging to and finding all local optima and specifically the global optimum. While this is valuable in clearance assessments, it is not the most important result as it is rather desired to find “all” critical points and specifically the associated critical regions. An approach in this direction was published in [11]: The study used a bisection approach and truncated Uniform distributions for uncertainties to assist in global exploration by means of finding critical regions and then applying local gradient-based optimisation techniques to find the worst cases.
Still, the combination of clustering and optimisation to explore more than just critical regions has not yet been studied deeply. This study addresses this topic by using clustering techniques on the optimisation results directly within the generations. These techniques are used to carve out e.g. the already identified critical envelope areas by means of finding similar patterns that lead to criteria violations. With this information, further, different scenario combinations can be explored to find other potentially critical cases, while the carved out areas can either be optimized in a separate manner or be analysed stochastically. The calculated clusters are, however, penalised such that the optimiser does not evolve inside them. Initially, this approach is applied in the context of envelope clustering.
Overall, the paper details the implementation and connection of clustering to and within optimisation algorithms in Section 2. Specifically, the handling of the clustering results by means of constraints is explained. Additionally, the definition of a stationary condition to trigger the clustering algorithm is detailed. Furthermore, a representative example for an A/C clearance assessment in Section 3 is used to show the benefits of the proposed algorithm in a large-scale application. An outlook and conclusive remarks are given in Section 4.

2. Theory

This section summarises the theory of the used algorithms connected within the proposed FCS flight mechanics clearance framework. It specifically introduces the considered optimisation problem formulation, details the introduction of clustering results as constraints, and introduces the algorithmic procedure of the optimisation with clustering.

2.1. Optimisation Problem

In general, the following multi-objective minimisation problem, including constraints, is solved in an FCS clearance assessment [2]:

$$\begin{aligned}
\underset{\mathbf{z}}{\text{minimise}} \quad & \mathbf{J}(\mathbf{x};\mathbf{z}) , && \mathbf{J} : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^o \\
\text{subject to} \quad & \mathbf{z}_{\text{lb}} \leq \mathbf{z} \leq \mathbf{z}_{\text{ub}} , && \mathbf{z} \in \mathbb{R}^n , \\
& \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x};\mathbf{z}) , && \mathbf{f} : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^m , \\
& \mathbf{c}_{\text{lb}} \leq \mathbf{c}(\mathbf{x};\mathbf{z}) \leq \mathbf{c}_{\text{ub}} , && \mathbf{c} : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^p , \\
& \psi_{\text{eb}} = \psi(\mathbf{x};\mathbf{z}) , && \psi : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^q .
\end{aligned} \tag{1}$$

In (1), $\mathbf{z}$ is the vector of optimisation parameters, while the objective functions, equality, and inequality constraints are given by $\mathbf{J}(\cdot)$, $\psi(\cdot)$, and $\mathbf{c}(\cdot)$, respectively. The state dynamics are defined by the functions $\mathbf{f}(\cdot)$ for the states $\mathbf{x} \in \mathbb{R}^m$, which are solved by a single shooting discretisation [2, 12]. The indices "lb", "ub", and "eb" denote the lower, upper, and equality bound vectors, respectively.

The main practical caveat of the problem specification in (1) is the multi-objective definition. Such problems are generally difficult to solve, because they require Pareto optimality [13], which may be difficult to achieve. However, they are of high relevance in practical FCS clearance assessments because there is not only one objective, but multiple ones, based on the different requirements [1]. For the solution of such problems, specifically considering the large amount of available parameter combinations, studies [2, 3] introduced a framework, which is used in this study and discussed in Section 2.2.

2.2. Island Model and Genetic Algorithm

To solve the problem specified in (1), genetic algorithms are proposed in [2], which are also used in this study. The following paragraphs will thus only give a general overview on some details of the algorithms required to understand the behaviour of the optimisation algorithm with clustering.

The basic algorithmic workflow of a genetic algorithm (e.g. used in NSGA2 [14], NSGA3 [15], or omni-optimisation [16]) is visualised in Figure 1: The idea is to define an initial population of the optimisation parameters, based on e.g. random sampling, and then evaluate the dynamics and constraints to get the initial results. Then, the non-dominated and crowding distance sorting step in Figure 1 is executed. Here, the individuals are sorted into different fronts and chosen for the next generation from best to worst performing. Afterward, the genetic algorithm operators of "selection", "crossover", and "mutation" are applied to generate a new offspring population ("new generation") that should improve on the current solution. This procedure is iterated until a user-defined stopping criterion is met. In general, the balance between exploitation and exploration of the search-space is essential to the performance of these algorithms. Such balance is achieved mainly via a trade-off between crossover and mutation [14–16].

One important aspect of the proposed genetic algorithms is the capability to identify converged states by means of a stopping criterion. This is necessary to trigger the clustering, as introduced in Section 2.3, because it should only be executed once a sufficient amount of critical cases has been found. The natural choice for a convergence criterion is the maximum number of generations, but in the context of clustering other criteria need to be assessed as the maximum number of generations is only terminal: These can e.g. be the number of solutions in the first Pareto front or the current cost function values. Additionally, the set of critical parameters may be assessed for changes. If those did not change for a certain amount of generations ("backtracking" steps), it is assumed that the algorithm has reached a "converged" state. Then, the clustering algorithm is used to exclude already identified critical regions.

Note that the genetic algorithm used in this study therefore has two different, individual stopping criteria associated to it: One for the actual termination of the optimisation (e.g. the maximum number of generations) and one for triggering the clustering algorithm (e.g. tracking the evolution of the number of solutions in the first Pareto front). This is visualised in Figure 1 by the two decision blocks with the penalty application as introduced in Section 2.3.

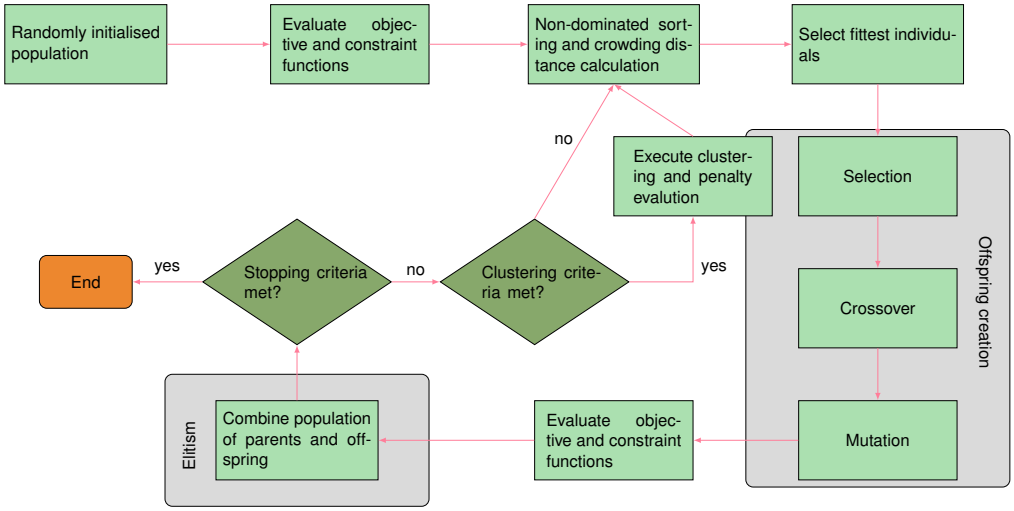

Figure 1 – Flowchart of a generic genetic algorithm with clustering applicable in the islands of the generalised island model (graphic extended from [2]).

With the genetic algorithm and the cluster triggering defined, Figure 2 gives an overview on the island model that was introduced in [2] and is used in this study together with the envelope clustering (see Section 2.3). The framework is implemented in Java™, but also provides interfaces to SQLite™ databases and the possibility to execute sub-modules written in other programming and scripting languages such as Fortran, Python™, or Matlab® code.
The algorithm has a two-stage parallelisation scheme in which not only each island is running in parallel its individual algorithm, but also the evaluation of the objective/constraint functions is performed in parallel: The outermost parallelisation is done by Open MPI (cyan box) [4], a message passing interface which handles the communication between the processes on a HPC. The distribution of the processes and allocation of the cores on the HPC nodes is handled by the Simple Linux Utility for Resource Management (Workload Manager) (SLURM) [17]. In general, different genetic algorithms (see Figure 1) may be executed in each individual island, while clustering is also conducted for each island individually. Thus, the original framework proposed in [2] is independent of the application of clustering and remains generically applicable. Furthermore, this consequently means that only dedicated islands may apply clustering constraints and thus, constrained and unconstrained optimisation approaches can be combined.

2.3. Envelope Clustering Constraint

The envelope clusters, currently used as constraints in the optimisation algorithm (see Section 2.1), are the result of a Density-based spatial clustering of applications with noise (DBSCAN) algorithm [18] applied to the envelope coordinates (Mach-altitude) identifying each simulation run with violating criteria [2]. Here, DBSCAN is a density-based, non-parametric clustering algorithm relying on the concept of connectedness to find arbitrarily-shaped clusters. It is one of the most popular clustering algorithms [19], by virtue of its robustness to noise and because it does not require prior knowledge or estimation on the number of expected clusters.
For the evaluation, the dataset of envelope points, i.e. Mach-altitude coordinates, is normalised applying user-defined scaling factors to each coordinate individually. These scaling factors correspond to the connectedness margins, i.e. the maximum distance in Mach or altitude, respectively, that two points may exhibit to still be considered connected. Here, the DBSCAN algorithm paired with a Chebyshev distance metric is applied to the normalised dataset with a neighbourhood radius, $\varepsilon = 1$, and a single-linkage condition.
This approach achieves to group envelope points within the specified margins in either Mach or altitude, by virtue of the fact that the scaling factors correspond to these margins. Here, the connectedness concept at the base of DBSCAN is exploited to expand the connected area as long as neighbours are found. With the connected areas defined, a hulling algorithm is executed to resolve

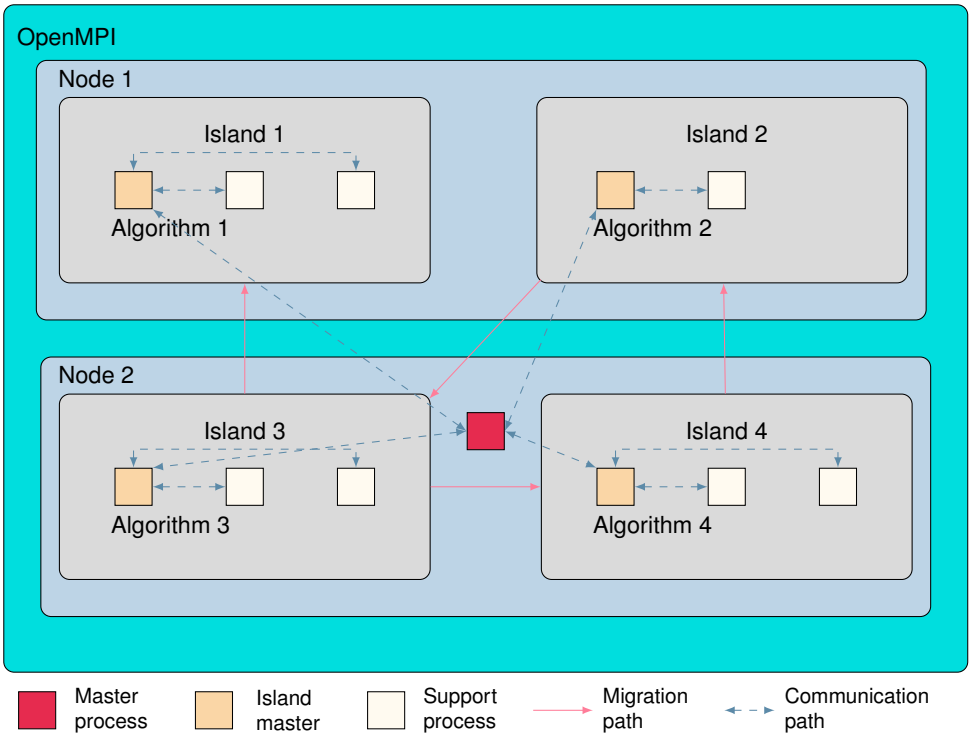

Figure 2 – High-level illustration of the island model implementation including different parallelisation layers (graphic taken from [2]).

the envelope patches that bound the clusters and provide the definition of them by a combination of Mach, altitude, and Displayed Air Speed (DAS).
For the purpose of optimisation, the clustering results are generally described by the convex hull surrounding the critical points in the envelope. The identified clusters are introduced as constraints within the optimisation as specified in [2]. The pseudocode in Algorithm 1 details the definition of a single envelope clustering constraint for the purpose of optimisation.

Algorithm 1 Modelling of single envelope clustering constraint in optimisation.

Require: Data file with envelope points from clustering defining the convex hull; current population.
1: Read the envelope points from the clustering data file.
2: Calculate polygon shape of current convex hull. ▷ Linear interpolation between points.
3: **for** $1 \leq k \leq n_{pop}$ **do** ▷ n_{pop}: Population size.
4: Get current individual k from the population.
5: Get current envelope point (Mach number and altitude) from individual k.
6: **if** Envelope point in current polygon **then**
7: Increment constraint violation value by 1 in individual k.
8: **else**
9: Increment constraint violation value by 0 in individual k.
10: **end if**
11: Add the constraint violation value to each objective of the individual k.
12: Re-assign individual k to population.
13: **end for**
14: **return** Population with information on constraint violation and penalised objectives.

It should be noted that, although the clustering constraint was introduced here for envelope clustering and a single constraint only for the sake of simplicity, the procedure in Algorithm 1 can be extended to other types of clusters and multiple constraints in a straightforward manner.

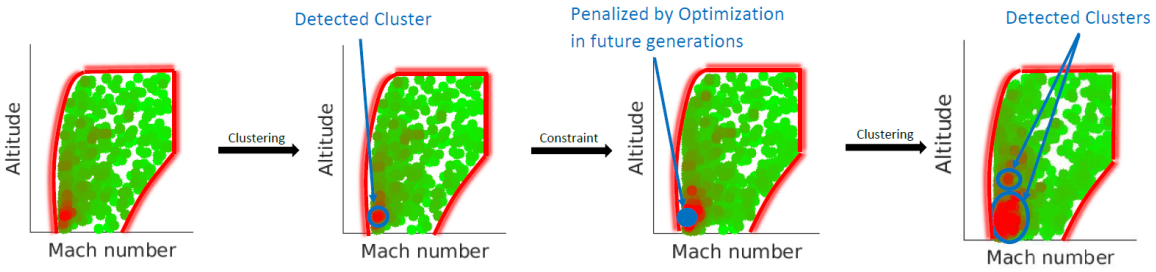

Figure 3 – Basic idea of clustering criteria violations in the envelope (indicated by red points) and exclusion of clusters (indicated by blue areas) by means of penalities during optimisation.

2.4. Connection of Clustering and Optimisation

As stated in Section 2.3, the results of the clustering step are provided to the optimiser as a constraint. In a genetic algorithm, this is achieved by penalising the resulting objectives such that it is no longer attractive for the algorithm to choose individuals in those conditions. Additionally, it may be sensible, but not mandatory, to re-initialize the population affected by the clustering constraint (i.e. the Mach number and altitude in the case of envelope clustering) such that the newly introduced constraints are already fulfilled properly within the first generation after introducing them.

Consequently, the optimiser is therefore forced to explore other parameter combination that may not have been a part of previous exploration. This normal optimisation operation mode, including the introduced constraints, is continued until the next convergence situation is reached. Then, clustering is again applied and the results are once more interpreted by the optimiser as constraints. It is important to note here that clustering is not only applied to the results obtained after enforcing the previous clustering result as a constraint, but for all available data points. This is done due to the fact that the further exploration might have resulted in e.g. connection of clusters or a different assembly of the clustered areas, which is only covered by considering all available results. The previously described algorithmic procedure, which is applied to each island individually (see Figure 2), is visualised in Figure 3. Here, it is shown that envelope clusters identified in the initial generations of the execution are penalised in future ones to be able to also explore different parts of the envelope more efficiently.

The general workflow of the algorithm, related to an application with envelope clustering, is summarized by the pseudocode in Algorithm 2 and detailed in the following: Overall, the optimisation, which is based on e.g. a genetic algorithm (see Section 2.2), is executed and tries to find the worst case response of the A/C with respect to the defined criteria. Once a potentially stationary/converged condition has been reached (see Section 2.2), which may be the case once the number of solutions in the Pareto front are no longer changing or when the worst results obtained by the algorithm are not changing for a specified number of generations, a clustering algorithm is triggered (in this study, envelope clustering as detailed in Section 2.3). This step analyses all results obtained up until this point and clusters the cases violating the specified criteria by a measure of the connectedness of their e.g. envelope coordinates, i.e. Mach number and altitude.

As already stated in Section 2.3, even though Algorithm 2 has been specified for envelope clustering only in this study, an extension to other clustering methodologies is straightforward.

Algorithm 2 Connection of envelope clustering and optimisation algorithm.

Require: Problem definition and formulation; A/C simulation environment; maximum number of clusters and input parameters required for Algorithm 1

Initialise generation counter $k \leftarrow 1$ and maximum number of generations $k_{\max}$.
while $k \leq k_{\max}$ **do**
    Create realisations of optimisation parameters.
    Evaluate the A/C response at those parameter realisations.
    Evaluate the criteria and non-clustering constraints.
    if Clustering constraint exists **then**
        Penalise individuals within clusters. ▷ Algorithm 1
    end if
    Sort violations according to criticality.
    if Genetic algorithm 'stationary' **then** ▷ Section 2.2
        Initialise clustering of envelope points for all available results. ▷ Section 2.3
        Create constraints in problem, which exclude clustered areas.
        Penalise individuals with violation. ▷ Algorithm 1
    end if
    Increase counter: $k \leftarrow k + 1$
end while
return Worst case parameter combinations for A/C response.

3. Application Example

This section shows an application example of the proposed framework from Section 2 for an exemplary A/C flight mechanics clearance task: The task is the assessment of reduced Air Data System (ADS) accuracy within an otherwise failure-free A/C. This A/C configuration should ideally be carefree, i.e. the pilot shall be able to input any (for the flight condition reasonable) command input without destabilizing the A/C [1]. These pilot inputs are modelled using a breakpoint-based manoeuvre generator as introduced in [3] for both pitch and roll stick commands. Further optimisation parameters are the envelope point, the air brake position, the configuration (i.e. external stores), throttle positions, and the mass direction estimation error (i.e. higher or lower estimated mass properties compared to actual values). For the dynamics, a full, non-linear rigid-body simulation of a A/C is used.

Within the framework of Figure 3, three islands and a total of 100 processes are used. This means that each island has 33 processes associated to it as one process is reserved for the "master" (Figure 2). In this example, all islands run using an omni-optimisation algorithm [16] with an initial population size for each island of 1000 and subsequent populations having the size of 500. The number of generations per island is limited to 150 and the migration size is 25 individuals every 25 generations [2]. The backtracking of the number of solutions in the first Pareto front over 25 generations is used as a trigger for the clustering. Here, the number of solutions has to change by at least $50\,\%$ over 25 generations to initiate a clustering. This setup results in a total of $226,500$ non-linear simulations, which completed after roughly 8 hours on an HPC cluster including clustering and time history analysis.

A two-objective optimisation problem is looked at with one objective being the normalized overshoot above the design FCS Angle of Attack (AoA) limit for the corresponding flight condition, J_{AoA}, and the second one being the normalized maximum absolute Angle of Sideslip (AoS), J_{AoS}. Both of these variables are common initial assessment quantities to find critical situations in an FCS clearance [1]. According to [2], a value of 0 indicates an exceedance of the specified normalisation limits, while the objective value is limited to a maximum value of 1 indicating no exceedance.

Flight envelope plots ("altitude over Mach number") for different generations (indicated by the condition with the variable "GEN") of the algorithms are discussed in the following. These plots display the two considered criteria individually for all individuals calculated until the specified generation. The clusters are indicated by their polygon representation of the convex hull in solid blue. It should be taken into account that the clusters are calculated for the combination of both criteria (i.e. if at least one is violating, the envelope point is considered within the clustering algorithm). Still, the clusters

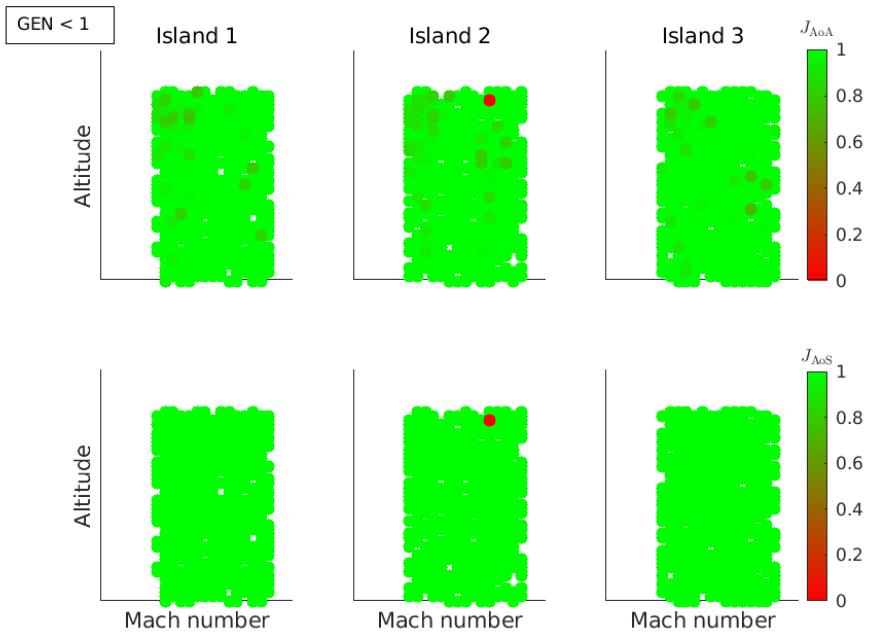

Figure 4 – Initial objective values (actual) for AoA (top row) and AoS (bottom row) objective plotted inside flight envelope for the three islands based on random sampling in zero-th generation.

are depicted in the figures for each criterion individually, which may therefore yield displays where the cluster only covers “green” regions (i.e. non-violating points) for one criterion, because the cluster is driven by the “red” regions (i.e. violating points) of the other criterion. It should also be noted that the terminology “actual objective value” indicates the non-penalised value, while “penalised objective value” describes the one with penalty from the constraint (see Algorithm 1).

First of, Figure 4 shows the flight envelopes for the three islands and two objectives for the initial, i.e. the randomly-sampled, population. Here, the different starting conditions that later on affect the convergence and triggering of the clustering algorithm are displayed. It can specifically be seen that “Island 2” already found a critical case solely based on the random sampling initialisation, which is a desired property of the island model to start with different setups supporting exploration of multiple critical cases.

Figures 5 and 6 display an intermediate generation, which is the first generation where all islands executed the clustering algorithm at least once. It can be seen in Figure 5 that the current cluster shapes (solid blue lines) are different for all islands and “Island 2” seems to evolve best as it found both the lower and higher problematic Mach areas. On the other hand, Figure 6 displays the effect that the clustering constraint had on the islands from the generation it was first introduced (i.e. generation 26 for “Island 1” and “Island 3” and generation 45 for “Island 2”). Both the initial as well as the current cluster shape is displayed. The criteria are containing their penalty according to Algorithm 1 and only the values starting from the first generation with at least one cluster are shown. For “Island 1” and “Island 3”, the clusters did not yet have significant impact on the critical cases and have not evolved over the number of generations they were active. However, it can already be observed, specifically for “Island 3”, that some additional critical cases can be seen just outside the clustered area, e.g. for the AoA criterion, which are seeds for further evolvement towards more critical areas in future generations.

The final objective function values as well as the detected clusters are displayed in Figures 7 and 8: Here, Figure 7 shows that “Island 2” found most of the problematic points and areas in the envelope for both the AoA and AoS objective. It is also imminent that outliers outside the clusters still exist, which may result in further worst cases when one would continue the evaluation. The evolution of the effect of the clustering constraint can be observed in Figure 8, which shows both the initial as well as final cluster shapes. It is depicted that the clusters evolved towards a larger shape and therefore covered more parts of the envelope. While “Island 2” only contains one cluster, the other two islands

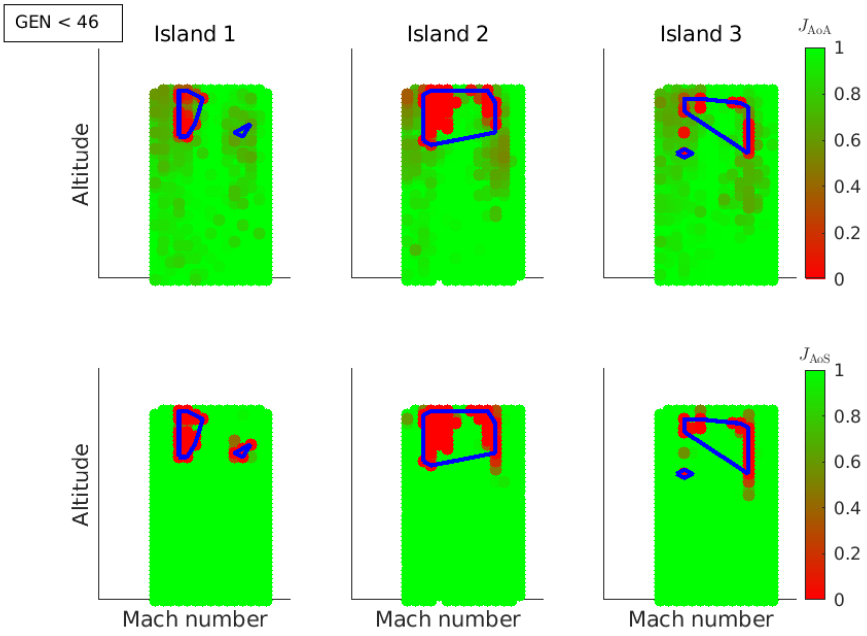

Figure 5 – Objective values (actual) for AoA (top row) and AoS (bottom row) objective including current cluster shape (solid blue) until specified generation plotted inside flight envelope for the three islands based on omni-optimisation algorithm evolution.

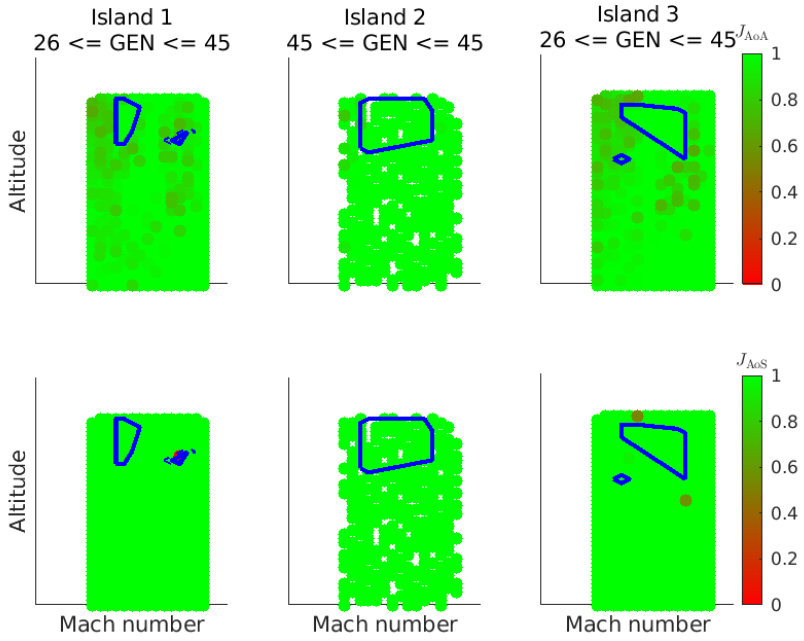

Figure 6 – Objective values (penalised) for AoA (top row) and AoS (bottom row) objective including current cluster shape at generation 45 (solid blue) and initial cluster shape (dashed blue; only visible in right cluster of Island 1 as the clusters in the other parts did not yet evolve) starting from generation where first cluster was applied to individual island (generation 26 for Island 1 and 3 and generation 45 for Island 2) until generation when all islands had a cluster plotted inside flight envelope for the three islands based on omni-optimisation algorithm evolution.

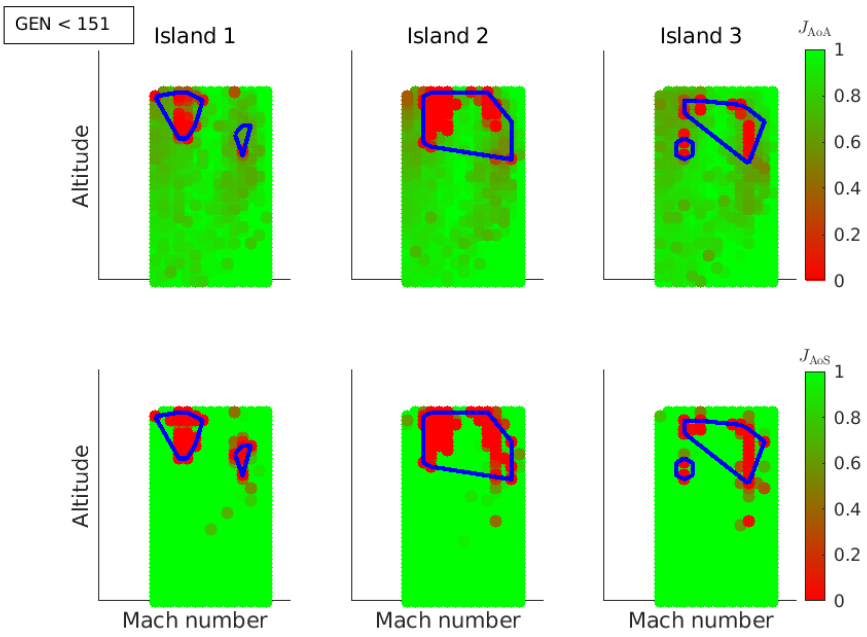

Figure 7 – Final objective values (actual) for AoA (top row) and AoS (bottom row) objective including final cluster shape (solid blue) after reaching maximum number of generations plotted inside flight envelope for the three islands based on omni-optimisation algorithm evolution.

contain two, highlighting the different evolution of the algorithms. It is furthermore visible, specifically for “Island 2” in the AoS criterion, that the critical points found after the initial clusters were introduced started to evolve from the edges of the constraint as desired. Additionally, it can be seen that critical cases even further away from the original critical regions were found (lower altitude domain; compare to Figures 5 and 6 for reference) by the algorithm, which can be attributed to the reduced local convergence tendency due to the clustering constraints in already critical regions.
Summarising, this section has shown that the proposed island model framework with clustering is suitable for the application in highly-complex optimisation tasks, which was shown here for a typical A/C clearance assessment. Specifically, the effect of the clustering to avoid already identified critical regions is depicted in the results, displaying the benefits of the approach.

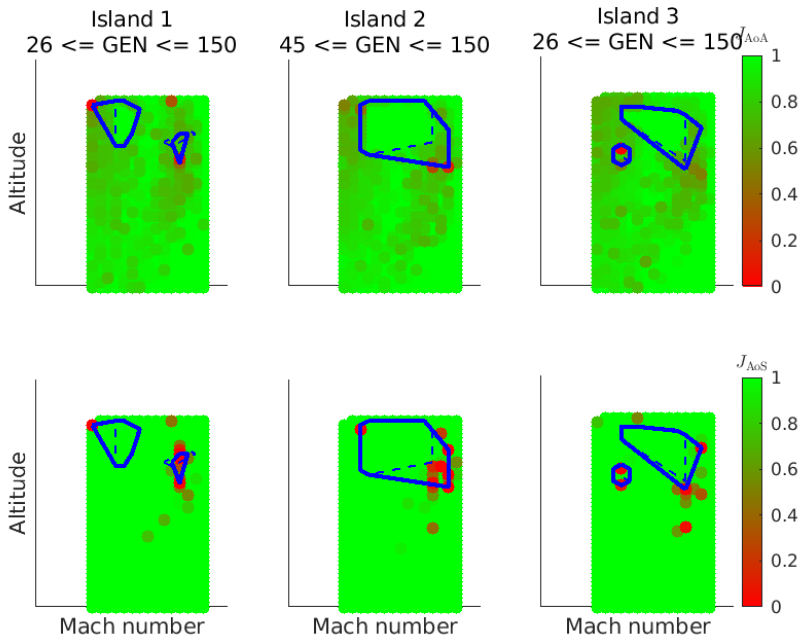

Figure 8 – Final objective values (penalised) for AoA (top row) and AoS (bottom row) objective including final cluster shape at generation 150 (solid blue) and initial cluster shape (dashed blue) starting from generation where first cluster was applied to individual island (generation 26 for Island 1 and 3 and generation 45 for Island 2) until maximum number of generations plotted inside flight envelope for the three islands based on omni-optimisation algorithm evolution.

4. Outlook

This paper presented a framework for clearance of FCSs using multi-objective optimisation algorithms within an island model approach, which is connected to clustering algorithms. The method is tailored to not only explore the critical regions and finding the global and local optima, but rather to find all critical scenario topologies. By this, the proposed scheme specifically thrives in A/C clearances as it is very important to find all operationally relevant critical scenarios and not only a subset of worst cases. The framework gives great flexibility in terms of the problems it can be applied to and how the clusters are calculated.

Future developments may deal with an automatic stochastic analysis of the identified clusters to directly assess their relative criticality. Additionally, further optimisation and clustering algorithms should be explored to find the best possible and suited combinations as well as algorithm parameters for certain application scenarios.

5. Contact Author Email Address

patrick.piprek@airbus.com

6. Copyright Statement

References

[1] R. Stich, *Clearance of Flight Control Laws for Carefree Handling of Advanced Fighter Aircraft*, pp. 421–442. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[2] P. Piprek, P. M. Dias, and D. Schwalb, "A Generalised Multi-objective Island Model for Flight Control System Clearances," in *33rd Congress of the International Council of the Aeronautical Sciences*, (Stockholm), International Council of the Aeronautical Sciences, 09 2022.

[3] P. M. Dias, P. Piprek, and D. Schwalb, "Optimization-based Flight Control System Clearance Philosophy for Fighter Aircraft," in *AIAA SCITECH 2023 Forum*, American Institute of Aeronautics and Astronautics, Jan. 2023.

[4] R. L. Graham, T. S. Woodall, and J. M. Squyres, "Open MPI: A Flexible High Performance MPI," in *Parallel Processing and Applied Mathematics*, pp. 228–239, Springer Berlin Heidelberg, 2006.

[5] A. Törn, "Clustering Methods in Global Optimization," *IFAC Proceedings Volumes*, vol. 19, p. 247–252, May 1986.

[6] A. H. G. Rinnooy Kan and G. T. Timmer, "Stochastic global optimization methods part I: Clustering methods," *Mathematical Programming*, vol. 39, p. 27–56, Sept. 1987.

[7] A. H. G. Rinnooy Kan and G. T. Timmer, "Stochastic global optimization methods part II: Multi level methods," *Mathematical Programming*, vol. 39, p. 57–78, Sept. 1987.

[8] F. Schoen and L. Tigli, "Efficient large scale global optimization through clustering-based population methods," *Computers & Operations Research*, vol. 127, p. 105165, Mar. 2021.

[9] Q. Yang, G.-D. Jiang, and S.-G. He, "Enhancing the Performance of Global Optimization of Platinum Cluster Structures by Transfer Learning in a Deep Neural Network," *Journal of Chemical Theory and Computation*, vol. 19, p. 1922–1930, Mar. 2023.

[10] D. Ikami, T. Yamasaki, and K. Aizawa, "Local and Global Optimization Techniques in Graph-Based Clustering," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, June 2018.

[11] J. Diepolder, P. Piprek, B. Grüter, T. Akman, and F. Holzapfel, *Aircraft Safety Analysis Using Generalized Polynomial Chaos*, p. 67–81. Springer Singapore, 2019.

[12] J. T. Betts, *Practical methods for optimal control and estimation using nonlinear programming*. Advances in design and control, Philadelphia, Pa.: Society for Industrial and Applied Mathematics (SIAM 3600 Market Street Floor 6 Philadelphia PA 19104), 2nd ed. ed., 2010.

[13] J. D. Knowles, *Local-search and hybrid evolutionary algorithms for Pareto optimization.* Doctoral thesis, University of Reading, Reading, 2002.

[14] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2013.

[15] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[16] K. Deb and S. Tiwari, "Omni-optimizer: A procedure for single and multi-objective optimization," in *Evolutionary Multi-Criterion Optimization* (C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, eds.), (Berlin, Heidelberg), pp. 47–61, Springer Berlin Heidelberg, 2005.

[17] A. B. Yoo, M. A. Jette, and M. Grondona, "Slurm: Simple linux utility for resource management," in *Job Scheduling Strategies for Parallel Processing* (D. Feitelson, L. Rudolph, and U. Schwiegelshohn, eds.), (Berlin, Heidelberg), pp. 44–60, Springer Berlin Heidelberg, 2003.

[18] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, p. 226–231, AAAI Press, 1996.

[19] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN," *ACM Trans. Database Syst.*, vol. 42, jul 2017.