Anastasija Cumika

Company: Datadvance

Internship adviser: Anton Saratov

07 August 2020

Topic: Multiobjective optimization of the airplane wing

Internship Report

1. Intro (motivation)

Wing shape research has been a big topic for almost 100 years. The goal of the shape optimization is to develop more efficient and cheaper aircraft designs. Historically it happened that profiles are defined with points. When people started thinking about the simple optimization, the profile parametrization question came up. The most straightforward thinking is that we can independently put coordinate points and that way we can get all possible profiles. However, in terms of optimization this is a bad idea because it would be very cost expensive computation. Our goal is to get feasible and smooth profile. The feasible profile does not have any self-intersections between the top and bottom parts. In the smooth profile the variation between the consecutive points and its nearest neighbors is not large. If the profile is not smooth, its aerodynamic properties are falling down dramatically.

The probability of getting a feasible and smooth profile is so low that for solving such an optimization problem we need so huge computational resources that do not exist yet. Therefore, scientists have developed parametrization techniques where point's coordinates are not independent. For example, NACA airfoils, that have been developed in the late 1920s, provide simple but robust parametrization with 3,4 or 5 parameters. NACA parametrization was good, but nowadays aircrafts require more flexible parametrization where we would have more control over the leading, trailing edge and curvatures in general. PARSEC algorithm provides more sensitive parametrization. However, it still limits the variation of profiles that can be created. The current problem is that many optimization algorithms are restricted with specific airfoil profile parametrization algorithms such as NACA or PARSEC. Therefore to get outside this parametrization restriction Datadvance offered to work with DR (Dimension Reduction) algorithm for wing shape optimization. DR greatest benefits are that it does not have analytical description and parametrization components are orthogonal. We mostly focused on the cruise case, where the lift is set as a constraint, but we minimize the drag. Drag minimization, reduces the fuel consumption of the aircraft.

2. Airfoil Parametrization

There exist several ways of airfoil parametrization. The most popular are NACA parametrization. We looked at NACA 4-digit algorithm. Also we looked at the PARSEC parametrization. Finally DR algorithm was used to generate new airfoil shapes, based on the existing airfoils.

2.1 NACA parametrization

The NACA airfoils are constructed by combining a thickness envelope with a camber or mean line. The airfoil geometrical construction is shown on Figure 1.

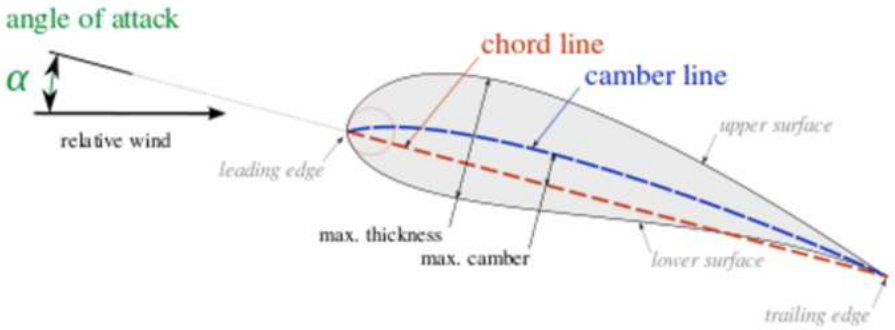


Figure 1 – Airfoil geometrical construction definitions

The equations which describe this procedure are shown below were u – upper and l –lower part of the airfoil. They depend on the thickness function $y_t(x)$ and the camber line function $y_c(x)$:

$$x_u = x - y_t(x)\sin\theta$$
$$y_u = y_c(x) + y_t(x)\cos\theta \qquad \text{(A-1)}$$

and

$$x_l = x + y_t(x)\sin\theta$$
$$y_l = y_c(x) - y_t(x)\cos\theta \qquad \text{(A-2)}$$

Camber line slope is expressed in the expression below:

$$\theta = \tan^{-1}\left(\frac{dy_c}{dx}\right) \qquad \text{(A-3)}$$

Thickness and camber line functions can be found with either 4, 5 or 6-digit NACA algorithm. We paid our attention to the basic NACA 4-digit algorithm. NACA number for the airfoil has 3 numbers - NACA MPXX, where:

- XX is the maximum thickness, t/c, in percent chord (XX/100).
- M is the maximum value of the mean line (max camber) in hundredths of chord (M/100)
- P is the chord wise position of the maximum camber in tenths of the chord. (P/10)

Even though NACA numbering is usually done with integer numbers, as one can observe below, NACA numbering is not restricted to integers. We will use this idea further in our algorithms. 4-digit NACA is parametrized with only 3 parameters. This way is very robust and any number (within boundaries) will give us feasible airfoil. However, we can vary only big features like thinness and main curvature. While, we do not have a control over the trailing edge, leading edge and other small variation.

The thickness distribution of a 4-digit NACA airfoil for chord length = 1 is expressed as follows:

$$y_t = \frac{T}{0.2}\left(a_0 x^{0.5} + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4\right)$$

$a_0 = 0.2969$ $a_1 = -0.126$ $a_2 = -0.3516$ $a_3 = 0.2843$
$a_4 = -0.1015$ or -0.1036 for a closed trailing edge

The coefficients “a” shown are for the case of the thickness to be 20% of the chord length. Also, a_4 coefficient varies based on what kind of trailing edge we are using – closed or open.

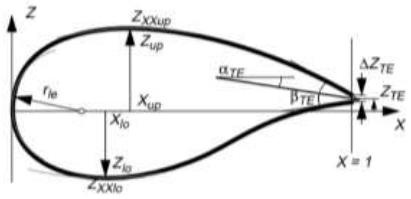
The camber line is expressed as follows:

$$\left.\begin{aligned}\frac{y_c}{c} &= \frac{M}{P^2}\left[2P(x/c)-(x/c)^2\right]\\ \frac{dy_c}{dx} &= \frac{2M}{P^2}\left(P-(x/c)\right)\end{aligned}\right\} \quad \left(\frac{x}{c}\right) < P \qquad \text{(A-7)}$$

$$\left.\begin{aligned}\frac{y_c}{c} &= \frac{M}{(1-P)^2}\left[1-2P+2P(x/c)-(x/c)^2\right]\\ \frac{dy_c}{dx} &= \frac{2M}{(1-P)^2}\left(P-(x/c)\right)\end{aligned}\right\} \quad \left(\frac{x}{c}\right) \geq P \qquad \text{(A-8)}$$

2.2 PARSEC parametrization

PARSEC parametrization algorithm involves eleven parameters shown below:



Description	Symbol	Description	Symbol
Leading Edge Radius	R_{LE}	Lower Crest Curvature	Z_{XXLO}
Upper Crest Abscissa	X_{UP}	Trailing Edge Ordinate	Z_{TE}
Upper Crest Ordinate	Z_{UP}	Trailing Edge Thickness	ΔZ_{TE}
Upper Crest Curvature	Z_{XXUP}	Trailing Edge Direction	α_{TE}
Lower Crest Abscissa	X_{LO}	Trailing Edge Wedge Angle	β_{TE}
Lower Crest Ordinate	Z_{LO}		

PARSEC algorithm gives a better control over the curvature of the airfoil, as well as, a better control of the trailing edge, which is a very important part of the airfoil. However, PARSEC algorithm is not as robust as NACA algorithms. Setting wrong values to parameters lead to the unfeasible profile. Examples of unfeasible airfoils obtained with PARSEC are shown in Figure2

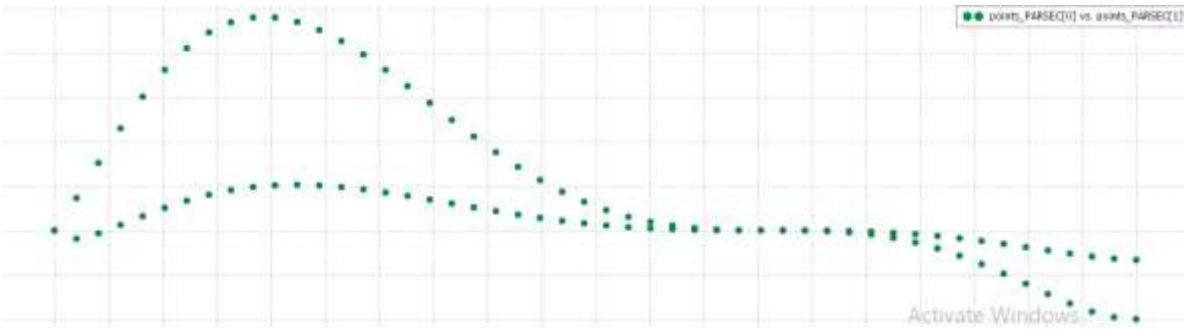


Figure 2 – Unfeasible airfoil obtained by PARSEC parametrization algorithm

The airfoil point coordinates (thickness envelope) are found with the following equations:

$$z_{up} = \sum_{i=1}^{n=6} a_{up}^{i} \cdot x^{i-\frac{1}{2}}, \qquad z_{lo} = \sum_{i=1}^{n=6} a_{lo}^{i} \cdot x^{i-\frac{1}{2}}$$

The coefficients “a” are found from the parameters by solving the following matrix equations:

$$C_{up} \times a_{up} = b_{up}, \qquad C_{lo} \times a_{lo} = b_{lo}$$

$$C_{up} = \begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ p_2^{\frac{1}{2}} & p_2^{\frac{3}{2}} & p_2^{\frac{5}{2}} & p_2^{\frac{7}{2}} & p_2^{\frac{9}{2}} & p_2^{\frac{11}{2}} \\ \frac{1}{2} & \frac{3}{2} & \frac{5}{2} & \frac{7}{2} & \frac{9}{2} & \frac{11}{2} \\ \frac{1}{2}p_2^{-\frac{1}{2}} & \frac{3}{2}p_2^{\frac{1}{2}} & \frac{5}{2}p_2^{\frac{3}{2}} & \frac{7}{2}p_2^{\frac{5}{2}} & \frac{9}{2}p_2^{\frac{7}{2}} & \frac{11}{2}p_2^{\frac{9}{2}} \\ -\frac{1}{4}p_2^{-\frac{3}{2}} & \frac{3}{4}p_2^{-\frac{1}{2}} & \frac{15}{4}p_2^{\frac{1}{2}} & \frac{15}{4}p_2^{\frac{3}{2}} & \frac{63}{4}p_2^{\frac{5}{2}} & \frac{99}{4}p_2^{\frac{7}{2}} \\ 1 & 0 & 0 & 0 & 0 & 0 \end{vmatrix}$$

$$b_{up} = \begin{vmatrix} p_8 + p_9/2 \\ p_3 \\ \tan(p_{10} - p_{11}/2) \\ 0 \\ \frac{p_4}{\sqrt{2p_1}} \end{vmatrix}, \qquad b_{lo} = \begin{vmatrix} p_8 - p_9/2 \\ p_6 \\ \tan(p_{10} + p_{11}/2) \\ 0 \\ \frac{p_4}{\sqrt{2p_1}} \end{vmatrix}$$

Both NACA and PARSEC algorithms were implemented in python code. As an output we get a “dat” format file with the airfoil points coordinates that can be used for Xfoil or Ansys computations.

2.3 DR parametrization algorithm

Dimension reduction algorithm is based on the PCA (principal component analysis). DR takes the multidimensional domain and compresses it to a lower number of features (modes). In the introduction, we mentioned that DR algorithm has two great benefits. First, DR does not involve analytical description. DR extracts the most important linear combination of features based on the dependencies in the sample provided by the user. Multidimensional domain can be compressed to any number of features. The second benefit is that all the features in the compressed state are orthogonal. The first feature will have the most influence, second one has less and so on. In the airfoil case, first feature would carry the idea that airfoil must have some thickness. The second feature might have the information that the airfoil is stretched in the horizontal direction. Every consecutive feature has less overall weight, however it is very important for the airfoil fine tuning, which is what we are looking for.

In this work, pSeven Generic Tool for Dimension Reduction with feature extraction were used to build the model for dimension reduction. The model was trained in the airfoil data base that consists of 199 airfoils. Each airfoil has 59 coordinate points. All coordinate points we consider as a separate parameters/features. We have chosen to make a DR model to have 6-dimetnions in the compressed state. We can generate random vectors of size 6 and decompress them to get a random airfoil. Below on Figure 3, we demonstrate 3 airfoils generated with DR algorithm implemented in python. Vectors for the airfoils were:

1. [-0.00053118, 0.00172369, 0.00307265, -0.00329883, 0.0089273, -0.00141753]
2. [-0.04186426, -0.00089491, -0.01361736, -0.00736629, -0.00495626, 0.00961083]
3. [-0.00879375, -0.02230005, -0.01547194, 0.0045737, 0.00873982, 0.00493302]

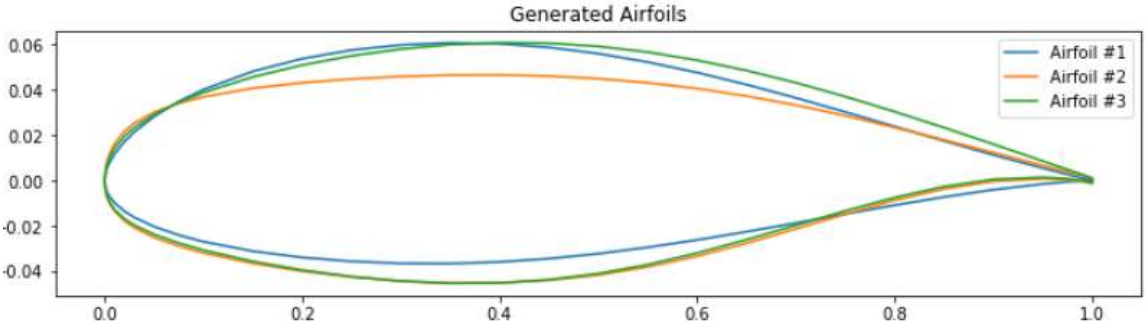


Figure 3 – Airfoils generated with the DR algorithm

3. DR Optimization: 2D case in pSeven

Optimization was implemented in pSeven software. It consists of 2 parts: Finding linear constrains for DR and optimization workflow. If we did not use linear constrains, most of the generated airfoils would be not feasible, as explained in section 3.1, and optimization would take much more time to process large amount of unfeasible design points.

3.1 Linear constrains for feasible airfoils.

While DR gives us a great freedom in creating various airfoils with a fine tuning, the problem, that we face is – what boundaries does each mode must have. Theoretically, first mode would have the largest boundaries, while last modes will only have to be changed slightly. However, we want to give some freedom for the model. However, when we give too much freedom and randomly choose vectors for the compressed airfoil, there is a high chance that decompressed airfoil does not look good. On the pictures below one can observe the examples of bad airfoils. For example, some airfoils have places where top part is crossing the bottom part. Also, airfoils might have some unrealistic curvature.

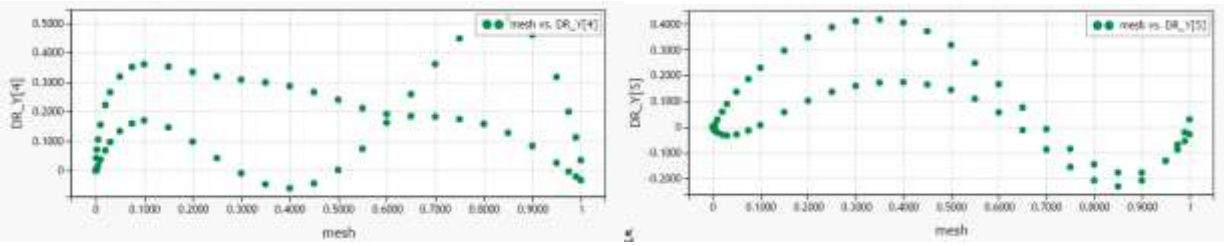


Figure 4 – The examples of bad airfoils generated with DR algorithm

We do not know what boundaries we should give to each mode in order to get good airfoils. Therefore we have built the workflow, shown in Figure 5 that generates many airfoils and puts each airfoil trough the check block. We use in build DR model builder and decompressor blocks.

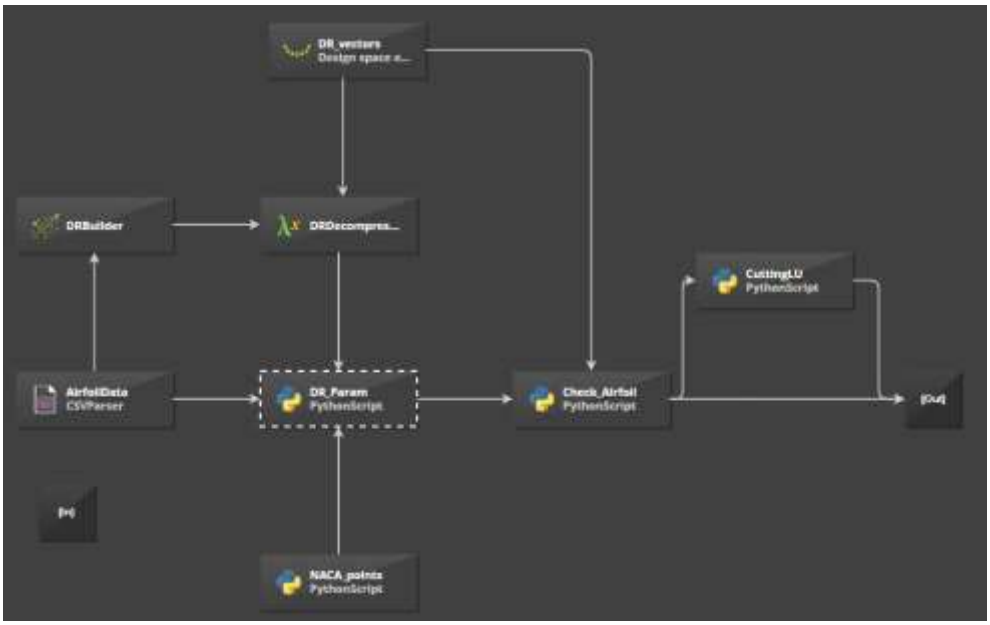


Figure 5– Linear constrains workflow

In the “DR_Param” block we create the linear combination of the DR generated airfoil, mean airfoil (from AllFoilData) and the reference airfoil (generated NACA airfoil in the “NACA_points” block. We use the following linear combination formula:

DR_Y=ref_airfoil-meanairfoils+gen_airfoils

Below we demonstrate how a generated airfoil (on the left) differ from the airfoil with the linear combination (on the right) mentioned above. One can observe that linearization makes the generated airfoil more realistic in terms of its physical characteristics, especially at the trailing edge. In addition, with the linearization combination we get a more smooth profile.

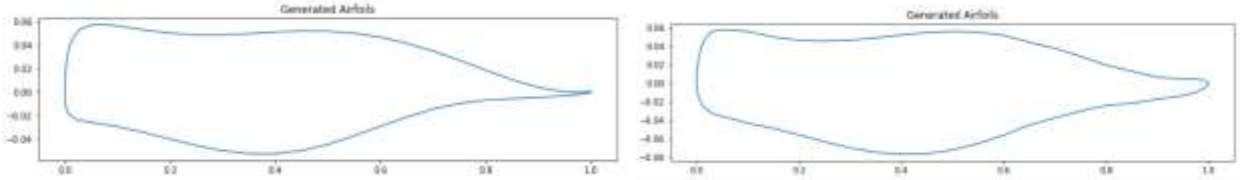


Figure 6 – On the left, airfoils directly decompressed form the DR mode. On the right, the same airfoil with in the linear combination with reference and mean airfoil.

In the “Check_Airfoil” block we check:

1. Self-intersections (top part with bottom part)
2. upper airfoil side remains at positive y-values and has proper curvature sign for $x < 0.5$
3. lower airfoil side remains at negative y-values for $x < 0.2$

In addition, we had to reduce the random vector generator boundaries from (-1,1) to (-0.07,0.07) because in the very big random domain, we geta very low percent of a good airfoils. We collect all the good vectors (vectors that give us an airfoil that passed all the test) and in “CuttingLU” block we build the linear constraint for our 6-dimentional space. Further we use this constraint in our optimization workflow, so that DOE block generates vectors only within this constraint.

On the picture below, we show the 3-dimentional graphical representation of how we build the constraint. Points in space are the design points – a vector of three numbers. With blue color we have good points and with pink color we have bad points. With the planes we cut out the domain parts that have only bad design points. These planes are our linear constrains. We save the coefficients that would be used to build the linear constraint to the files.

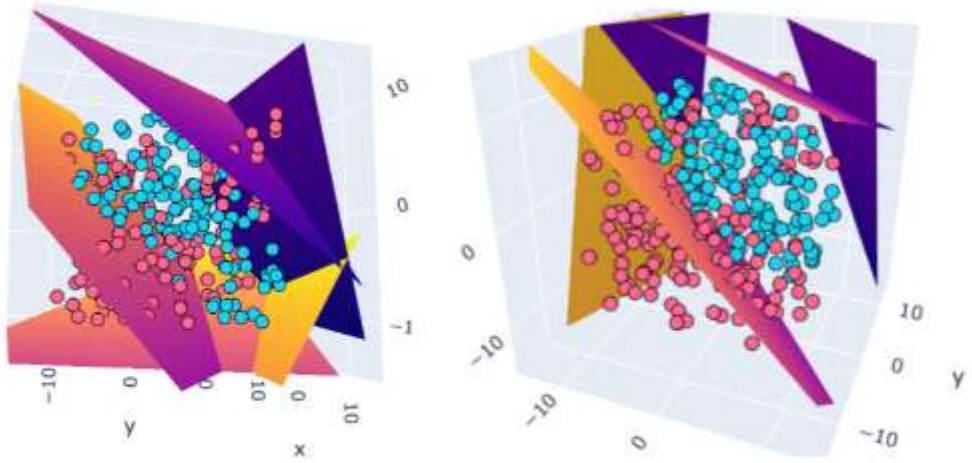


Figure 7– Linear constraint graphical representation in 3-dimentional space

3.2 Building the optimizer

The workflow for the optimizer is shown on Figure 8. The python script in "L" block builds the linear constraint from the coefficients that we have found with the Linear Constraint workflow. With "DSE" block new vectors in the 6-dimentional space are generated and in the in-build decompressor we create the airfoil 59 coordinate points. In "PythonScript" block we use the same linear combination of generated, mean and reference airfoils as in the Linear Constraint workflow.

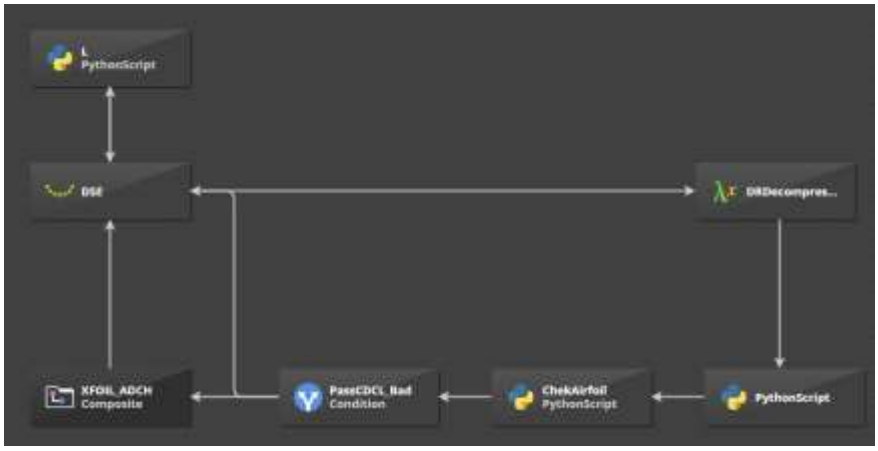


Figure 8– 2D optimizer workflow with Xfoil integration

Linear constrains are decreasing the number of bad airfoils, however does not eliminate them. As we could also see on Figure 7 some “bad points” still stay within the “good area”. Therefore we add an extra check for the airfoils in the optimization workflow. Here we check only for self-intersections, because this is what causes fatal errors when XFOIL is trying to compute lift and drag coefficients. “PassCDCL_Bad” block is controlling, so that only good airfoils get to the Xfoil computational workflow. If the airfoil is bad, NaN values are returned to “DSE” block for CD and CL variables.

On Figure 9 the “DSE” block is demonstrated. One can observe that we have three responses: L constraint, CD (drag coefficient) minimization and CL (lift coefficient) maximization. L has a size of 500, because, if we speak in terms of the example of 3D representation, we have 500 planes that constrain our domain.

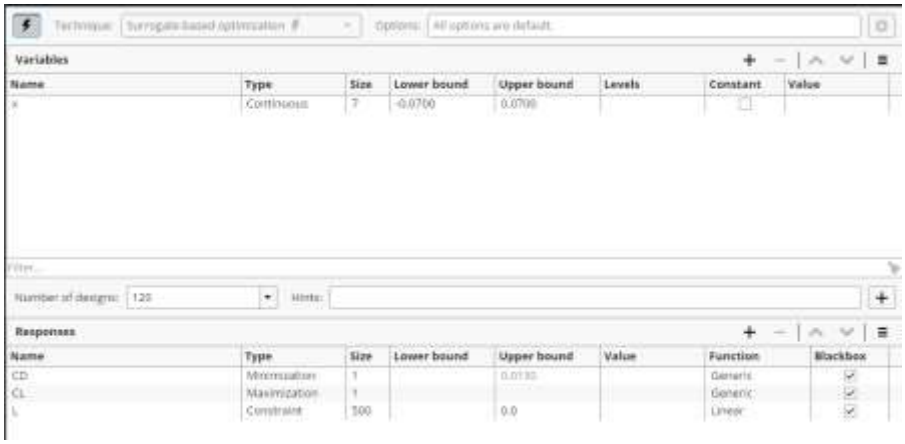


Figure 9– “DSE” block configurations for the workflow if the 2D optimizer with Xfoil integration

Xfoil computation workflow shown below consists of three parts. In the first text block we create the script for the Xfoil software. There we have computational parameters such as Mach number, Reynolds number, angle Alpha and number of interactions. “xfoil.exe” block runs the software and executes the prewritten script in Xfoil. The second text block is used to retrieve the results from the file that Xfoil has generated after the run. We take the results for lift and grad coefficients for each airfoil.

Figure 10– “DSE” block configurations for the workflow if the 2D optimizer with Xfoil integration

The results of the optimization run are shown on Figure 11. This optimization run was performed with 0.3 Mach and 30000 Reynolds number. Unfortunately, Xfoil is not stable for high Mach numbers. The picture on the right is just the enlarged area of the Pareto Frontier area from the picture on the left. One can observe that we see ~10 points at the Pareto Frontier. We decided to take three good points and perform the validation calculation in Ansys fluent software.

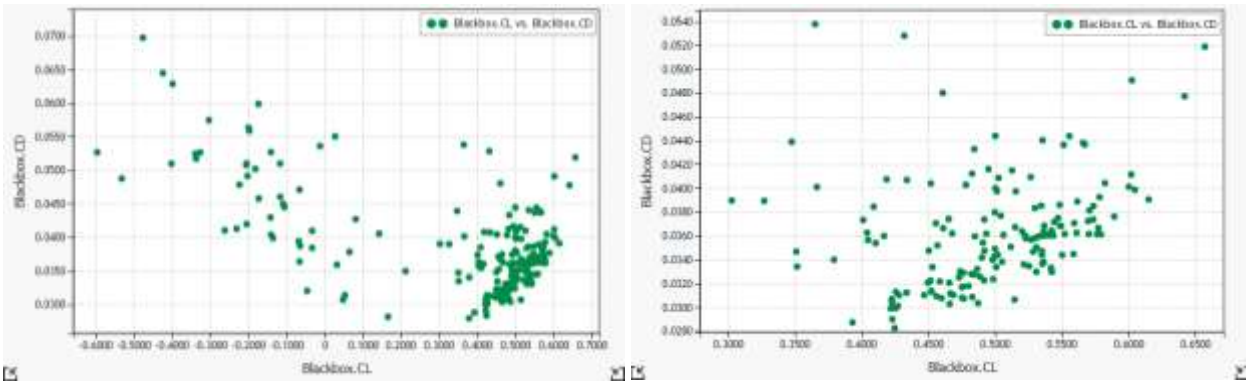


Figure 11– Results of the 2D optimizer workflow with Xfoil integration

3.3 Result validation via Ansys calculation

When we are generating different profiles with DR, even after the check, we do not always get a good smooth profile. Xfoil software does not use mesh but paneling method for the analysis of the flow field around the airfoil. Panels might make a profile smoother that it actually is, therefore Xfoil computation might not be good enough. Couple profile that gave a good result in the optimization run were chosen to be tested in Ansys fluent solver.

Mesh building and computations were performed in the Ansys workbench. First, a good mesh has to be created. Structured mesh is the best choice for airfoil computations since it usually tend to provide more stable results. The example of a structured mesh is shown in Figure 12. The k-omega SST viscous model was used in the solver. The solition workflow was validated on the NACA6412 airfoil. Results obtained for the lift coefficients in Ansys had 6% error from the results in the Zubin Zaheer paper (*CFD analysis of the performance of different airfoils in ground effect*). Results for the drag coefficient were only 2% of from the Zaheer results. We are working with drag minimization, therefore such a good result on the drag is good for us.

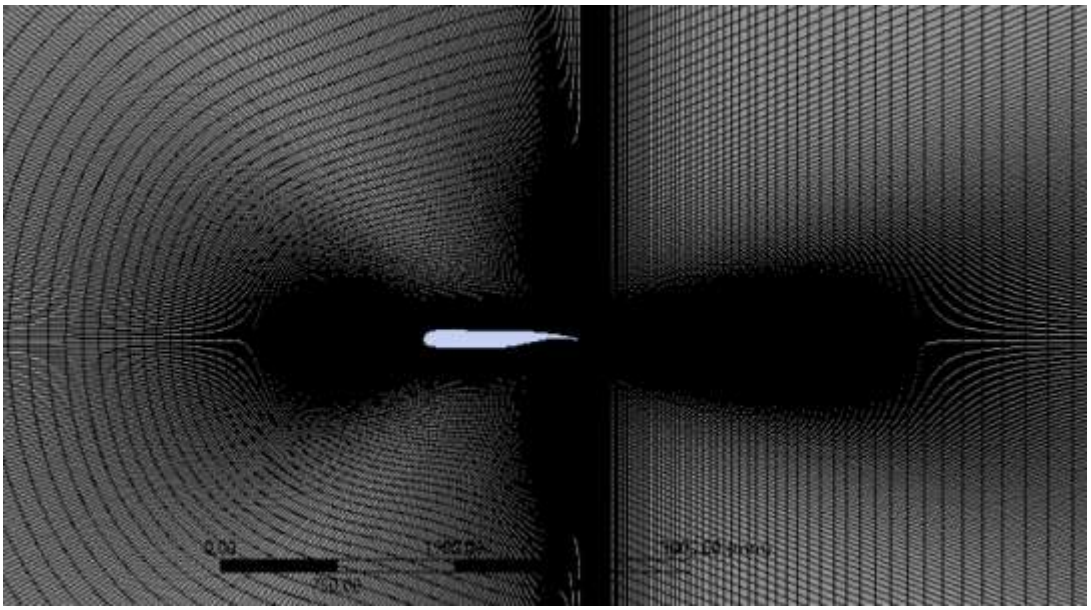


Figure 12– The example of a structured mesh used for one of the DR profiles

For profiles generated with DR, extra mesh sensitivity study was required because DR profiles are less smooth. DR profiles required ~2x more refined mesh than a smooth NACA airfoil. From the Xfoil optimization run we took four airfoils to test in Ansys. Below we show an example of the velocity field around one of the DR airfoils. In Xfoil, this airfoil draf coefficient (CD) was 0.03035 and lift coefficient (CL) was 0.4871. For the comparison, NACA6412 airfoil has CD of 0.013279 and CL of 0.65437. NACA airfoil is very smooth and in general a chance that DR airfoil has drag coefficient 4 times less than NACA airfoil is very low. In Ansys, the coefficients for this DR airfoil are: CD = 0.010185 and CL = 0.343991. Ansys is giving more real results. Therefore, we might say that optimization of DR airfoils with Xfoil solver is not the robust decision. In future, it would be a great to build a 2D optimization with Ansys solver.

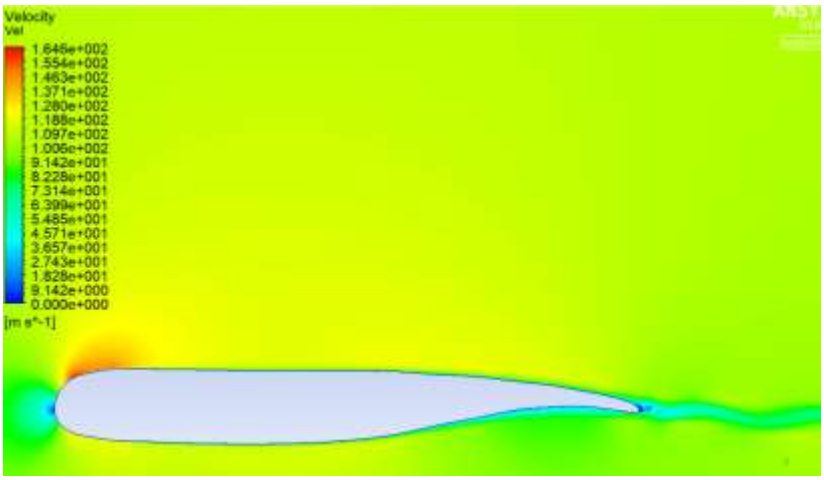


Figure 13– Velocity field around the DR profile.

4. NACA 3D wing from a profile section

After working with a 2D model, we moved on to the 3D case. Parametrization of the 3D wing is a complicated process because besides the base profile it also has changes along the extrusion length. There is a known technique to build a 3D wing based on the set of profiles that are set at the cross sections along the extrusion. Profiles may also have different angle of attack, that way we include a twist in the 3D model if the wing. The space between the profile cross section is a smooth transition from one profile to another. Using a set of base profiles is a good approach, but if we want to have a good fine tuning we might want to use many base profiles. Such a parametrization might be costly.

4.1 Building the 3D wing model

In our work we want to stay on the border of giving the model a good freedom to generate a variety of designs with a fine tuning. However, we want to preserve the profile basic characteristics. Out approach in generating the 3D wing involves one base profile. In this example, we started working with 4-digit NACA profile. We use the polynomial extrusion as a function of the wing extrusion length L shown in the equations below. Alpha, betta and gamma are the coefficients that denote by how much the specific profile parameters are changed along the length L.

$$M(L) = \alpha_1 L^3 + \alpha_2 L^2 + \alpha_3 L + M_0$$

$$P(L) = \beta_1 L^3 + \beta_2 L^2 + \beta_3 L + P_0$$

$$T(L) = \gamma_1 L^3 + \gamma_2 L^2 + \gamma_3 L + T_0$$

In addition, the scaling factor is used to narrow down the wing as the length L increases. Using the formulations described above the wing is built in the Solidworks via the parametrized macros script.

4.2 3D Wing generator and solver workflow

For the physical computation a separate workflow was used. As an input it takes nine geometric parameters (M, P, T, alphas, bettas and gammas). In "GenSetFoils" we generate many profile sections along the length L based on the formulation shown in the equation above. Then we pass this data to the macros generator, where a macros script for the specific design point is obtained. In the "Run_Macro_SW" block a 3D model of a wing is generated in Solidworks. "ICEM_CFX" block generates a mesh for the current 3D wing model and performs calculations in CFX solver. In the "Results" block we retrieve lift force Fy and drag force Fx results and send them to the output.

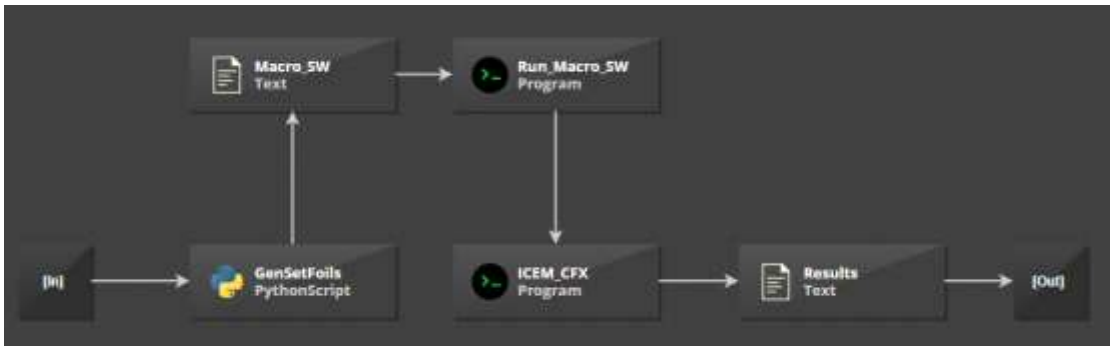


Figure 14– 3D wing generator and solver workflow

4.3 3D Optimization workflow

Our optimization workflow consists of three blocks: DSE, its linear constrains and a composite solver block that was described above. For current optimization we have total of 12 variables that include three NACA parameters for the base profile (m0, t0 and p0), as well as alphas, bettas and gammas (three of each). Linear constrains are for the M (L), P (L) and T (L) parameters, so that they stay within the NACA parametrization allowed borders.

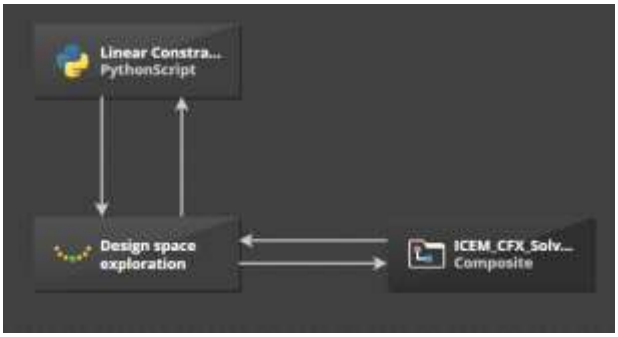


Figure 15– 3D wing generator and solver workflow

In this work we focused on looking at the cruise case, so the goal is to minimize the drag, which leads to the fuel saving. We are not looking for the lift maximization, however we want to find some lift force that we set as a minimum. To do so, it was decided to calculate lift force for the wing with the base profile NACA 4412 that has no changes along the L (alphas, bettas and gammas are set to 0). The 3D model of this profile is shown on the pictures below. This base 3D model has Fy =2138 N.

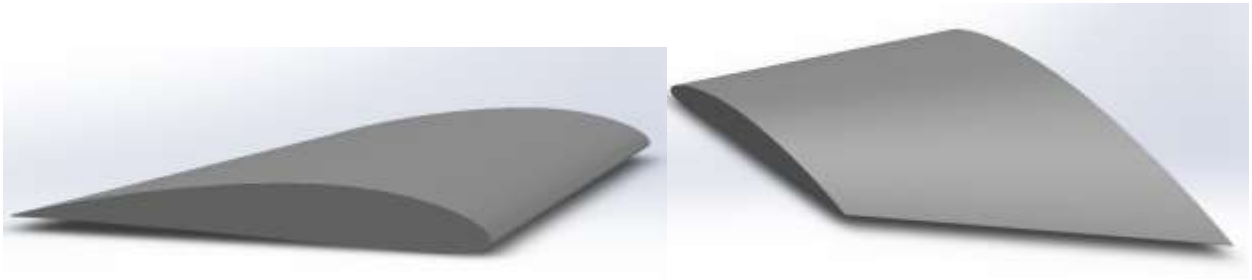


Figure 16– 3D wing generator and solver workflow

Optimization workflow is set with the constrain Fy > 2138 N and is aimed to minimize the drag. Current calculations were performed at Mach number equal to 0.5. In the first optimization run it was chosen to variate NACA parameters and gammas, therefore base profile and it's thickness along the extrusion length L are variated. Alphas and bettas are set to 0, therefore we do not have any curvature change along the wing. Thickness has a big effect on the aerodynamic parameters that is why gamma variation was chosen. The results for this optimization run are shown on Figure 17. We have set 130 number of design points, however, we got many NaN results because many designs did not converge. Therefore, on the plot there is not as many points as we would want to. From this plot one best point (highlighted with red), second best (highlighted with blue) and another good point that had best Fy/Fx ratio (highlighted yellow) were chosen. Best point has the least drag within the lift force constrain:

- Red point -> Fy = 2449 N (CL = 0.551) and Fx = 316.4 N (CD = 0.0714)
- Blue point -> Fy = 2811 N (CL = 0.551) and Fx = 345.5 N (CD = 0.0714)
- Yellow point -> Fy = 4940 N (CL = 0.551) and Fx = 706.6 N (CD = 0.0714)

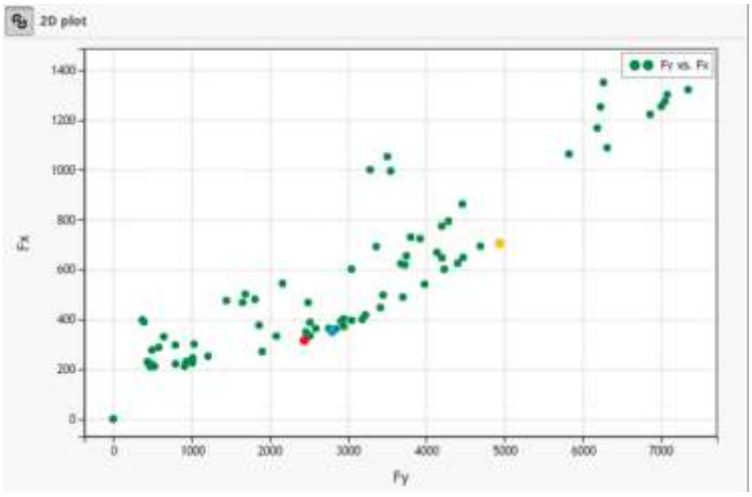


Figure 17 – Results of the optimization with base profile and thickness variation.

On Figure 18 all three designs are shown with the pressure distribution and airflow velocity streamlines. In the left column the red point design is shown, middle column – blue and right column – yellow. One can observe that the best wing design trailing edge is straight. Even though the best design point is thick, straight trailing edge lead to the low drag. Also, the third point has an interesting design. This wing is very thin, and it has a trailing edge with a high curvature. This wing has big drag, however it also provided a good results for the lift. One can observe how the airflow is changing its direction around the trailing edge.

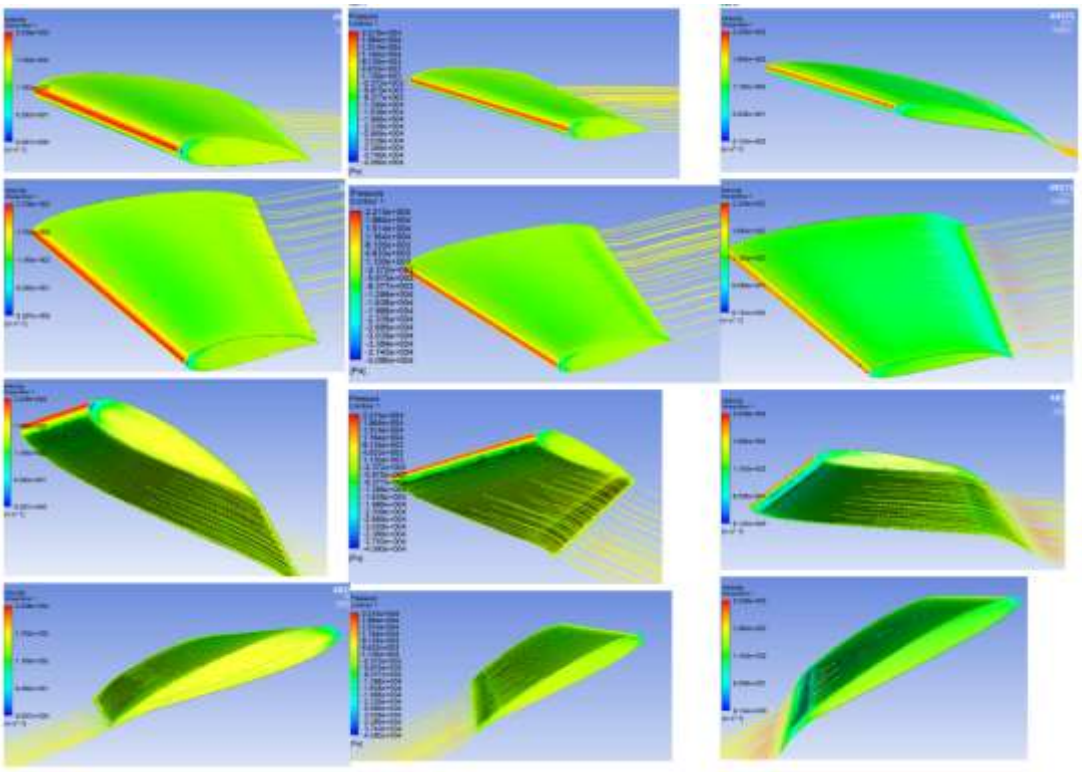


Figure 18 –Pressure distribution and airflow streamlines for the three points chosen form the optimization results. Red point – left, blue point – middle and yellow point – right columns.

Another optimization run was performed, where the base profile NACA4412 was chosen (because it has good aerodynamic characteristics for high speeds). In this run all the constants (alphas, bettas and gammas) were variated. This approach, would let us find the best 3D wing design and see what characteristics these designs have.

The results are shown on the plot in Figure 19. One can observe that the design pints are all pretty close to the Pareto front. In the previous run, where base profile parameters were variated some design points were far away from the Pareto front. From that we can conclude that base profile has a high effect on the aerodynamic characteristics of the wing. Two best design points are highlighted with red and yellow:

- Red point -> Fy = 2166 N (CL = 0.491379) and Fx = 295.5 N (CD = 0.0662517)
- Yellow point -> Fy = 2190 N (CL = 0.551) and Fx = 295.2 N (CD = 0.0714)

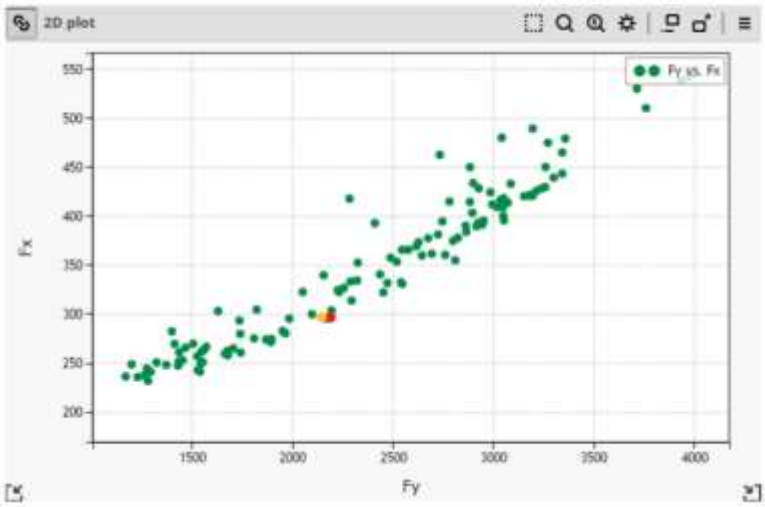


Figure 19 – Results of the optimization with constant (NACA4412) base profile and all coefficients variations.

On Figure 20 all both designs are shown with the pressure distribution and airflow velocity streamlines. In the left column the red point design is displayed and in the right column the yellow point design is displayed. One can observe that both wings look almost identical. Coefficients also have similar values for the most part.

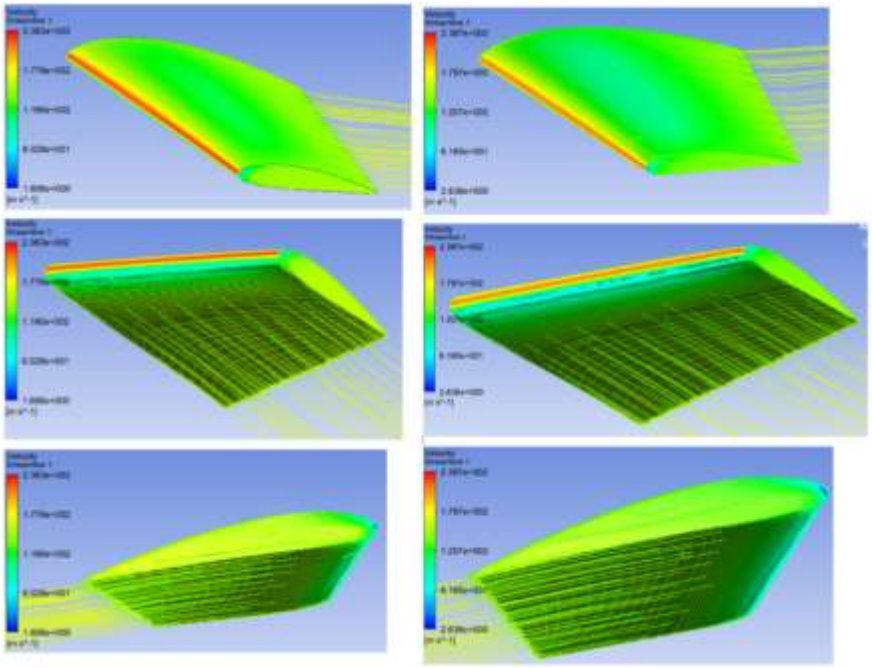


Figure 20 – Pressure distribution and airflow streamlines for the two best points form the optimization results in Figure 19. Red point – left column and yellow point – right column.

With this NACA 3D parametrization we have learned how to create and work with the workflow. It was observed, that the workflow where we variated base profile and thickness parameters had issues with coming to close to the minimum lift constraint (minimizing the drag). The workflow where we choose a base profile and variated all coefficients gave couple of good points right at the Fy constraint. We were

able to get wings within the constraint and with the drag force below 300 N. However there is still a lot of work that can be done in order to get good results. One of the main goals is to build a DR 3D parametrization based on the base profile that we find with the 2D DR Parametrization.

5. Future improvements

In 2D optimization part it was revealed that Xfoil capabilities are not robust for DR profile flow field computations. Therefore, we want to build a 2D optimization workflow with Ansys fluent solver. With a good and validated 2D optimization workflow we can find vector space area that results in a good and effective profile. Then we take this profile and use it as a base profile for the DR 3D wing parametrization. The profile would not be set to constant, however the variation of the vectors would be minimal. The 3D parametrization will be performed in the same manner as for the NACA profile. While for NACA we had 3 parameters for the base profile and 3 constants for each parameter (total of 12 parameters), for DR we will have 6 parameters for the base profile that result in total of 24 parameters.

Optimization for so many parameters might be complicated, so we would have to use iterative approach. We know that the first modes have the greatest influence on the aerodynamic coefficients. Therefore, we can run the optimization varying only the one or two first modes coefficients (alpha and beta), while have all the rest modes coefficients set to zero. After we find the range for the coefficients of the first modes, we proceed to the rest of the modes to perform the fine tuning (finding the best of the best designs). We believe that such an iterative approach might lead to a good results. In addition, there is another parameter that we must introduce, profile twist along the extrusion length L, because it has a big effect on the aerodynamic parameters.

Besides looking at the cruise case, we can also look at the more complicated take off case. We would start with the 2D optimization that involves optimization based on two computations. When we run the optimization for the best profile shape, we also have to take into the account that for each profile there is some most efficient angle of attack. Therefore we would have two optimization loops, where inside the big profile shape optimization, for each airfoil we have to find the most efficient angle of attack (best CL to CD ration) as well as. After finding the nest profile, we could proceed to the 3D take off case.

Finally, any 3D wing design must have things to take into account from industrial point of view. For example, there is a minimum thickness that a wing can have. Also, we are looking at the part of the wing, while a real wing has effects from wingtip device, fuselage and the wing sweep angle. Therefore, further models must take into accounts these real life appliance cases.