The handle http://hdl.handle.net/1887/22875 holds various files of this Leiden University dissertation

Author: Reehuis, Edgar
Title: Guiding evolutionary search towards innovative solutions
Issue Date: 2013-12-17
Application to Airfoil Optimization

In design optimization of airfoils, one is typically faced with high-dimensional search spaces that are constrained by geometric and manufacturing requirements. In certain problem instances, the simulator used for modeling the aerodynamic properties of candidate geometries will only return trustworthy results if the geometry under consideration is located well in feasible space, that is, not violating any such constraint. The search thus needs to be started from a basic solution known to be feasible. Furthermore, the mesh grid, i.e., the discretization of the space around a geometry, used in simulating, will be distributed according to this basic geometry. Therefore, at each stage of the search, the candidate solutions have to resemble the basic shape to a critical extent for the simulator to produce meaningful results, or produce results at all.

When trying to find innovative solutions in an automated way, we are hindered by the fact that the search needs to be started from basic feasible solutions. Though applying a stochastic optimization method, multiple instances of the search may converge to largely-similar optimized solutions. In order to uncover innovative solutions, we have to escape the basin of attraction of the solution that optimization typically converges to when started from a certain basic solution. Moving sufficiently far away, guided by some exploration scheme, has to be done stepwise, as we need to check that we are still in or close enough to feasible space to be able to start quality-based optimization again.

In Chapter 3, we motivated that a tangible approach for finding innovative solutions
is through aiming for a diverse set of high-quality solutions. With increased diversity and quality in a set of solutions, the potential of innovative solutions being present amongst these increases as well, ultimately to be determined by a human domain-expert. Diversity is determined with respect to a domain-specific distance measure. For the considered two-dimensional aerodynamic optimization of flow bodies, the difference between solutions will not be expressed by looking at the solution vectors in the search space (the genotypes), but by comparing other solution representations like the decoded two-dimensional shapes of the designs (the phenotypes). Note that instead of solely with these airfoil geometries, patents are typically concerned with the flow phenomenon that a certain geometry gives rise to, which is thus another possible source for determining difference between solutions.

This chapter deals with applying the restart scheme that was laid out in Chapter 5, alternating between phases of quality-optimization and exploration, to a real-world airfoil-optimization task, using the three tested exploration variants: Distance-based novelty measure uniqueness (Un), interestingness measure reducible error (RE\textsubscript{mDP}), based on memories of dispersion-in-prediction novelty, and the baseline method of exploring in a random straight line. First, the domain-specific distance measure is defined in Section 6.1, which will be used in diversity calculations as well as in distance-based novelty measure Un, after which the airfoil optimization test case is described in Section 6.2. Section 6.3 then lists the settings used for the alternating-restart-scheme variants in the experiments, of which the results are described in Section 6.4. A summary of the chapter is given in Section 6.5.

6.1 · Domain-specific Distance Measure

With respect to the domain of two-dimensional aerodynamic optimization of flow bodies, a distance measure is to be defined that will be used in scoring the diversity of sets of solutions after optimization, as well as for determining distance-based novelty of solutions during optimization. The two dimensions refer to the airfoil representation used in simulating aerodynamic behavior, the actual solution vectors from which these two-dimensional representations are derived are of (much) higher dimensionality. One aspect of the domain-specific distance measure is concisely expressing the diversity of sets of solutions, another is assisting the distance-based Un scheme in moving through the search space with reasonable computational cost. To come to an appropriate distance measure, we have to choose a solution representation first, and then a way
6.1 · Domain-specific Distance Measure

of calculating distance between two instances of that representation.

To lay out the different possible sources for determining the distance between solutions in the considered problem class, we sketch the path from solution vector to quality value. For a solution vector that represents a certain airfoil geometry, the aerodynamic behavior of that geometry is approximated by computational fluid dynamics (CFD) simulation. The simulated flow field around the geometry is numerically abstracted into a quality value, schematically

\[
\begin{align*}
\text{genotype } & \xrightarrow{\text{decoding}} \text{phenotype } \\
& \quad \xrightarrow{\text{simulation}} \text{behavior} \\
& \quad \xrightarrow{\text{analysis}} \text{properties} \\
& \quad \xrightarrow{\text{aggregation}} \text{quality}
\end{align*}
\]

The \textit{genotype} \( \textbf{x} \), consisting of \( n \) input parameters \( x_i \), represents an airfoil geometry that is obtained through interpreting the input parameters as \textit{non-uniform rational B-spline} (NURBS) \textit{control points}, resulting in the \textit{phenotype} of the solution, denoted \( \text{P}(\textbf{x}) \). CFD simulation is used to generate a flow field visualization around the phenotype, termed the \textit{behavior} of \( \textbf{x} \), denoted \( \text{B}(\textbf{x}) \). From this flow field approximation, scores for pre-defined properties \( \text{prop}_j(\text{B}(\textbf{x})) \) are determined, which are then aggregated into a single quality value according to some heuristic weight vector \( \textbf{w} \),

\[
f(\textbf{x}) = \sum_j w_j \text{prop}_j(\text{B}(\textbf{x})).
\]  

(6.2)

From the different available characterizations, we determine distance based on the two-dimensional phenotypes, as these allow for an effective but relatively simple domain-specific distance measure. The two phenotypes are sampled equidistantly along the spline, and each sample is again divided into two sets of points, one set representing the \textit{suction} side (side featuring higher air speed, upper side in Figure 6.1(b)) and one set representing the \textit{pressure} side (side featuring lower air speed, lower side in Figure 6.1(b)) per airfoil. The \textit{summed-up Hausdorff distance} [Olhofer et al., 2002] is then applied on the suction sets and the pressure sets. It is defined as

\[
d_{\Sigma H}(A, B) = \sum_{a \in A} \min_{b \in B} ||a - b|| + \sum_{b \in B} \min_{a \in A} ||a - b||,
\]  

(6.3)

that is, for each two-dimensional point \( \textbf{a} = (a_1, a_2) \in A \), calculate the distance to the
closest point \( b \in B \), and vice versa, and sum all these distances. To determine the distance between two solutions \( x_1 \) and \( x_2 \), we add the summed-up Hausdorff distance between the suction side samples and the distance between the pressure side samples,

\[
d_{\text{airfoil}}(x_1, x_2) = d_{\Sigma H}(\text{suction}(P(x_1)), \text{suction}(P(x_2))) + d_{\Sigma H}(\text{pressure}(P(x_1)), \text{pressure}(P(x_2)))
\]  

(6.4)

In this formulation of domain-specific distance, the fact that certain differences are generally weighted as more important by domain experts is omitted, for instance characteristics of the trailing edge, see Figure 6.1, that can induce important features in the resulting flow field. Structural weighting could be included in the way that domain-specific distance is calculated, although it is likely to prove difficult to comprehensively account for all aspects that a human engineer might consider.

The most-characterizing representation is the aerodynamic behavior that a geometry gives rise to, containing relevant information available through simulation, reflected in the fact that quality is derived from it as well. While more involved, distance could be calculated between Mach number distributions, see Figure 6.1, for instance by transforming an airfoil geometry into a comparison geometry (morphing), and having the surrounding flow field representation morph with it. Pixel-wise, difference can then be determined between the resulting morphed image and the comparison image. Alternatively, distance is determined by comparing Mach-number-distribution profiles along the airfoils, much like the way distance is determined between the phenotypes by using summed-up Hausdorff distance. At the end of this chapter, examples of Mach-number-distribution profiles are shown (see Figure 6.9, top row). In using this abstraction, however, we might not capture certain relevant features that are taken into account in calculating distance over the full Mach number distribution.

6.2 · Test Case: Two-dimensional Stator-blade Optimization

Optimization is performed on the outlet guide vanes (OGVs) of the low-pressure compressor in a small turbofan engine that is intended for propelling small business jet aircraft. Two-dimensional airfoil representations are used, involving 16 control points and thus 32 decision variables, see Figure 6.1. OGVs are stator blades, non-moving blades intended for straightening the air flow, arranged around a central axis in a cylindrical fashion. The compressor, displayed as a holistic unit in Figure 6.2, is in this case divided in a low-pressure compressor followed by a high-pressure compressor.
6.2 · Test Case: Two-dimensional Stator-blade Optimization

Figure 6.1 Initial Geometry. In (a), the 16 control points and their non-uniform rational B-spline (NURBS) decoded geometry is shown. In (b), the Mach number distribution (the relative air speed divided by the local speed of sound) of the simulated flow field around the geometry is displayed, showing a deviating outflow of $-5.96^\circ$ for the initial geometry. The control points can move on the horizontal and vertical axis, in all directions, as before simulation the geometry is translated to have its leading edge (left-most point) in (0,0), and is normalized by the chord length (distance between the leading edge and the trailing edge, right-most point).

After each compressor stage, stator blades are included to straighten the outflow.

The test case is a problem instance in which optimization needs to be started from a basic initial solution, displayed in Figure 6.1, as the search space features large infeasible regions containing solutions for which the quality cannot be determined through simulation. Simulating aerodynamic behavior is done using an in-house, quasi-three-dimensional Navier–Stokes flow solver, Honda Software for Turbomachinery Aerodynamics Research (HSTAR) [Arima et al., 1999]. Navier–Stokes equations are partial differential equations, involving multivariate functions, that express the motion of fluids. HSTAR is a deterministic simulator, hence simulation is not prone to noise, i.e., varying outcome of repeated simulation of a single solution [Kruisselbrink, 2012]. However, in approaching infeasible space, non-stationary uncertainty, i.e., locally-increased irregularity in quality values [Kruisselbrink, 2012], will be observed as the simulator has increased difficulty in reliably approximating the flow field around the candidate geometries.

6.2.1 · Quality Function

The fitness of a solution is based on seven properties that are aggregated into a to-be-minimized quality value, namely

- the resulting deviation angle of the air flow, that is, the outflow angle $\alpha_2$ minus the target outflow of $0^\circ$ (minus tolerance of $0.3^\circ$),

- the coefficient $\omega$ of the occurring pressure loss,
Three Dimensional Evolutionary Aerodynamic Design
Optimization with CMA-ES
Martina Hasenjäger
Honda Research Institute Europe GmbH

The task of the turbine is to convert gas energy into mechanical power in form of PC clusters made it possible to treat complex, computationally intensive application problems that hitherto were only tested on toy problems.

In this paper, we present an approach to 3D aerodynamic design optimization using evolutionary strategies. We show that, despite of its difficulty and high computational costs, our approach not only successfully optimizes the aerodynamic design but also yields interesting results from an engineering point of view.

The resulting hot, high-energy gases go into the turbine (d), causing the turbine blades to rotate, and the turbine to drive the compressor. The nozzle (e) is the exhaust duct of the engine. Image and description courtesy of [Hasenjäger et al., 2005].

- the extent of violation of four minimal-thickness constraints, i.e., a listed thickness has to be larger than its respective minimal thickness value, all values expressed as percentage of the blade’s chord length \( c_l \), see Figure 6.1,

  - leading edge radius \( r_{le} \), larger than 0.5\% \( \times c_l \),
  - trailing edge radius \( r_{te} \), larger than 0.5\% \( \times c_l \),
  - maximum thickness \( \theta_{max} \) of the blade, orthogonal to the outflow angle, larger than 5\% \( \times c_l \),
  - minimum thickness \( \theta_{min} \) of the blade, orthogonal to the outflow angle, larger than 0.5\% \( \times c_l \),

- and a penalty depending on the extent to which the simulator has not converged in solving the system of flow equations; \( \varepsilon_{solver} \) is the observed residual value, and the maximally allowed residual is \( 10^{-4} \).

The to-be-minimized quality function is defined as

\[
 f(x) = w_{a_2} \cdot g \left( |a_2(x) - 0^\circ| - 0.3^\circ \right)^2 + \\
 w_{\omega} \cdot \omega(x)^{1.5} + \\
 w_{\text{th}} \cdot \left( g \left( (0.5\% \times c_l - r_{le}(x))^2 + g(0.5\% \times c_l - r_{te}(x))^2 \right) + \\
 g(5\% \times c_l - \theta_{max}(x))^2 + g(0.5\% \times c_l - \theta_{min}(x))^2 \right) + \\
 w_{\varepsilon} \cdot \left[ g \left( \varepsilon_{solver}(x) - 10^{-4} \right) \right] \cdot \varepsilon_{solver}(x) / 10^{-4},
\]

(6.5)

Figure 6.2 Sketch of a Gas Turbine. The fan (a) pulls air into the engine. Part of this air is compressed in the compressor (b) and then forced into the combustion chamber (c), where it is mixed with fuel and ignited. The resulting hot, high-energy gases go into the turbine (d), causing the turbine blades to rotate, and the turbine to drive the compressor. The nozzle (e) is the exhaust duct of the engine. Image and description courtesy of [Hasenjäger et al., 2005].
with
\[ w_{\alpha_2} = 1, \ w_{\omega} = 20, \ w_{th} = 5 \times 10^6, \ w_{\varepsilon} = 10^5, \]
and
\[ g(a) = \max(0, a). \]  

### 6.3 · Modus Operandi

The alternating restart scheme, defined in Section 5.3.1, is run on top of a \((12,24)\)-CMA-ES for 5000 generations (120k quality evaluations by the simulator). For search dimension 32, see Section 6.2, the minimally prescribed population sizes for the CMA-ES are \((7,14)\), see Chapter 2. Larger population sizes where chosen to make the search less locally-oriented, in both quality optimization and exploration. The alternating restart scheme is compared against 5 standard optimization runs using the same \((12,24)\)-CMA-ES, running for 1000 generations each (24k evaluations). A problem-specific initial stepsizes \(\sigma_{\text{init}}\) of \(1 \times 10^{-3}\) is used.

Within the alternating restart scheme, we stop the quality-optimization phase when the stepsize is smaller than \(0.5 \times \sigma_{\text{init}}\). The exploration phase is also stopped when the stepsize is smaller than \(0.5 \times \sigma_{\text{init}}\), to account for possible stagnation, and is stopped when infeasible space is entered, considered such when the best-ranked solution with respect to the exploration measure

- is in a maximally erratic area for still allowing quality-based search to find a way back to a high-performing solution, i.e.,
  1. exceeds problem-specific hard-limit quality value \(1 \times 10^6\);
  2. exceeds problem-specific soft-limit quality value \(1 \times 10^3\), for 5 consecutive generations;

- is strictly infeasible, in which case a penalty is also added to the quality value, i.e.,
  3. the simulator does not converge for it: A variable penalty depending on simulator non-convergence is included, see Equation 6.5;
  4. contains a loop, that is, the lines defining the suction side and pressure side cross, i.e., upper and lower side in Figure 6.1(b): The solution gets the death penalty, i.e., its quality value is set to \(\gg 1 \times 10^6\);
5. the leading edge is not the lowest point of the blade, see Figure 6.1: The solution gets the death penalty.

The penalty values are included in quality scores of strictly infeasible solutions to assist the search in subsequently leaving infeasible space, in the quality-optimization phase that follows.

In the exploration phases, for escaping the basin of attraction of the lastly converged-to optimum, deviant geometries will be generated for which we will not get an accurate result from the simulator. As the mesh grid used in simulating is prepared for shapes with “standard” characteristics, more resemblant of the basic initial solution, the mesh might not be dense enough around certain areas of the shapes evolved in exploration to get a good outcome. The accuracy of the simulated aerodynamic behavior, however, is not a primary consideration, as long as, roughly speaking, on the path back to feasible space, solutions get increasingly better quality values. This is either through the intrinsic quality values that, despite their initial inaccuracy, should steadily decrease towards feasible space, or through the levels of non-convergence that also decrease in moving towards shapes sufficiently similar to the basic initial solution.

The conditions that lead to the death penalty, on the other hand, are treated differently, as these solutions should be strictly avoided: A loop or the leading edge no longer being the lowest point in a geometry is difficult to eliminate again through optimizing on quality. For this reason, in exploring, all offspring individuals are evaluated to check their feasibility, not only the selected parents, and therefore, generations in the exploration and quality-optimization phases use the same amount of quality evaluations. Moreover, because these forms of infeasibility are difficult to optimize out and because of the resulting costly exploration phases with respect to quality evaluations, we do not try to traverse infeasible space but instead halt exploration when it is entered.

Three variants of exploration are applied with the alternating restart scheme: Distance-based novelty measure uniqueness (Un), interestingness measure reducible error (RE_{mDP}), based on memories of dispersion-in-prediction novelty, and a baseline method of exploring in a random straight line (line explore). The exploration steps in line exploring, using a multivariate normal distribution with stepsize 0.01 (10 times \( \sigma_{\text{init}} \)), are not counted as a generation, as they involve evaluating only a single solution
on quality. Line explore therefore runs for (slightly) less than 5000 generations.

In calculating novelty as DP for $\text{RE}_{\text{mDP}}$, as described in Chapter 4, local model stacks consisting of 10 feed-forward neural networks (FFNs, see Chapter 2) are used. In each generation, a new stack is trained on the last $\gamma = 3$ generations of evaluated solutions, using maximally 72 training points per local model stack (cf. the last 5 generations in Chapter 4 and 5, maximally 30 and 60 training points respectively). An ensemble of disjoined local stacks is maintained by keeping the local model stack of each third generation, indicated by $\gamma = 3$. Sample points that are infeasible, either through exceeding quality limits or violating one of the strict feasibility conditions, are included in training, as these just have a bad intrinsic or penalized quality value.

The FFNs have linear outputs and a hidden layer of 25 sigmoidally-activated nodes, a bias node, and are trained using improved Rprop [Igel and Hüskens, 2003] for a maximum of 10k epochs from random initial connection weights in $[-0.1, 0.1]$. They are fully connected, including direct connections from input to output nodes. All data is used for training, no validation set is used. The CMA-ES and learning-guided restart scheme have been implemented using the Shark Machine Learning Library v2.3.43.

### 6.4 · Results

For each alternating-restart-scheme variant, 20 runs were performed on the task of two-dimensional stator-blade optimization. Sets of 5 standard-optimization runs are used as comparison, which can be seen as a single run of 120k evaluations that was restarted four times, performing 20 of such sets of 5 runs, thus 100 standard-optimization runs in total. Primary performance criterion is the surface dominated in quality/diversity space, laid out in Section 3.2.1, accounting for the diversity in a result set of solutions as well as the quality of these solutions. Measuring this surface leads to a singular hypervolume score that we can use to perform statistical comparison of the four methods: Un, $\text{RE}_{\text{mDP}}$, line explore, and 5 standard-optimization runs.

Per method, we select a run with median hypervolume score with respect to all runs of that method and plot the surface dominated by the solutions in its result set, in Figure 6.3. Only solutions with quality value and associated diversity score strictly better than those of the reference point of $(0.11, 0)$, used in the hypervolume calculations, are considered. The quality values are to be minimized: Beyond this quality value of 0.11, solutions are regarded not to be of high-quality. Remarkably,

---

Figure 6.3 **Median-run Performance Comparison.** Per method, one run with median hypervolume score is selected. Of each median run, the quality threshold and diversity score is plotted for each possible level set in its result set. Interpreting these pairs jointly as a Pareto-front approximation, we visualize the dominated surface in quality/diversity space. Based on a reference point of (0.11,0), standard optimization has an obtained hypervolume of 0.124 (result set of 5 solutions), uniqueness of 0.139 (result set of 4 solutions), reducible error of 0.329 (result set of 8 solutions), and line explore of 0.232 (result set of 9 solutions).

RE$_{mDP}$, showing worst hypervolume scores on AckleyInfeasible, see Section 5.3.1, is the best performing method on the real-world airfoil-optimization task.

When we further analyze the hypervolume scores, see Figure 6.4, it becomes clear that, like on AckleyInfeasible, line explore does well because of the large result sets it produces. Accounting for this, line explore gets the worst of the hypervolume-per-solution scores. Most notably, RE$_{mDP}$ has, next to the best median hypervolume score, the best median hypervolume-per-solution score, and additionally, it finds the best-quality solutions. Un follows closely in hypervolume-per-solution scores, but through producing smaller result sets, falls behind in hypervolume scores. Nevertheless, next to their good results, both RE$_{mDP}$ and Un include very weak runs with respect to the number of solutions found and hypervolume per solution. In these runs, the search ends up in an area of the search space in which no high-quality solutions are found, repeatedly converging to solutions with quality values worse than 0.11, which are therefore omitted in the results.

To explain the difference in performance between RE$_{mDP}$ and Un, we analyze the
Figure 6.4 Statistical Comparison. Per method, of 20 runs performed, the hypervolume score of each result set in quality/diversity space is calculated, displayed in (a). The hypervolume score is influenced by the number of solutions per result set, displayed in (c). Hypervolume score divided by the number of solutions in the result set is therefore displayed in (b). Lastly, another performance aspect considered is the to-be-minimized quality value of the best solution found per run, shown in (d). REMDP has the best median hypervolume score, the best hypervolume-per-solution scores, and finds the best-quality solutions.

alternating-restart-scheme variants in Figure 6.5. Averaged per run, Un needs many more steps (i.e., generations) per exploration phase to cover a similar distance in the decision space, i.e., Euclidean distance between solution vectors, as REMDP, that is, REMDP finds the boundary of infeasible space quicker. In contrast to the results on test function AckleyInfeasible, see Section 5.3.1, in the higher-dimensional search space of the stator-blade optimization, selecting on REMDP induces faster exploratory behavior with a larger stepsize for the CMA-ES than in selecting on Un. Next to greater hypervolume per solution, which was already shown on AckleyInfeasible, REMDP is now thus also able to produce more solutions than Un. The simpler line explore, with its isotropic search distribution, runs into infeasible space after only a small distance was covered in the decision space, deducible from the small number of steps taken (median average of 7) and the used fixed stepsize of 0.01. This leads to result sets containing lots of solutions, but close together, as is visible from Figure 6.4.

Moreover, we determine in what generation the best-performing solution per run is found, see Figure 6.5(c), to rule out that continued exploration is unbeneficial. This generation number turns out to be approximately normally-distributed, with a bias to
Figure 6.5 Analysis of Alternating-restart-scheme Variants. For the 20 runs performed per method, the average number of steps per exploration phase in a run is displayed in (a), and the average distance covered in the decision space per exploration phase in a run is displayed in (b), the latter not being available for line explore. Furthermore, the spread of the generation in which a run’s best-quality solution was found is shown in (c). Un and RE\textsubscript{mDP} cover a similar distance per exploration phase, but Un needs much more generations to do so. Furthermore, the generation in which the best solution is found is roughly normally-distributed.

Figure 6.6 Median-run Runtime Analysis. Of an Un and RE\textsubscript{mDP} run with median hypervolume score, the development of the running time per generation is plotted. Un shows faster quality-optimization phases and increasingly-slower exploration phases as the run continuous. RE\textsubscript{mDP} shows an equivalent running time for the quality-optimization and exploration phases, depending on the time needed to train the surrogate models used in calculating DP. In the RE\textsubscript{mDP} run, the start of a new phase is marked by a lower running time, because within a certain phase, only the solutions generated in that phase are used for training. Starting with a quality-optimization phase, it can be seen that the exploration phases in RE\textsubscript{mDP} are much shorter than in Un.
6.4 Results

(a) Standard Optimization (5)  
(b) Uniqueness

(c) Reducible Error – mDP  
(d) Line Explore

Figure 6.7 Median-run Found Geometries. Per method, three geometries found in a run with median hypervolume score are displayed. The variants of the alternating restart scheme first converge to a local optimum close to the geometries found by standard optimization, but then generate more-deviating geometries. Nevertheless, all Mach number distributions show high resemblance, seemingly as the left part of the suction side of all blades is similar. Quality values, from top to bottom and to be minimized, of solutions in (a), 0.0838, 0.0796, 0.0784, in (b), 0.0881, 0.0637, 0.0614, in (c), 0.0956, 0.0625, 0.0484, and in (d), 0.0803, 0.0761, 0.0681.

The time to complete one run per method is approximately, using 24 processor cores in parallel per run, 30 hours for a single standard-optimization run, two weeks for Un, three weeks for RE_mDP, and one week for line explore. Of a run with median hypervolume score for Un and RE_mDP, the runtime is further analyzed in Figure 6.6. For Un, the running time of generations in the quality-optimization phase...
Figure 6.8 Best Solution Found. Per method, the best geometry found is displayed. All best solutions found adhere to the thickness constraints: The alternating-restart-scheme variants therefore seem to require adjustment of these constraints, as their best geometries found appear too thin to be feasible in practice.

For comparing the actual geometries found by the different methods, of the runs with median hypervolume score, three solutions found are displayed in Figure 6.7 (on the previous page). After converging to the default optimum, the alternating-restart-scheme variants find more diverse and better-quality solutions outside of the default basin of attraction. Generally, the thinner the geometries get, the better their quality scores become. Notwithstanding, the distributions of the relative air speed around the airfoils are highly similar, as the variance in the shape of the left part of the suction sides (top side of the blades) is low.

In Figure 6.8, the best geometry found per method is displayed. Given the fact that all these geometries adhere to the minimal-thickness constraints, the way in
Table 6.1 Best Solution Found. Per method, the best solution found is listed. Used notation:
Quality value $f$, pressure loss coefficient $\omega$, outflow angle $\alpha_2$, leading edge radius $r_{le}$, trailing edge radius $r_{te}$, maximum thickness $\theta_{\text{max}}$, minimum thickness $\theta_{\text{min}}$, and chord length $cl$. The best values are in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>$f$</th>
<th>$\omega$</th>
<th>$\alpha_2$</th>
<th>$r_{le}$</th>
<th>$r_{te}$</th>
<th>$\theta_{\text{max}}$</th>
<th>$\theta_{\text{min}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Optimization</td>
<td>0.0679</td>
<td>2.26</td>
<td>0.299°</td>
<td>0.503</td>
<td>0.513</td>
<td><strong>5.00</strong></td>
<td>0.503</td>
</tr>
<tr>
<td>Uniqueness</td>
<td>0.0491</td>
<td>1.82</td>
<td>0.303°</td>
<td>0.501</td>
<td>0.501</td>
<td>5.01</td>
<td>0.501</td>
</tr>
<tr>
<td>Reducible Error – mDP</td>
<td><strong>0.0481</strong></td>
<td><strong>1.79</strong></td>
<td><strong>0.277°</strong></td>
<td><strong>0.500</strong></td>
<td><strong>0.500</strong></td>
<td>5.02</td>
<td><strong>0.500</strong></td>
</tr>
<tr>
<td>Line Explore</td>
<td>0.0595</td>
<td>2.07</td>
<td><strong>0.277°</strong></td>
<td>0.505</td>
<td>0.504</td>
<td>5.02</td>
<td>0.504</td>
</tr>
<tr>
<td>Initial Geometry</td>
<td>132</td>
<td>6.28</td>
<td>−5.96°</td>
<td>0.627</td>
<td>0.643</td>
<td>4.55</td>
<td>0.627</td>
</tr>
</tbody>
</table>

Figure 6.9 Comparison of Initial and Best Solution Found. The top-row plots show the Mach-number-distribution profile directly on the surface of the airfoils, and the bottom-row plots the Mach number distribution around the airfoils. First of all, the highest Mach-number-peak is lower for the best solution found, and located closer to the leading edge of the airfoil. Moreover, the best solution minimizes all minimal-thickness constraints to their lowest-allowed levels, see Table 6.1. Furthermore, while the outflow angle of the initial solution shows a deviation of −5.96°, in the best solution this is minimized to 0.277°, smaller than the tolerance of 0.3° and thus not degrading the quality score. Related to this outflow angle is the maximum blade thickness, which is determined orthogonally to it. For the best solution’s slightly down-pointing outflow, this, clearly undesirably, leads to a greater blade thickness of 5.02% × cl than the initial blade’s 4.55% × cl.
which the minimum and maximum thickness is determined requires adjustment, as the alternating-restart-scheme found geometries appear too thin to be realized in practice. This is further demonstrated in Table 6.1 and Figure 6.9, when comparing the best solution found to the initial solution from which the search is always started. The best solution, clearly with a smaller maximum thickness, gets a higher maximum-thickness score than the initial solution because of the way in which these are determined, orthogonally to the outflow angle.

6.5 · Summary

This chapter reports on the application of the most-promising exploration measures identified in Chapter 4, as integrated into quality-based search using an alternating restart scheme proposed in Chapter 5, to a real-world airfoil-optimization task. It concerns the optimization of a two-dimensional representation of a stator blade involving 16 control points, and hence 32 decision variables. Instances of the stator blade in question are used directly after the low-pressure compressor in a small turbofan engine, for straightening the flow of the compressed air.

After defining a domain-specific distance measure to be calculated over the two-dimensional blade geometries, three variants of exploration within the alternating restart scheme were tested on the real-world test case: Distance-based novelty measure uniqueness (Un), which utilizes the domain-specific distance measure, interestingness measure reducible error (RE_{mDP}), based on memories of dispersion-in-prediction novelty, and the baseline method of exploring in a straight line, for which the gradient is randomly chosen. The instantiations of the alternating restart scheme were compared with the combined outcome of multiple standard optimization runs.

For determining diversity in the result sets of the different methods, the domain-specific distance measure is again used. Utilizing the performance measuring scheme introduced in Chapter 3, appreciating both quality and diversity in a set of solutions, RE_{mDP}-based exploration was clearly shown to deliver the best-performing method. It outperforms the combined outcome of multiple standard-optimization runs, using the same number of quality evaluations, in the sense that it finds more solutions, more diverse solutions, and finds solutions of better quality! Importantly, it manages to do so in spite of consuming quality evaluations in its exploration phases, which reduces its budget of evaluations available for the actual quality optimization.

Exploration based on Un provides more diverse and higher-quality solutions than
standard optimization as well, but induces slower-moving exploration than RE$_{mDP}$ in the sense of distance covered in the decision space per generation, thereby resulting in less solutions found. The baseline method of line exploring finds lots of solutions but with poor diversity, because its rudimentary exploration approach only allows for small distances to be traveled before running into infeasible space.