Evolutionary Optimization of Ship Propulsion Systems

Boris Naujoks, Max Steden, Sven-Brian Müller, and Jochen Hundemer

Abstract—The concept of a linear jet yields the potential to play a decisive role in modern ship propulsion systems technology. A linear jet consists of multiple components, namely a rotor, a stator, and a nozzle. The optimization of a complete linear jet is a complex and challenging task, which requires a couple of comprehensive simulation tools to work together. All required analysis and simulation issues are subject to ongoing research. The aspired optimization algorithm is already available and can be tested on the different components that can already be analyzed. In this way, the optimization is first limited to the propeller blade to determine good parameterizations for the optimization method. The results are analyzed in detail performing deeper investigations with additional optimization runs. Furthermore, one parameterization identified to perform best results is incorporated for the optimization of a complete propulsion system featuring rotor, hub, and nozzle.

I. INTRODUCTION

We aim at a method for the automatic optimization of a linear jet. The linear jet is a ship propulsion system for a speed range between conventional propellers and water jets. Conventional propellers tend to cavitate at high ship speeds. This causes losses in efficiency and damages at the propeller. Water jets deliver high thrust at high speeds, but their efficiency is not competitive at lower speeds.

As a first step an optimization of a propeller blade has been carried out to get first information concerning the interaction of the optimization algorithm and the simulation tool. This is based on potential theory for the calculation of the hydrodynamic properties of propeller. The generation of varied propeller geometries, as well as their calculation and evaluation must take place automatically, therefore, at first, special attention is paid to this procedure. Furthermore, it is essential to find out if the optimization algorithm is able to generate solutions which show an aspired physical behavior. According to this, the first optimization is not based on a complex problem.

The main target of the optimization is the maximization of the total system efficiency at the investigated operation point. The operation point describes the total thrust at a certain ship velocity. Another important task is the avoidance of cavitation in the whole system. That is why we include the efficiency, the amount of cavitation and the total thrust in the target function, which is being minimized by the optimization program.

The optimization method in use is a meta-model assisted evolutionary algorithm (MAEA) according to Emmerich et al. [1] featuring Gaussian random field models to predict fitness function values. It yields local modeling and individual-based control according to the classification by Jin [2] and allows for passing the percentage of individuals to be evaluated exactly to the algorithm as a parameter.

In the following section, a short introduction to the optimization of ship propulsion systems is given. The simulation method is described and attention is paid to the different problems resulting from the decomposition of the linear jet into components to be optimized separately. More detailedly, the optimization of a stand alone propeller blade is described next to a linear jet engine featuring a rotor, a nozzle, and a hub.

The section thereafter describes the optimization method applied to the presented optimization task, namely the meta-model assisted evolutionary algorithm. Results received from this algorithm are presented and discussed in section IV. Section V summarizes the results and provides directions for future work.

II. OPTIMIZATION OF SHIP PROPULSION SYSTEMS

Generally speaking, there are two main possibilities to calculate the flow around propellers. One approach is to solve the Reynolds-Averaged-Navier-Stokes-Equations (RANSE) (cf. [3], [4]). This is usually done by finding an approximated solution for the set of differential equations at discretized points of space and time. Hence, a 3D mesh needs to be generated manually, which describes the liquid phase around the propeller. Such a simulation leads to very accurate results, but is very time consuming due to the mesh generation and the calculation times.

As friction can be almost neglected in the flow around a propeller, a potential based approach (cf. [5], [6], [7]) can be used to calculate the main performance values. Potential solvers are able to predict the performance of a propeller design with reasonable accuracy. This performance of a ship propeller is implemented as a function of its resulting efficiency, torque coefficient, thrust coefficients, and cavitation.

All these properties of the blade and the resulting flow are computed by potential flow numerical solvers and aggregated to a single-objective fitness function afterwards. The simulator, an implementation of a first-order potential-based panel method, calculates the hydrodynamic performance of a given propeller at a given advance coefficient. Furthermore, the area where pressure yields values below the vapor pressure is calculated to evaluate the magnitude of cavitation.

The flow is described as a velocity potential $\Phi$, which fulfills the Laplace equation:

$$\nabla^2 \Phi = 0.$$
So called sources and doublets, which are special solutions of the Laplace equation, are distributed on the propeller surface. The strength of those sources and doublets need to be determined so that no flow passes through the propeller surface and the normal boundary condition is fulfilled:

$$\nabla (\Phi + \Phi_\infty) \cdot \vec{n} = 0. \quad (2)$$

The incoming flow towards the propeller is described by the potential $\Phi_\infty = u_\infty x + v_\infty y + w_\infty z$, while the vector $\vec{n}$ is pointing outward normal to the propeller surface. The boundary conditions can be transformed into a set of linear equations. The solution gives the strength of each doublet ($\mu$), while the strength of the sources is $\sigma_i = -\vec{n} \cdot \vec{V}_\infty$. This leads to the following potential around the propeller:

$$\Phi(x) = \frac{1}{4\pi} \left( \sum_{i=1}^{N_k} \int_{A_i} \frac{1}{r} dS \right) + \sum_{j=1}^{N_k+N_w} \mu_j \int_{A_j} \vec{n}_j \nabla \frac{1}{r} dS + \Phi_\infty. \quad (3)$$

The potential based approach in use is a boundary element method with a quadrilateral source and doublet distribution on each panel. The panels are located on the surface of the propeller, which makes an automated grid generation possible.

The number of unknowns, which need to be solved, is about a few thousands in contrast to a few million which have to be solved during the RANSE calculation. Hence, the procedure is quick enough to include it into an optimization algorithm, though each propeller evaluation may last between a minute and up to several hours depending on the complexity of the investigated problem.

### A. Optimization of a propeller blade

For the optimization of the propeller blade as a stand alone component, the geometry of the propeller was defined by:

- the propeller’s diameter,
- the rate of revolution,
- the pitch, and
- the chord length (cf. Fig. 1).

The gradients are based on spline curves, whose bases are defined by the parameters. Moreover, a modification of the rotation speed of the propeller was permitted. The observation of the required thrust even with smaller area ratio (AE/AD) in connection with a constant diameter (D) is possible by variation of the pitch ratio (P/D), the cord length (C) and the rotation speed of the propeller (n). All together, nine parameters are changed by the optimization algorithm and provided to the simulation tool, i.e. the decision space features a dimension of nine.

The goal of this optimization was the maximization of the propeller efficiency ($\eta$) by an observation of the required thrust ($T_{req}$) with an allowed deviation of maximum 4%. This condition was included into the fitness function, because the diameter was not variable and thus a later thrust adaptation by selection of the necessary diameter was excluded.

The optimization of the propeller efficiency with a required thrust inevitably leads to slender propeller blades in combination with high pitch ratios. This has negative effects on the cavitation behavior (under-usage of the vapour pressure), because the loading of the propeller blade increases and the pressure on the suction side sinks. Due to this fact, a further condition was formulated. This includes an area, where the pressure at the surface is below the vapour ($A_{cav}$), which has to be less than by 1,5% of the total area of the propeller ($A_{tot}$) without having an influence onto the fitness function. From these postulations the following fitness function $f$, which must be minimized in the optimization, was formulated:

$$f(.) = -\eta + a \cdot \left( \max(1 - \frac{T}{T_{req}} - 0, 04), 0 \right)^2$$

and

$$\eta = \frac{T \cdot \nu}{Q \cdot 2 \cdot \pi \cdot n}.$$

Weighting factors $a$ and $b$ for both penalties are kept constant throughout the whole investigation.

![Fig. 1. Visualization of some decision parameters for the propeller blade optimization problem. Because only the propeller blade is optimized, nozzle and hub of the more complex linear jet are omitted.](image)

### B. Optimization of a linear jet

A linear jet normally consists of four components: Rotor, hub and nozzle, as to be seen in Fig. 2 and 3, as well as a stator behind the rotor. The development of such a system is quite complex as these components have an influence one onto each other. Changing any geometry or operational parameter without adapting the others, can have a significant influence on efficiency or cavitation.
Additional variable geometry parameters for the linear jet in contrast to the more simple propeller blade optimization are:

- the hub’s diameter and length
- the nozzle’s length and profile angle.

This results in an optimization problem featuring 14 decision parameters in contrast to the nine decision parameters of the pure propeller blade optimization problem described above. The stator is not included into the optimization yet.

Again, like in the task presented before, a geometry to deliver more or less thrust compared to the desired value is “punished” with a higher target value. By a similar “punishment” of a geometry generating cavitation and a “reward” for efficiency we calculate a single target value which is returned to the optimization program after every hydrodynamic simulation.

III. META-MODEL-ASSISTED EVOLUTIONARY OPTIMIZATION

The idea to assist direct search algorithms by meta-models has first been explored by Torczon et al. [8], [9] for pattern search algorithms. A similar approach can be employed in evolutionary algorithms (EA) by incorporating a pre-screening procedure before the offspring population is evaluated with the time consuming evaluation tool. Algorithm 1 gives an outline of MAEA which is, in fact, a modified version of the basic \((\mu+\lambda)\)-EA described by Bäck, Hammel, and Schwefel [10] or Beyer and Schwefel [11]. Two features distinguish MAEA from standard EA.

1) All exactly evaluated individuals are recorded and stored in a database. Up to 15 nearest neighbors are considered to set up the meta-model for each of the \(\lambda\) individuals per generation.

2) During the pre-screening phase, the objective function values for new solutions are predicted by the meta-model, before deciding whether they need to be re-evaluated by the exact and costly tool.

Thereby, at generation \(t\), the set of offspring solutions \(G_t\) is reduced to the subset of offspring solutions \(Q_t\), which will be evaluated exactly and will also be considered in the final selection procedure (cf. [1]).

Algorithm 1 \((\mu+\nu<\lambda)\)-MAEA

\[
\begin{align*}
\text{Algorithm } & \text{1 (} \mu+\nu<\lambda \text{)-MAEA} \\
\text{1) } & t \leftarrow 0 \\
\text{2) } & P_t \leftarrow \text{init()} \quad \# \text{ } P_t: \text{ Set of solutions } \\
\text{3) } & \text{initialize database } D \\
\text{4) } & \text{while } t < t_{\text{MAX}} \text{ do} \\
& \quad G_t \leftarrow \text{generate}(P_t) \quad \# \text{ } \lambda \text{ new offsprings } \\
& \quad \text{evaluate } G_t \text{ with meta-model} \\
& \quad Q_t \leftarrow \text{select}(G_t) \quad \# \text{ } |Q_t| = \nu \quad \# \\
& \quad \text{evaluate } Q_t \text{ precisely} \\
& \quad \text{update database} \\
& \quad P_{t+1} \leftarrow \text{select}(Q_t \cup P_t) \quad \# \text{ Select } \mu \text{ best } \\
& \quad t \leftarrow t + 1 \\
\text{5) } & \text{end while}
\end{align*}
\]

A. Pre-screening procedures

A ranking algorithm, applied over the offspring population \(G_t\), identifies the most promising individuals in the new generation. In the general case, this algorithm is based on the values \(\hat{y}(x)\) (predictions for \(f(x)\)) and \(\hat{s}(x)\) (corresponding standard deviations) obtained for each individual \(x \in G_t\) through the meta-model. Comparisons with objective function values for the parent population \(P_t\) are necessary. Various criteria for identifying promising solutions are discussed by Emmerich et al. [1]. Once the promising subset \(Q_t\) of \(G_t\) has been found, its members undergo exact evaluations.
The simplest way to pre-screen new solutions is by ranking them through the most likely improvement criterion. This criterion makes use of the predicted mean function value without considering the confidence information $\hat{s}(x)$. Stated differently, this criterion reflects the improvement achieved when the response value with maximum probability density function is considered. Conceptually, this criterion is similar to any pre-screening which is based on any kind of neural networks.

An important issue is to benefit from the uncertainty information provided by the meta-model. For minimization problems, Torczon et al. [8], [12] suggested the use of the lower confidence bound (LCB) of a prediction

$$f_{lb}(x) = \hat{y}(x) - \omega \hat{s}(x), \quad \omega \in [0, 3]$$  \hspace{1cm} (4)

instead of the predicted value itself. The idea is to increase the number of evaluations in promising but less explored regions of the search space by directing the search towards them. Emmerich et al. [13] demonstrated that $f_{lb}$ is also a good criterion in the context of MAEA. In particular, in multi-modal optimization problems the optimization results improved significantly.

By choosing $\omega$, the user can scale the MAEA from fast local search to more explorative global search [12]. However, the choice of an extra parameter might also be seen as a burden to the user. A reasonable choice is $\omega = 2$, which leads to a high confidence probability (ca. 97 %) for $f_{lb}(x)$ to be the lower confidence bound of $f(x)$, once the GRFM assumptions are valid. Emmerich et al. showed that a fast local convergence on smooth problems is still possible [13].

### B. Variation operators

For variation, standard operators for continuous optimization problems from EA literature [10], [11] are utilized. For recombination, two parents were chosen randomly, whenever the number of parent $\mu$ was greater than two. For decision space parameters as well as for step sizes, intermediate recombination was involved. This choice on the decision space parameters was due to the setting of interval ranges for each parameter. These were chosen large enough to expect optimal values to lay really inside the interval, not on the borders. But of course, interactions of parameter settings cannot be foreseen and are not considered.

For mutation, one step size for each parameter was invoked. The different parameters involved in the optimization yield completely different interval ranges like $[0.71, 1.64]$ or $[495.00, 815.00]$. Due to not normalizing these different ranges, using only one step size seemed inadequate. For each new individual per generation and each parameter $j$, mutation was performed according to

$$\sigma_{j}^{new} = \sigma_{j}^{old}, \tau \cdot N(0,1),$$

$$x_{j}^{new} = x_{j}^{old} + \sigma_{j}^{new}, \tau \cdot N(0,1),$$

with $N(0,1)$ being the implementation of a standard normal distribution and $\tau \approx 1.543$. Note that only a maximum of 1000 exact fitness function evaluations were allowed per run.

### IV. RESULTS

The report on the results received is executed according to an advice by Preuß [14]. He suggests to subdivide the report into seven parts, namely research question, preexperimental planning, task, setup, results/visualization, observations and discussion. This division origins from the framework of sequential parameter optimization by Bartz-Beielstein [15]. It demands the author to think about the questions at hand in more detail and to report the results more carefully.

#### A. Optimization of a propeller blade

**Research question**

The first task investigated is the search for good parameter settings for the optimization of the propeller blade.

**Preexperimental planning**

Only a few optimization runs have been performed before with some modified source code. These optimization runs lead to the identification of problems on step size adaptation. This is, why major parts of the optimization software were changed afterwards. The results received with the problematic source code hint to larger populations sizes being superior to smaller ones. Due to the problems with the software, this fact was not studied with the former software in more detail.

Within the mentioned runs, reliable results have been received concerning the run-time of the simulation software. It turned out that one simulation run, i.e. linking the surface grip, calculation of the potential solver etc, i.e. one fitness function calculation for the EA, takes about half a minute for the propeller blade problem. This results in an approximate run-time of about eight hours for the 1000 fitness function evaluations allowed. The run-time of the EA was neglected.

**Task**

An explorative study is to be performed to determine good and robust parameter settings for the propeller blade design problem. It is not expected that parameter settings are determined exactly but the following questions should be answered:

1. Are larger or smaller population sizes beneficial to address the problem at hand, i.e. are larger gene pools advantageous or are smaller population sizes with a more restricted gene pool better?
2. How does the selection pressure influence the results?
   - Does a small selection pressure $\frac{\mu + v}{\mu}$ perform better than a larger one $\frac{\mu + v}{\mu} \rightarrow \infty$?
3. Are significant improvements noticeable due to the incorporation of the meta-model? Does it pay to involve the complex approximation of fitness function values and what about the corresponding parameters of the meta-model assistance?

**Setup**

Due to the large calculation times it seemed unrealistic to set up the whole framework of sequential parameter optimization according to Bartz-Beielsten et al. [15], [16]. It seemed more adequate to test certain parameter settings known from the literature to reduce the number of necessary optimization
runs to receive first results. According to these assumptions, optimization runs were executed with different population sizes of $\mu \in \{1, 2, 3, 5, 10\}$, different fractions of individuals evaluated exactly per generation $c = \frac{\nu}{\lambda} \in \{0.2, 1.0\}$ and different selection pressures $\frac{\mu + \nu}{\mu}$.

All other parameters were kept constant for the different runs performed. This means the LCB-criterion was involved for the prediction of fitness function values with a constant confidence value of $\omega = 2$. In the beginning of the optimization run, twice the decision space dimension minus the number of parents exact fitness function evaluations were performed to fill the data base for the approximation.

**Results/Visualization**

Tab. I presents the average best fitness function value received from five runs of each parameterization next to the corresponding standard deviation. Furthermore, it was determined, when this result was received, i.e. all exact fitness function evaluations were counted and the number of the best evaluation returned. This way, the average over all five runs given in the second column of Tab. I could be determined, next to the corresponding standard deviations given in brackets.

Fig. 4 depicts the best fitness function values plotted against evaluation count for the two best parameterizations, i.e. the $(1 + 1)$-EA on the left hand side and the $(1 + 1 < 5)$-MAEA on the right hand side. For both parameterizations, all five runs are shown in the graphics.

**Table I**

<table>
<thead>
<tr>
<th>Selection</th>
<th>Result (Std.-Dev.)</th>
<th>Evaluation (Std.-Dev.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(2 + 10 &lt; 50)$</td>
<td>-0.79090 (0.002571)</td>
<td>638.8 (218.997)</td>
</tr>
<tr>
<td>$(5 + 10 &lt; 50)$</td>
<td>-0.79136 (0.002007)</td>
<td>677.2 (278.088)</td>
</tr>
<tr>
<td>$(3 + 4 &lt; 20)$</td>
<td>-0.79054 (0.003537)</td>
<td>667.6 (292.934)</td>
</tr>
<tr>
<td>$(10 + 10 &lt; 100)$</td>
<td>-0.78982 (0.003511)</td>
<td>875.5 (120.522)</td>
</tr>
<tr>
<td>$(1 + 1 &lt; 5)$</td>
<td>-0.79292 (0.002927)</td>
<td>459.2 (101.888)</td>
</tr>
<tr>
<td>$(1 + 1)$</td>
<td>-0.79310 (0.004217)</td>
<td>842.2 (133.719)</td>
</tr>
<tr>
<td>$(1, 5)$</td>
<td>-0.78870 (0.006352)</td>
<td>647.7 (130.715)</td>
</tr>
<tr>
<td>$(1 + 5)$</td>
<td>-0.78928 (0.005665)</td>
<td>642.6 (270.908)</td>
</tr>
<tr>
<td>$(2 + 10)$</td>
<td>-0.78946 (0.002159)</td>
<td>705.6 (212.890)</td>
</tr>
<tr>
<td>$(5 + 50)$</td>
<td>-0.78797 (0.002673)</td>
<td>344.3 (190.538)</td>
</tr>
</tbody>
</table>

**Observations**

Surprisingly, the parameterization yielding best fitness function values on average is a simple $(1 + 1)$-EA without meta-model assistance (0.7931). The second best result was received with a very similar strategy but incorporating the meta-model. This was the $(1 + 1 < 5)$-MAEA that preselects one solution from five alternatives preevaluated by the meta-model for a $(1 + 1)$ selection scheme.

The worst results have been received with large population sizes, a contradiction to the results from the first implementation, reported in the preexperimental planning part. The worst parameterization within the chosen setup was the $(5 + 50)$-EA with a mean fitness of -0.78797. Invoking the meta-model for a similar parameterization, turning it into a $(5 + 10 < 50)$-MAEA, yielded quite good results (mean fitness of -0.79136). Like it can be observed from Tab. I, all other parameterizations performed between the given values on average. A tendency toward better results for smaller population sized can be noticed. An exception is the comma selection strategy $(1, 5)$-EA performing comparably bad although featuring a small population size.

Turning to the evaluation count, when the best fitness function value has been received per optimization run on average, large differences can be detected. These differences range from a rather small value of 344.3 for the $(5 + 50)$-EA up to a $(10 + 20 < 100)$-MAEA yielding a comparably high value of 875.5. It should be pointed out again that these results are averaged values over five runs, not single outliers.

Interestingly, the second highest value has been received with the parameterization yielding best fitness function value on average, i.e. the $(1 + 1)$-EA. The second lowest value has been received with the second best performing parameterization considering fitness function values, i.e. the $(1 + 1 < 5)$-MAEA. Hence, the two best performing strategies behave rather differently. While one receives the best fitness function values quite early after less then half of the allowed fitness function evaluations ($(1 + 1 < 5)$-MAEA value: 459.2), the other parameterization improves nearly until the end of the optimization run ($(1 + 1)$-EA value: 842.2).

With respect to the standard deviations of the values just discussed, all mentioned strategies yield comparably low values. All standard deviations are smaller than a value of 300 while half of these values from Tab. I are between 200 and 300.

Comparing strategies invoking the preevaluation by the meta-model with their direct counterparts not using the meta-model information, it must be pointed out that the use of the meta-model is able to improve results. This is shown by comparing the strategies $(2 + 10)$-EA and $(2 + 10 < 50)$-MAEA, $(5 + 50)$-EA and $(5 + 10 < 50)$-MAEA, as well as $(1 + 5)$-EA, $(1 + 1 < 5)$-MAEA and $(1 + 1)$-EA. In all three cases, except for $(1 + 1 < 5)$-MAEA and $(1 + 1)$-EA, the meta-model assisted strategy performs better than the strategy without preselection. This still holds, whether the strategies are identical with respect to the exactly evaluated individuals (identical $\mu$ and $\nu$ like in the two parents case) or with respect to all generated offsprings (identical $\mu$ and $\lambda$ like in the five parents case).

A last observation addresses the difference between comma and plus selection schemes. Here only two parameterizations can be compared, but both perform comparably bad. As a result, no final conclusion can be drawn and further investigations seem to be necessary. Nevertheless, the best strategy is the $(1 + 1)$-EA, where no reasonable comma strategy can be deduced.

The behavior reported for the two best strategies can be reflected in the course of the best fitness function values...
received over the evaluation counter in Fig. 4. It can clearly be observed that all the runs of the (1 + 1)-EA (left graphic in Fig. 4) are able to generate improvements nearly until the end of the corresponding runs.

In the right part of the picture, the same is plotted for the (1 + 1 < 5)-MAEA. Except for one run, improvements stop before 500 fitness function evaluations have been reached. The single exception manages to generate better solutions up to about 600 fitness function evaluations and stagnates like all other runs thereafter. Moreover, the courses of the (1 + 1 < 5)-MAEA show that four of the five runs reach fitness function values less than −0.79 while there is one outlier reaching a value of −0.785 only.

**Discussion**

The meta-model assisted strategies were expected to outperform the other ones and this is strengthened by the comparison of similar parameterizations with and without meta-model assistance. Nevertheless, best results on average have been received with the most simple EA and without meta-model assistance.

For the (1 + 1)-EA (left graphic in Fig. 4) it is self-evident that the results can be further improved by allowing more that 1000 fitness function evaluations. For the (1 + 1 < 5)-MAEA one can assume that the optimization is ended after 500 to 600 fitness function evaluations and no further improvements are possible. Consequently, there is no need to spend more time on more that 600 fitness function evaluations for this parameterization.

It is concluded that good results can be received much faster with meta-model assistance than without. In detail, the (1+1 < 5)-MAEA receives almost comparable values within about half the time of a (1 + 1)-EA. But (1 + 1)-EA may get better, if more than 1000 fitness function evaluations are allowed.

**B. More detailed investigation of the propeller blade**

**Research question**

Do the results reported above hold, if more than five optimization runs are performed?

**Preexperimental planning**

All results received until now and reported within the previous section belong to the preexperimental planning. The task at had is directly derived from the observations and discussion provided before.

**Task**

More optimization runs were to be performed for the parameterizations identified within the analysis before, i.e. the (1 + 1)-EA and the (1 + 1 < 5)-MAEA. The results are to be reported giving the same numbers and plots like before. Due to the large calculation times, additional runs for other parameter settings have not been considered.

**Setup**

The setup was identical to the one described above, except for the seeds of the random number generator. Here, five different seeds were used resulting in five new optimization results summing up to a value of ten performed optimization runs.

**Results/Visualization**

Again, results are reported like in the section before. Tab. II gives the average of the best fitness function values received over all ten runs executed until now. In brackets, the corresponding standard deviations are given. The second column presents the average evaluation number, when the best fitness function value has been received, over all ten runs, next to the corresponding standard deviations again.

Additionally, Fig. 5 presents the course of the newly performed runs giving fitness function values vs. the corresponding evaluation count. Note, that the values within Tab. II are calculated considering all ten runs performed per parameterization, while Fig. 5 presents only the new results.

**Observations**

As can be seen from Fig. 5, most of the runs end up with best fitness function values between -0.8 and -0.79 again. In contrast to the courses presented before, (cf. Fig 4), next to an outlier with a best value greater than -0.79, for the (1 + 1 < 5)-MAEA another outlier with best fitness function values less that -0.81 is observed that all the runs of the (1 + 1)-EA (left graphic in Fig. 4) are able to generate improvements nearly until the end of the corresponding runs.

In the right part of the picture, the same is plotted for the (1 + 1 < 5)-MAEA. Except for one run, improvements stop before 500 fitness function evaluations have been reached. The single exception manages to generate better solutions up to about 600 fitness function evaluations and stagnates like all other runs thereafter. Moreover, the courses of the (1 + 1 < 5)-MAEA show that four of the five runs reach fitness function values less than −0.79 while there is one outlier reaching a value of −0.785 only.

**Discussion**

The meta-model assisted strategies were expected to outperform the other ones and this is strengthened by the comparison of similar parameterizations with and without meta-model assistance. Nevertheless, best results on average have been received with the most simple EA and without meta-model assistance.

For the (1 + 1)-EA (left graphic in Fig. 4) it is self-evident that the results can be further improved by allowing more that 1000 fitness function evaluations. For the (1 + 1 < 5)-MAEA one can assume that the optimization is ended after 500 to 600 fitness function evaluations and no further improvements are possible. Consequently, there is no need to spend more time on more that 600 fitness function evaluations for this parameterization.

It is concluded that good results can be received much faster with meta-model assistance than without. In detail, the (1+1 < 5)-MAEA receives almost comparable values within about half the time of a (1 + 1)-EA. But (1 + 1)-EA may get better, if more than 1000 fitness function evaluations are allowed.

**B. More detailed investigation of the propeller blade**

**Research question**

Do the results reported above hold, if more than five optimization runs are performed?

**Preexperimental planning**

All results received until now and reported within the previous section belong to the preexperimental planning. The task at had is directly derived from the observations and discussion provided before.

**Task**

More optimization runs were to be performed for the parameterizations identified within the analysis before, i.e. the (1 + 1)-EA and the (1 + 1 < 5)-MAEA. The results are to be reported giving the same numbers and plots like before. Due to the large calculation times, additional runs for other parameter settings have not been considered.

**Setup**

The setup was identical to the one described above, except for the seeds of the random number generator. Here, five different seeds were used resulting in five new optimization results summing up to a value of ten performed optimization runs.

**Results/Visualization**

Again, results are reported like in the section before. Tab. II gives the average of the best fitness function values received over all ten runs executed until now. In brackets, the corresponding standard deviations are given. The second column presents the average evaluation number, when the best fitness function value has been received, over all ten runs, next to the corresponding standard deviations again.

Additionally, Fig. 5 presents the course of the newly performed runs giving fitness function values vs. the corresponding evaluation count. Note, that the values within Tab. II are calculated considering all ten runs performed per parameterization, while Fig. 5 presents only the new results.

**Observations**

As can be seen from Fig. 5, most of the runs end up with best fitness function values between -0.8 and -0.79 again. In contrast to the courses presented before, (cf. Fig 4), next to an outlier with a best value greater than -0.79, for the (1 + 1 < 5)-MAEA another outlier with best fitness function values less that -0.81 is observed that all the runs of the (1 + 1)-EA (left graphic in Fig. 4) are able to generate improvements nearly until the end of the corresponding runs.

In the right part of the picture, the same is plotted for the (1 + 1 < 5)-MAEA. Except for one run, improvements stop before 500 fitness function evaluations have been reached. The single exception manages to generate better solutions up to about 600 fitness function evaluations and stagnates like all other runs thereafter. Moreover, the courses of the (1 + 1 < 5)-MAEA show that four of the five runs reach fitness function values less than −0.79 while there is one outlier reaching a value of −0.785 only.

**Discussion**

The meta-model assisted strategies were expected to outperform the other ones and this is strengthened by the comparison of similar parameterizations with and without meta-model assistance. Nevertheless, best results on average have been received with the most simple EA and without meta-model assistance.

For the (1 + 1)-EA (left graphic in Fig. 4) it is self-evident that the results can be further improved by allowing more that 1000 fitness function evaluations. For the (1 + 1 < 5)-MAEA one can assume that the optimization is ended after 500 to 600 fitness function evaluations and no further improvements are possible. Consequently, there is no need to spend more time on more that 600 fitness function evaluations for this parameterization.

It is concluded that good results can be received much faster with meta-model assistance than without. In detail, the (1+1 < 5)-MAEA receives almost comparable values within about half the time of a (1 + 1)-EA. But (1 + 1)-EA may get better, if more than 1000 fitness function evaluations are allowed.

**B. More detailed investigation of the propeller blade**

**Research question**

Do the results reported above hold, if more than five optimization runs are performed?
Fig. 5. Fitness vs. evaluations plots of five additional runs for two parameterizations (left: \((1+1)\)-EA, right: \((1+1<5)\)-MAEA) of the optimization methods employed for the addressed propeller blade task.

TABLE II

<table>
<thead>
<tr>
<th>Selection</th>
<th>Result (Std.-Dev.)</th>
<th>Evaluation (Std.-Dev.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>((1+1))</td>
<td>-0.79378 (0.004216)</td>
<td>797 (177.8)</td>
</tr>
<tr>
<td>((1+1&lt;5))</td>
<td>-0.79431 (0.005972)</td>
<td>443.6 (238.3)</td>
</tr>
</tbody>
</table>

value smaller than -0.8 is present now.

Moreover, the average of the values generated with the \((1+1)\)-EA approach seem to be greater than the one observed for the \((1+1<5)\)-MAEA. Consequently, the calculated average for the new five runs is expected to be smaller for the \((1+1<5)\)-MAEA than for the \((1+1)\)-EA. As an effect, the values for all ten runs change in ranking, i.e. the average over all ten runs performed with the \((1+1<5)\)-MAEA turned to be better than the one for the \((1+1)\)-EA (cf. Tab. II).

Concerning the course of the optimization runs in Fig. 5, the results of the first five runs have been confirmed only partly. Again, the \((1+1)\)-EA manages to find improvements during the whole optimization run. In contrast to the first five runs, this can also be observed for three out of five new runs performed with the \((1+1<5)\)-MAEA.

Discussion

The \((1+1)\)-EA is able to produce improvements over the whole range of 1000 fitness function evaluations. The new results for the \((1+1<5)\)-MAEA show that this is partly possible with this parameterization as well. But still, the last strategy reached the averaged best fitness function value much earlier than the simple \((1+1)\)-EA strategy (cf. Tab. II again). Due to the fact, that after averaging ten runs, the \((1+1<5)\)-MAEA also performs better, this strategy must be labeled the best one within the study.

C. First results of linear jet optimization

Until now, only one optimization run for the complex and time consuming task of the linear jet has been performed. This seems to be too little to justify the whole procedure of presenting the results most demonstrative and descriptive according to the recommendations of Preuß [14] like it was done before.

Nevertheless, Fig. 6 presents the course of a \((1+1)\)-EA on the linear jet optimization task. Due to one fitness function evaluation taking about 1.5 hours, the whole optimization run of 1000 fitness function evaluations took more than six weeks. Although the \((1+1)\)-EA strategy was used, no improvement could be found after 597 fitness function evaluations.

The different range, the result is located in, can be explained by the other values included for fitness function calculation. This changes the number of parameters and consequently the values considered for the fitness function calculation. More runs are started and results will be pub-
lished accordingly.

A design yielding a rather bad performance from the beginning of the optimization run is displayed in the second section of this paper where the linear jet optimization problem is introduced (cf. Fig 2). Fig. 7 shows an optimized design received from the mentioned optimization run. In contrast to the previous picture, the chord length of the optimized geometry is much higher and the diameter of the hub is larger.

V. SUMMARY AND FUTURE DIRECTIONS

Using an automatic optimization technique in conjunction with geometry generation tools and hydrodynamic simulation tools, we were able to find a competitive linear jet geometry. Applying MAEA, the (1 + 1)-EA manages to produce improvements even in the later stage of the optimization run. But a more reasonable compromise between a fast and a powerful parameterization is the (1 + 1 < 5)-MAEA. This parameterization receives good results after about half the time of the other parameterization only. It gained second ranks with respect to both average best fitness function values and average evaluations after five optimization runs. The second rank for fitness function values changed to a first rank after five runs more. Within this parameterization, the meta-model is used to pre-evaluate five offsprings of the single parent, the best one is chosen to be evaluated exactly and joins the following (1 + 1) selection scheme afterwards.

The received results will be verified by more optimization runs in the near future. Furthermore, the received parameterization will provide a base for the optimization of more components of the linear jet, i.e. the tackled geometry together with a stator placed right behind the rotor inside the nozzle. With increasing number of components, the simulation time will increase again and an analysis of parameter settings will become more and more time consuming. Nevertheless, some changes within the parameterization will be tested to confirm the assumption that good parameterizations for single components provide good results for more component propulsion systems as well. Upon this, optimization runs for the (1 + 1)-EA and the (1 + 1 < 5)-MAEA are started for the linear jet optimization run.

ACKNOWLEDGMENT

This paper is based on investigations within the research project 361/15-1 “Ein Verfahren zur Optimierung von aus Mehrkomponenten bestehenden Schiffsantrieben”, which is kindly supported by the Deutsche Forschungsgemeinschaft (DFG).

REFERENCES