EUROPEAN CONSORTIUM FOR MATHEMATICS IN INDUSTRY

ECMI Modelling Week 2018
University of Novi Sad, Faculty of Sciences

OPTIMIZATION OF PLANE ASSEMBLY PROCESS

Churilova Maria (Peter the Great St.Petersburg Polytechnic University)

Cholakov Denis (University of Sofia "St. Kliment Ohridski")
Huynh Ngoc Mai Monica (University of Milan)
Karlsson Tobias (Chalmers University of Technology)
Kopanja Marija (University of Novi Sad)
Stange Simone (Technical University of Darmstadt)

December 7, 2018

Contents

1 Introduction and Background **3**

1.1 Tools 4

2 Problem Formulation **5**

2.1 Optimize Positions 6

2.2 Minimize Number of Fasteners 6

3 Algorithms for Optimizing Positions of Fasteners **7**

3.1 Initial Solution 7

3.2 Improving Solutions - Method 1 8

3.3 Improving Solution - Method 2 9

3.4 Results 11

4 Optimizing Number of Fasteners **14**

4.1 Theory 14

4.2 Practical Approach 14

4.3 Results 15

5 Conclusions **16**

6 Group Work Dynamics **17**

7 Instructor's Assessment **18**

1 Introduction and Background

Over the last few years the demand for the production of airplanes has grown exponentially and, in order to satisfy all the demand, companies need to quicken the production.
Besides that, the assembly process is very complicated, time-consuming and expensive, hence reducing the assembling time is of financial interest and it will save a lot of resources. Indeed, the components of an aircraft are usually made in different countries and transported into one place for assembling (Fig. 1).

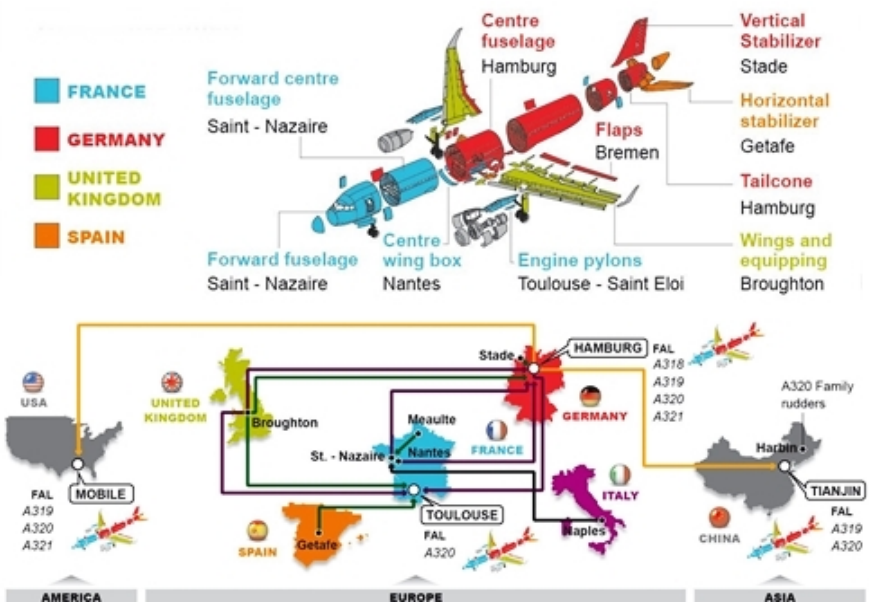

Figure 1: Different parts of the aircraft are made in different countries, as well as assembly processes for each type of airplane.[1]

Even though parts of different airplanes look similar, in reality they all have local irregularities. Indeed, the production chain has its own tolerance and parts can be damaged during transportation and positioning. This means that two different parts of an airplane can never fit together perfectly, but it is necessary to make the gap between them as small as possible or the connection will not be as strong as it needs to be.

In order to minimize the gap between assembled parts, structure technicians use temporary fastening elements (fasteners, Figure 2) before installing permanent ones (rivets). Usually about half of the holes are filled with fasteners, but the fastener installation is a very long process, as it is done manually; so, in order to decrease the assembly time, companies would like to reduce the number of fasteners without losing the quality.

In this project we were working on the optimization of the assembly process for the wing and the fuselage parts. Although there are a large variety of aircraft models, these two parts have similar features for all of them.

[1]Taken from *"Airbus A320 Assembly – A truly international effort."* modernairliners.com. <http://www.modernairliners.com/airbus-a320-introduction/airbus-a320-assembly/>

Figure 2: Example of a fastener.

The report is organized as follows:

- In Section 2 we briefly explain and introduce the given problems. The first problem is to optimize the positions of fasteners in order to minimize the gap between the wing and the fuselage, while the second one is to decrease the number of temporary fasteners which are needed to connect these two parts without loss of quality.
- As regard Sections 3 and 4, we propose algorithms for the solution of the problems above. Furthermore, numerical results will be provided.
- We eventually discuss possible future works and development in Section 5.

1.1 Tools

As a starting point of this project, we were given some information about the background as well as some technical tools. These tools include a solver that calculates the residual gap at each node of the FEM mesh for a given set of fasteners and a given geometry. As our goal was to find the best fastener positions for all geometry variations (initial gaps between parts), we also had been provided with ten different initial gap fields in order to test our program properly. Therefore, we only have to consider the optimization method itself, not the mechanical approach of how to calculate gaps between two structures.

2 Problem Formulation

The main goal of this project is to optimize the plane assembly process, focusing on the junction between a wing and the fuselage. Hence, the two main issues will be both to define properly what we mean by "quality" and to choose a suitable optimization criterion. As it can be imagined, the two connected parts taken in consideration are quite large, but we are working only on their junction area, that looks like a narrow strip (Fig. 3).

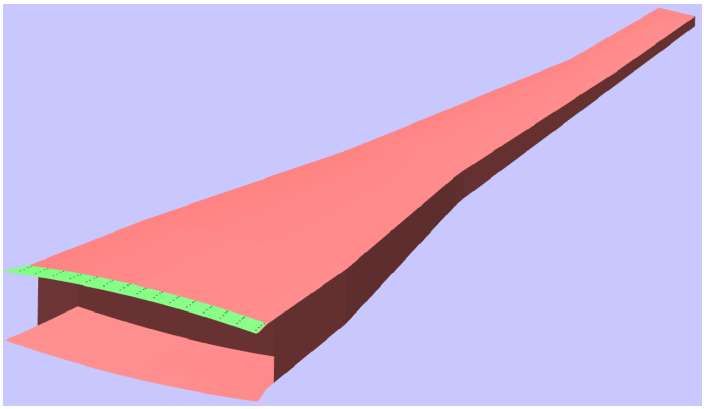

Figure 3: The junction area (green) looks like a narrow strip.

As an example, in the first problem we would like to get the best position for the fasteners, in order to minimize the distance between parts. Of course any choice about the former affects the solution of the latter, and there is the need to define *what* it is better to minimize (the mean distance, the maximum gap, etc.).

On the other hand, if we consider the second problem, there is need to reduce the assembly time by decreasing the number of fasteners without loss of quality: in this case a definition of "quality" needs to be provided, in order to generate the solution algorithm.

In this report we will denote with

- N, number of fasteners.
- $n = 60$, number of computational nodes.
- g_i, value of gap in node i, $i = 1...n$.

We can choose, as a measurement of quality, between lots of quantities: indeed, a suitable choice could be to minimize the total mean gap, as well as a weighted mean gap or the maximum gap.

For simplicity, we decide to test our solution algorithms on the total mean gap - bearing in mind that the algorithms we propose can be applied to any quantity (from now on, called the *objective function*), as they are independent from it.

In this framework, the objective function f is given by

$$f = \frac{1}{n} \sum_{i=1}^{n} g_i. \tag{1}$$

From a computational point of view, we use the maximum gap

$$g_{\max} = \max_{i=1,\ldots,n} g_i \tag{2}$$

as well, in order to decide where to optimally put a fastener.

2.1 Optimize Positions

In this first problem, the goal is to find the best position for a fixed number of fasteners in order to get a solution which maximizes the assembly quality, i.e. minimize the distance between parts.

This problem is solved in two steps: first, two different initial positions of fasteners are developed, both based on distinct concepts.

Secondly, we elaborate two diverse optimization algorithms for improving the initial positions. The first one is based on the idea of moving a fastener towards the largest gap in its neighbourhood, while the other relies on the idea of an equal distribution of fasteners.

2.2 Minimize Number of Fasteners

As previously mentioned, aircraft companies want to reduce assembly time in order to fasten their production. Hence, to reduce assembly time and save a lot of resources, it is desirable to have as less fasteners as possible, but keeping a tolerable assembly quality.

In this problem, the number of fastener is not fixed since our goal is to decrease it without loss of quality. Namely, by changing number of fasteners and positions of them, we have direct influence on assembly quality.

Notice that we have two mutually conflicting objectives, thus solving this kind of problem can be seen as more challenging.

3 Algorithms for Optimizing Positions of Fasteners

As already introduced, we are given ten different sets of initial gaps, on which we test our code. We decide to develop the solving algorithms by working on only one of them. As in reality it is not possible to find the best position of fasteners for each geometry (just try to think about measuring the gaps and applying the following algorithm for each airplane you have to assemble!), our solution should find the *best position* for a wide range of settings: to do so, we test our code on a new geometry, obtained as a mean of all the given sets. Unfortunately by doing so, we have to simulate ten times and calculate the gaps of our resulting geometry every time that we want to update the gaps. Nevertheless, the algorithms can be adapted easily on finding the best positions for a variable number of geometry sets.

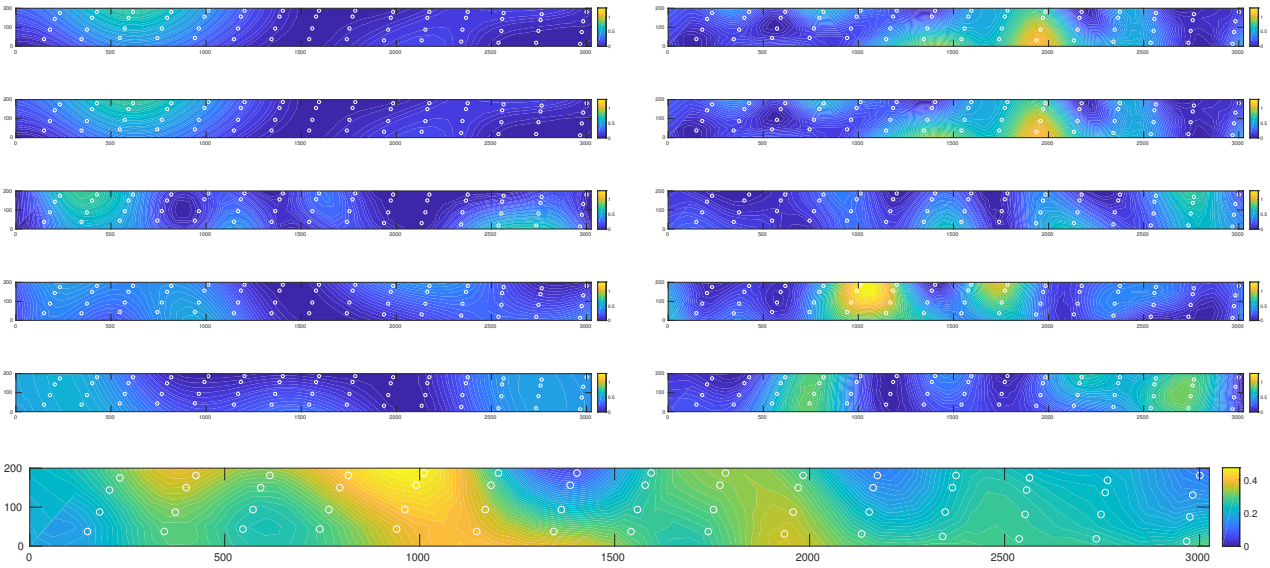

Figure 4: Above: different given initial settings of the junction area. Bottom: initial geometry of the junction area obtained by a mean of the given initial sets of gaps.

3.1 Initial Solution

The first step is to find an initial position of the fasteners. This can be done mainly in two ways, a deterministic and a random way.

The deterministic way (Alg. 1) is pretty simple, as it consists in putting one fastener per time. After calculating the gaps on an initial setting without any fastener, the next step is to find the node with maximum gap and try to minimize that one, by putting a fastener in the nearest hole to this node. Then the process is iterated until all fasteners are set.

On the other hand, the random algorithm is trivial: given N fasteners that need to be set, decide randomly where to put them - ideally spreading them over the surface. As

Algorithm 1 Initial solution - 1

N number of fasteners
for $i = 1...N$ **do**
  Run the solver
  Find the node with maximum gap
  Put the fastener in a hole that is closest to this node
end for

each hole is indexed, this process can be easily done by generating N random numbers, corresponding to the holes where to put the fasteners (see Algorithm 2).

Algorithm 2 Initial solution - 2

N number of fasteners
Generate N different random integers between 0 and n
Put the fasteners in the corresponding holes

Of course these two Algorithms are used to obtain an initial solution and there is no guarantee about the quality of it, even if we notice that Algorithm 1 already generates a relatively good setting: indeed it generates local optima at each step so, even though we are not sure that a global optima is reached, for sure we have generated one of the best position we can get.

After creating an initial position of fasteners, we now want to eventually optimize/improve this solution in order to satisfy the objective function. Here we present two different ways of improving the solution.

3.2 Improving Solutions - Method 1

The aim of Algorithm 3 is to repeatedly improve our solution until we have reached in some meaning an "optimal solution", according to our optimization criterion.

The idea for improving the solution is to find a direction where a fastener should be moved to decrease the objective value f. We decide, for each fastener, to compute vectors for each node in a neighbourhood of nodes. The vectors are having origin in the fastener and directions given by the node we are pointing at. The length of these vectors will depend on the gaps at the nodes and the size of our neighbourhood.

Because of our geometry of the wing part (which is wider than high), we decide to use an ellipse to define the neighbourhood of the fasteners, in order to capture the same amount of holes (circles, white) and nodes (points, black) in both directions (Fig. 5).

The neighbourhood to a fastener $\mathbf{x} = (x, y)$ is defined as

$$\mathcal{E}(\mathbf{x}) := \left\{ (\tilde{x}_i, \tilde{y}_i) : \frac{(\tilde{x}_i - x)^2}{a^2} + \frac{(\tilde{y}_i - y)^2}{b^2} < 1 \right\},$$

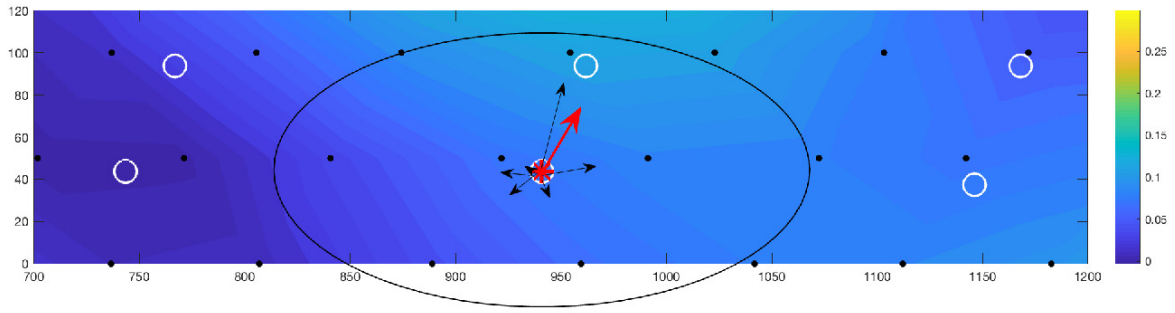

Figure 5: Elliptic neighbourhood around a fastener, vectors towards each node and resulting vector.

where $(\tilde{x}_i, \tilde{y}_i)$ are the coordinates of node i on the wing part and a and b are the major and minor semi-axis. This means that the neighbourhood is the set of nodes inside an ellipse centered in $\mathbf{x}$.

After computing the neighbourhood, the vector $\mathbf{v}_i$ corresponding to a node $\tilde{\mathbf{x}}_i \in \mathcal{E}(\mathbf{x})$ is calculated as

$$\mathbf{v}_i = \frac{(a+b)g_i}{2g_{\max}}(\tilde{\mathbf{x}}_i - \mathbf{x}).$$

Summing up all these vectors for each node in $\mathcal{E}(\mathbf{x})$, we obtain the direction $\mathcal{S} = \sum_{i \in \mathcal{E}(\mathbf{x}_k)} \mathbf{v}_i$ where to move the fastener. Of course, it could happen that the resulting movement takes the fastener outside $\mathcal{E}(\mathbf{x})$: in this case, we decided to rescale it to the border of the ellipse.

Then we find the coordinates of the nearest hole to the new position $\mathbf{x}+\mathcal{S}$ and eventually move each fastener to its new (or old) hole.

In order to adapt the choice of the maximum measure for the major and minor semi-axis a and b, we set three different set of values; in each iteration, we compute and simulate the gaps for every set of values and we pick the result that better improves our objective function.

In this way, we continue to iterate until we are not able to get any better solution.

3.3 Improving Solution - Method 2

Another idea for improving the solution is given by dividing the junction area into sub areas and working locally on each of them (Alg. 4). Indeed, the algorithm systematically tries to improve the solution by moving one fastener at a time to decrease the total sum of gaps.

We decide to split longitudinally the junction area, because of the geometry: in this way, we obtain an equal subdivision of the area we are working on.

Algorithm 3 Improving solution method 1

while the obj. f is improving and the max gap is not getting higher **do**
  Compute for each set of a, b the following:
  for each fastener $k = 1, \ldots, N$ **do**
    % Calculate the neighbourhood
    Define the ellipse $\mathcal{E}(\mathbf{x}_k)$ *with* a, b *s.t.* $\mathcal{E}(\mathbf{x}_k)$ *contains* $1/N$ *of all gaps* $g_{norm} \geq 1/N$ *or is of maximum allowed size* $a_{\max}, b_{\max}$.
    while $a < a_{\max}$, $b < b_{\max}$ and $g_{norm} < 1/N$ **do**
      for each node $i = 1, \ldots, n$ **do**
        if node $i \in \mathcal{E}(\mathbf{x}_k)$ & node i is not already stored **then**
          Store node i and its corresponding gap
        end if
      end for
      if $\mathcal{E}(\mathbf{x}_k) \neq \emptyset$ **then**
        Normalise the sum of the gaps g_{norm}
      end if
      Increment a and b
    end while
    % Move the fastener
    for each fastener $k = 1, \ldots, N$ **do**
      for each node $i = 1, \ldots, n$ **do**
        Compute the vector $\mathbf{v}_i$ for all nodes in the neighbourhood $\mathcal{E}(\mathbf{x}_k)$
      end for
      Sum all the vectors $\mathbf{v}_i$ as $\mathcal{S} = \sum_{i \in \mathcal{E}(\mathbf{x}_k)} \mathbf{v}_i$
    end for
    if $\mathbf{S} \notin \mathcal{E}(\mathbf{x}_k)$ **then**
      Restrict $\mathcal{S}$ to the border of the ellipse
    end if
    Find the nearest hole and put the fastener.
  end for
  Choose the best result
  Update objective function
end while

In this method, the algorithm first detects the subarea with the largest number of fasteners; then, each fastener (denoted by f_j in Alg. 4) is moved to the subarea that has the largest sum of gaps. Within this specific subarea the fastener is placed close to the largest gap.
After each movement, the gaps are recalculated and the algorithm check if this action has reduced the total sum of gaps.
If the answer is positive, then we have got a new and better solution. If it is not, we undo the action and try with the next fastener from the same subarea.
The algorithm stops when it finds a better solution.

Algorithm 4 Improving solution method 2

Divide the area into K sub-areas.
while $\sum_{i=1}^{n} g_i^{new} < \sum_{i=1}^{n} g_i^{old}$ **do**
  Detect the sub-area with the largest number of fasteners.
  Set k the number of fasteners in this subarea
  for $j = 1, \ldots, k$ **do**
    Detect the subarea with the largest sum of gaps
    Move fastener f_j close to the biggest gap in this area
    Simulate the new gaps
    if $\sum_{i=1}^{n} g_i^{new} < \sum_{i=1}^{n} g_i^{old}$ **then**
      Break out of the **for** cycle.
    else
      Undo the movement and check the next fastener.
    end if
  end for
end while

3.4 Results

In this section we will present the results we obtained by applying these two algorithms. As previously said, we test our code on an initial setting obtained as a mean of all the ten given initial gap settings.

We choose to find the best positions for $N = 10,\ 20,\ 30$ fasteners. We decide to report the mean values of the total gaps before and after the optimization methods.

In order to find the initial positions of the fasteners, after some tests, we agree to use the initial/improving algorithm in the following way:

(A) coupling the initial Algorithm 1 (let us call it "the perfect way") with the optimizing Alg. 3;

(B) the randomly Algorithm 2 is associated to Alg. 4.

This choice is motivated by the fact that, as Algorithm 4 needs few iterations more rather than the other to reach the optimal solution, it cannot be used with Alg. 1, as the latter already gives a quasi-optimized solution.

In the following plots the movement of the fasteners and the decreasing of the distance between parts can be easily observed.

We collect in the table the mean values of the gaps before and after the optimization process, for each choice of number of fasteners.

As we can see from the results in Table 1, we have achieved an improvement for both methods (A) and (B) with 10, 20 and 30 fasteners. By comparing *Before* from method

		$N = 10$	$N = 20$	$N = 30$
A	Before	0.0760	0.0327	0.0205
	After	0.0686	0.0270	0.0199
B	Before	0.0904	0.0344	0.0263
	After	0.0685	0.0236	0.0141

Table 1: Mean values before and after the optimization process.

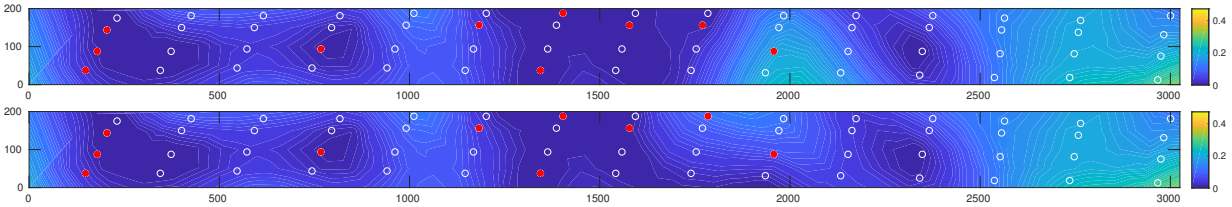

Figure 6: (A) before and after optimization results with $N = 10$.

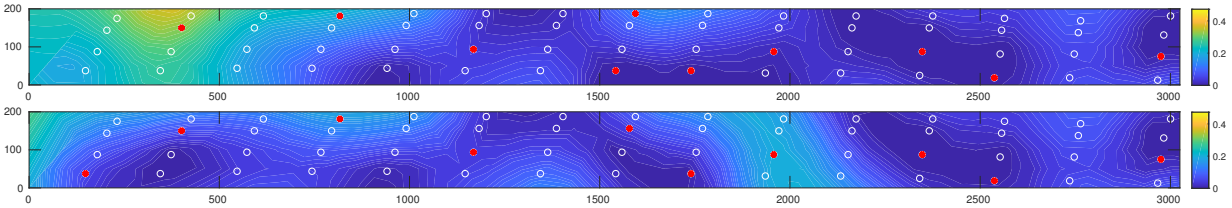

Figure 7: (B) before and after optimization results with $N = 10$.

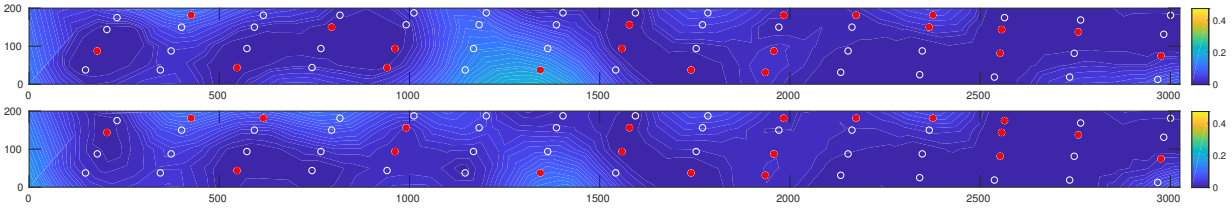

Figure 8: (A) before and after optimization results with $N = 20$.

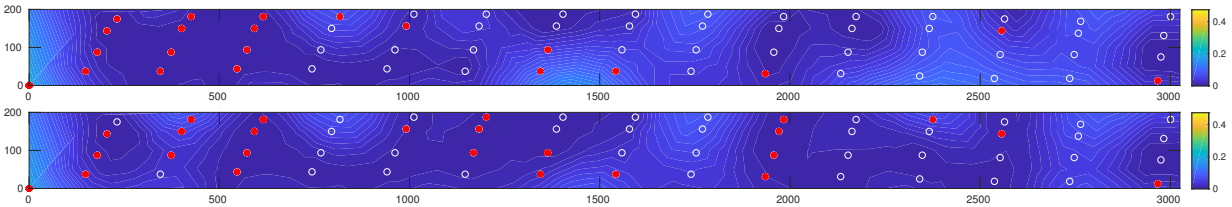

Figure 9: (B) before and after optimization results with $N = 20$.

(A) and (B) we can see that (A) already achieves a better solution than (B) – which is as we expected because (A) uses a heuristic approach for the initial setting while (B) just puts it randomly.

Indeed, when comparing the *After* of both methods we notice that method (B) achieves results that are a lot better than the ones from (A). We therefore conclude that (A), or

Algorithm 3, is a more local optimization while the redistribution of (B), or Algorithm 4 is able to do a more global optimization. For future optimization we recommend method (B).

Finally, we should have a look at the run time of the methods. The time consuming part of the optimization is the simulation of the gaps. Keep in mind that every simulation actually consists of ten (or how many different initial gaps you would like to consider) simulations. For the initial settings we need $N + 1$ simulation for the "perfect" initial setting, Alg. 1. The random initial setting, Alg. 2 only needs one simulation. For the "local" optimization, Alg. 3, 3 simulations per iteration are needed; for the "global" optimization, Alg. 4, N simulations per iteration are done. The latter scales a lot with the number of fasteners needed and is a lot more time consuming than the "local" optimization for a large number of fasteners.

4 Optimizing Number of Fasteners

As already stated, it is desirable to have as little fasteners as possible, but keeping a tolerable assembly quality. In this section we are presenting an algorithm to find such result. Since the number and positions of fasteners are directly correlated to the quality, the optimization problem has objectives in conflict and the problem can be seen as a multi-objective optimization problem.

4.1 Theory

When dealing with a multi-objective problem, it can be interesting to find solutions on the Pareto front, i.e. solutions where it is not possible to improve one objective value without worsening the other one. Since it is not possible for us to have theoretical knowledge about the optimal quality of our solutions, we have to loose on the theoretical restrictions on what a solution on the Pareto front is. Although it can be interesting to have this in mind and try to create methods that resemble a solution on the Pareto front.

One method to find solutions on the Pareto front is the so called ε-constraint method which could in some way be used for this problem. The problem can be formulated as

$$\begin{aligned} \text{minimize} \quad & \text{"number of fasteners"}, \\ \text{subject to} \quad & \text{"quality of assembly"} \geq \varepsilon. \end{aligned} \tag{3}$$

We would need an algorithm which reduces the number of fasteners in some systematically way such that the quality is worsen as little as possible when a fastener is removed. The value of ε also need to be decided. ε has to be less than the quality value we have when fasteners are placed in all holes and it should be larger than some minimum demand.

4.2 Practical Approach

The following steps explains a quite simple idea, by using our already created algorithms explained in section 3, for finding a Pareto optimal solution:

Step 0: Assign a value for ε in model (3) and simulate with $N = 1$ fasteners installed.

Step 1: Let $N = N + 1$ and execute the optimization for positioning the fasteners.

Step 2: If the ε-constraint in model (3) is satisfied, stop. Otherwise continue from step 1.

One issue with this idea is that we need to know how to measure the "quality of assembly" and which value it should have. This requires knowledge of which level of quality that is enough. One way to measure the quality is of course as we suggested with (1), so we get that $\varepsilon = \frac{1}{n}\sum_{i=1}^{n} g_i$.

Algorithm 5 is a practical approach for finding a result with good quality and few fasteners. By starting with no fasteners and increasing the number gradually we ensure the lowest run time possible when using Algorithm 4 for the positioning of the fasteners.

Algorithm 5 Finding the minimal number of fasteners

1. Assign quality value to ε
2. Number of fasteners $N = 0$

while Given quality is not achieved **do**

1. $N = N + 1$
2. Find best positioning of N fasteners: Algorithm 4
3. Calculate new gaps

end while

4.3 Results

Using algorithm 5 the results according to figure 10 were obtained. The algorithm was tested for different levels of ε and not surprisingly, more fasteners are needed to satisfy better quality. Figure 10 graphically shows the Pareto front and can be used to make a decision of how many fasteners to use versus what quality that should be satisfied.

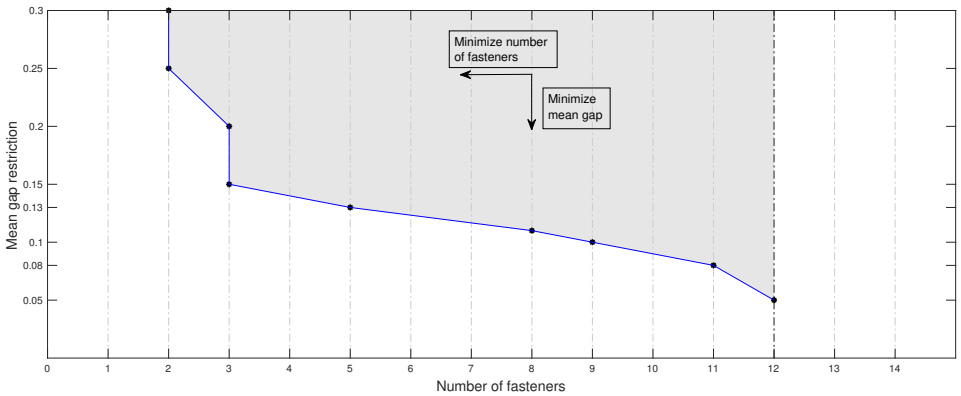

Figure 10: Number of fasteners obtained using algorithm 5 for different values of the quality restriction ε. The black dots on the blue line are the solutions found and every solution in the gray area are feasible but worse.

5 Conclusions

To summarize, we have achieved already very good results. We have developed two sets of initial settings as well as two methods for the optimization of the fastener positions. For problem 1 we have therefore managed to optimize the position of fasteners independently of the initial setting with both methods. Anyways, we have noticed that algorithm 4 has achieved much better results concerning a global optimization at the price of a higher run time due to a lot more simulations to be done.
We have then used this algorithm for problem 2 and achieved a Pareto front with the best designs.

Nevertheless, there is still some further development to be done.
First of all, the algorithms need to be improved concerning run time and placing method. Algorithms 3 and 4 could be combined in order to achieve a global optimization first by algorithm 4 and replace the fasteners in the sub-areas using algorithm 3. This could result into a very good optimization strategy.
Additionally, the algorithms should be tested using different objective functions. In this work, we have tested using the mean of all gaps or the sum of all gaps. Nevertheless, one could also use other objective functions, e.g. the maximum gap or different norms.
The results should then be compared to results achieved with other methods in order to verify them and one perfect solution should be found that is independent of the initial setting.
Finally, after improving run time and testing and improving the solutions, the optimization should be used on an actual plane geometry in order to reduce the costs and the installation time of assembling an airplane.

6 Group Work Dynamics

In our opinion, the group work during the Modeling Week has been almost ideal. We always kept everyone busy and motivated until the last minute and never got into any group work argument at all.

We started by doing a short discussion of the two problems and decided to first work on problem 1. Therefore, we collected some ideas of the basic algorithm and decided to start by finding an initial setting. We always worked in two groups, divided in a way that we could combine programming experience and both theoretical and practical knowledge. We regularly updated the other group about progress and brainstormed about questions and following procedures using the projector to present our codes and the blackboard to collect ideas. Whenever one group was waiting for the other, it added the new algorithms or ideas to the report. In consequence, the workload of the report after the Modeling Week has not been that high anymore. On the last day we worked on the presentation and spontaneously decided to try to solve problem 2 as well – which worked surprisingly well.

As we all came from different backgrounds, some of us had more programming experience while others were more into theory or used a practical approach. Therefore, we could divide ourselves in well-balanced groups that worked well together. As we were busy all the time, the work was well and equally distributed.

Of course, there were some things that benefited to the group work. First of all, we all came from different countries and spoke different languages. Therefore, it was crucial to speak English all the time – although it was sometimes hard to explain our ideas in a foreign language – and no-one felt excluded from the discussions. Furthermore, since we did not know each other at all before that work, we all felt obliged to stay polite and to appreciate everything that was said, also if it was not one's opinion. This is how we achieved a very motivated and pleasant working atmosphere. Additionally, as has been said before, we had all different backgrounds and were able to profit from each one's knowledge. One might have wondered that we have not referenced any additional sources at all – we did not need one. The algorithms were just developed by ourselves, the theoretic interpretation of the Pareto front for example came from one team member, the picture of the fastener and the practical knowledge from another... We have therefore omitted the references part from the template.
Challenging, on the other hand, has been e.g. that we first needed to get used to each other so that everyone dared to bring in whatever idea they had. Also, as most of the work needed programming it was sometimes hard to find work for those that were not that experienced in programming.
During the Modeling Week our instructor left us a lot of freedom for our group work. On the first day, she presented the problems and the solver that she gave to us, but apart from that, she left it all to us. We only talked to her about our approaches after a few days when she gave us some tips and clarified some misunderstandings of the task. Anyways, we could always come to her if we had questions. We have probably also achieved several solution techniques because we were given that much freedom. The overall relationship was friendly, polite, constructive and helpful.

7 Instructor's Assessment

The posed optimization problems were quite challenging, so the first task of fastener optimal positioning took the group the most of the time. It resulted in several optimization approaches that can be used not only separately but also combined. Despite the lack of time, the students managed to work on a second problem (fastener number reduction) and achieved good results even for half a day of work.

The summary of the group work was described in detail in paragraph 6. I fully agree with every statement. The group was very well-organized from the first to the last day of the Modelling Week. Especially I would like to thank them for supporting each other, explaining and sharing ideas within the group as it made everyone to be involved in the process. This was a good experience for all of us.