2nd International Conference on Engineering Optimization
September 6 - 9, 2010, Lisbon, Portugal

Design of Radial Turbine Meridional Profiles using Particle Swarm Optimization

Isaak Tsalicoglou[1], Bent Phillipsen[2]

[1] ABB Turbo Systems Ltd, Baden, Switzerland, isaak.tsalicoglou@ch.abb.com
[2] ABB Turbo Systems Ltd, Baden, Switzerland, bent.phillipsen@ch.abb.com

Abstract

In order to achieve high performance, the geometry of radial turbines in turbochargers by ABB Turbo Systems Ltd is being optimized in an iterative way through the use of both in-house and commercial CFD codes. To accommodate the lower turbine mass flow rate requirements of new applications, an existing high-performing turbine stage is modified by trimming the blade geometry to smaller outlet diameters. The new meridional contour of the blade tip must lead to the highest possible turbine stage efficiency. In this paper, an optimization system for the design of such meridional curves is presented, which applies the Particle Swarm algorithm on a radial turbine design parameterized by Bezier curve control points. The objective function is a linear combination of bounded sigmoid functions and favors designs for which the mass flow rate is achieved within a certain tolerance, the maximum Mach number of the relative velocity along the blade tip and an estimate for the energy dissipation are minimized, and various empirical geometrical constraints are fulfilled. Multiple runs, each differently initialized, show that within the time limit of the practically allowed number of design evaluations (15000), the chosen optimization scheme delivers similar variants with a maximum percent error of 0.45% for the parameter positions and a percent error of 0.05% for the mass flow rate. The optimization system is implemented in Python and calls shell scripts to handle pre- and post-processing. In-house FORTRAN software is used for the turbine geometry generation and the flow calculation. On contemporary hardware the quasi-3D flow calculation is fast enough to facilitate serial evaluation of designs at a rate of approximately 900 variants per hour and optimized design can be obtained in a daily workflow. The resulting high performance is verified computationally by fully-3D calculations using the commercial CFD code Numeca FINE/Turbo.

Keywords: turbine, design, particle, swarm, CFD

1. Introduction

High Performance Computing (HPC) has been a mainstay of engineering design for the past 30 years. Detailed CFD models (multilevel grids, turbulence modeling) allow the turbine designer to estimate the real-world performance of a turbine stage in terms of total-static efficiency with satisfactory accuracy (±0.02 points) in comparison to test rig measurements. However, the use of those code packages on the ever-increasing hardware resources is both technologically and economically constrained. The large number of parameters to be determined after the 1D design of the turbine stage in conjunction with the long calculation times arising from the complexity of simulating steady state operation of turbomachinery pose technological constraints. The use of detailed commercial CFD code packages also poses an economic constraint due to the still unresolved discrepancy between traditional “per CPU” licensing schemes and the mass adoption of multi-core processors in HPC clusters and even workstations in the past decade.

The result of these constraints is that a radial turbine stage is not being designed from the ground up using detailed, fully-3D CFD simulations. Rather, the turbine designer is called upon to apply his engineering experience and intuition on determining and evaluating a multitude of initial design variants based on simplified and much faster quasi-3D simulations in an iterative manner. Having attained a turbine design which looks promising in terms of the estimated flow conditions, a computationally more expensive and thus longer validation phase is then initiated using commercial CFD code packages. With more detailed models, the expected total-static efficiency of the turbine stage can be predicted. It is also possible to investigate those flow effects which cannot be accounted for by the simpler quasi-3D calculations and which will eventually cause a different evaluation of the efficiency in test rig measurements and ultimately in field applications.

Thanks to the immense computational power available even to workstation computers, quasi-3D CFD calculations can nowadays be executed faster than the time it takes to derive and input new design parameters. Consequently, it makes sense to automate the design process and in a second phase, to implement an optimization scheme around it. In this manner, multiple design candidates can be evaluated effectively in a non-interactive way, avoiding small errors occurring during the manual labor of inputting data and allowing greater accuracy of the attained designs.

2. Application

In order to achieve high performance, the geometry of the radial turbines in turbochargers by ABB Turbo Systems Ltd is being optimized in an iterative way through the use of both in-house and commercial CFD codes. The resulting turbine stages are both thermodynamically highly efficient, as well as satisfy mechanical reliability criteria such as HCF durability, creep resistance and burst containment.

To accommodate the lower turbine mass flow rate requirements of new applications while retaining the experimentally proven mechanical reliability, an existing high-performance, reference turbine stage is modified by "trimming" the blades down to smaller outlet diameters in post-production. A new "trim" is defined by determining a new meridional contour of the blade tip for the given 3D blade and hub geometry, so that the mass flow rate of the resulting turbine stage is equal to a certain percentage of the reference design. The new trim must satisfy the mass flow rate requirements at the highest possible turbine stage efficiency.

2.1. Formulation of the problem

The geometric problem to be solved is the identification of the two-dimensional meridional curve which optimally satisfies the criteria for the mass flow rate at the highest possible estimated efficiency, while being constrained by empirical shape factors such as waviness, curvature changes, as well as inlet and outlet angles.

Determining an optimal contour is a challenging task, as the turbine inlet shape can influence the incident flow negatively, which might lead to blade excitation and high cycle fatigue damage in operation. Additionally, the curve shape downstream of the inlet influences not only the mass flow rate, but also the Mach number of the relative velocity along the blade tip, a key contributor in energy dissipation. The iterative process of accounting for the sensitive interactions in the design can be greatly accelerated by employing an automated system.

2.2. Choice of optimization algorithm

According to [1], the mass flow rate through the nozzle-equivalent area $S_{T,eff}$ of the turbine is equal to

$$\dot{m}_T = S_{T,eff}\psi_{max}\sqrt{2p_3\rho_3} \tag{1}$$

The maximum value of mass flow function ψ_{max} depends only on the turbine pressure ratio and the upstream gas conditions, whereas p_3 and ρ_3 are the pressure and density of the exhaust gas at the turbine inlet, respectively. Since $S_{T,eff}$ is an "effective" flow surface area and correlated, yet not equal to the geometric turbine outlet surface area, trimming the blade tip radius to reduce the outlet surface area by a certain percentage leads, according to Eq.(1), to a reduction of the turbine mass flow rate by approximately the same percentage.

Implicitly, this means that optimizing only for the target mass flow rate is insufficient, as there is more than one acceptable – from a turbine engineering point of view – meridional curve that can be drawn between the inlet and outlet tip points to satisfy the mass flow rate within a specified tolerance and deliver satisfactory thermodynamic performance. Therefore, the application of an optimization meta-heuristic such as the Particle Swarm Optimization ("PSO") algorithm seems appropriate in order to automatically locate the globally optimal curve according to at least one additional engineering criterion, e.g. an estimate for the energy dissipation.

3. Implementation

The proposed optimization scheme using PSO has been implemented in Python 2.6, due to the accessibility of Python code to non-professional programmers, the fact that (Python being an interpreted language) no recompilation of the program is required after the numerous changes and adjustments undertaken during development and thanks to the plethora of available modules for performing common tasks such as reading and writing CSV files (module "csvwriter"), executing shell scripts and fetching their output (module "commands") for pre- and post-processing, generating plots (module "matplotlib"), as well as providing functionality (module "numpy") for numerical tasks, such as array operations and manipulation.

3.1. In-house turbine design and evaluation chain

The in-house software "RATIO" is used for generation of the 3D geometry of the trimmed radial turbine stage. The input to RATIO consists of a text file containing lists of control points of Bezier curves describing the rotor geometry (blade sections, blade tip contour, rotor hub contour, rotor backwall shape, blade thickness). The input also contains the thermodynamic and fluid-mechanical boundary conditions for the subsequent quasi-3D CFD calculation, for which an input file is generated.

The in-house software "HT017" is implemented in FORTRAN and calculates six blade-to-blade solutions of the quasi-3D flow through several stages of a turbomachine [2]. Polytropic efficiency or the boundary layer thickness and shape parameter at the casing surface may be prescribed to account for the viscosity influence. Flows with Mach numbers of 1.5 before or after the grid can be calculated, as long as good compatibility between local and boundary conditions of the cascade is provided. The continuity, the primary vortex systems (including the trailing edge vorticity) and the secondary vorticity of the casing boundary layers are determined iteratively. The calculated results are deemed reliable when the solution converges to less than one percent.

In 1991, the calculation time with HT017 for one blade row was about 100 seconds. In contemporary workstations equipped with multi-core x86-64 processors the total calculation time to convergence utilizing a single CPU core is typically lower than 2.5 seconds.

The in-house tool “GOPOST” is used to calculate the turbine stage mass flow rate by integrating the velocity profile over the outlet flow surface of the turbine.

3.2. Parameterization scheme and input

The input to the optimization program consists of the HT017 input file template containing the 3D geometry of the reference turbine. The reference meridional blade tip contour is input in cylindrical z-r coordinates, with z denoting the turbine axis, according to Fig. 1.

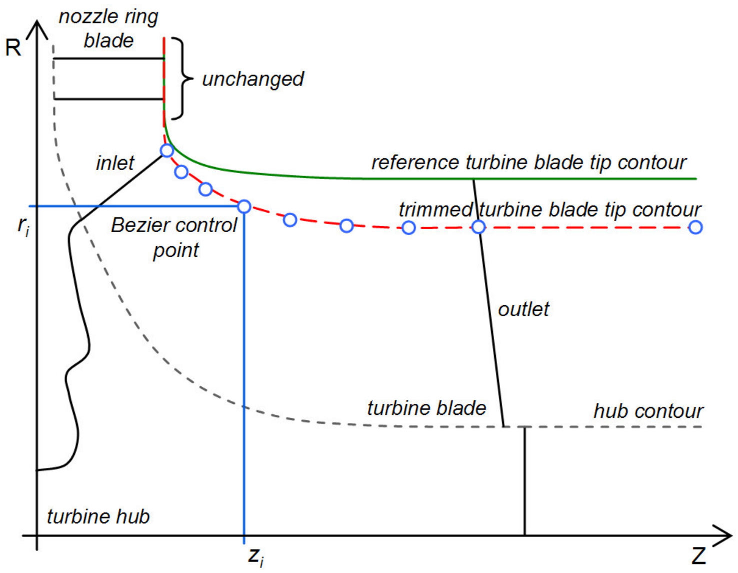

Figure 1: Radial turbine geometry nomenclature and parameterization scheme

Part of the reference contour must be defined as the “unchanged” part, since it describes the meridional contour of the chosen nozzle ring. The rest of the contour comprises the “changeable” part of the curve. This part is used as an array of parameters describing the meridional radii r_i of the k Bezier curve control points for the blade tip contour at specified axial positions z_i. The control points are distributed unevenly in the axial direction; there is an “expansion” of their positions from the inlet to the outlet by the factor ζ according to the recursive Eq.(2).

$$z_i = z_{i-1} + (z_k - z_1)\frac{1-\zeta}{1-\zeta^{k-1}} \cdot \zeta^{i-2} \tag{2}$$

Setting $\zeta = 1.3$ results in more fine-grained control of the Bezier curve in the region closer to the turbine inlet, where the slope is greater. The larger spacing between control points close to the turbine outlet implicitly helps avoid “wavy” contours, which are technically not realizable and fluid-mechanically undesirable due to boundary layer separation and the accompanying reduction of total-static efficiency. To enhance the diversity of the generated curves over multiple runs however, a random perturbation of the expanded positions by as much as $\pm\Delta z_{max}=\pm(z_2-z_1)/2$ takes place at the beginning of each run.

3.3. Particle Swarm Optimization parameters

The standard PSO algorithm described in [3] is used, and all particles communicate with each other. The position of each particle is an array of k floats, representing the radial positions of the Bezier control points.

Minor changes have been implemented in the initialization of the particle swarm parameters. The coefficients c_1 and c_2 of the particle velocity components and the inertia coefficient ω of each particle are initialized randomly (uniformly) at the beginning of the PSO loop (c_1, $c_2 \in$ [1.4, 2.2], $\omega \in$ [0.5, 0.9]).

In order to improve convergence at later iterations while allowing exploration of the parameter space in the beginning of the loop, the inertia coefficient of each particle is decreased linearly over the total number of iterations, so that by the time the maximum number of iterations is reached, all particles have reached the minimum ω value, as shown in Fig. 2.

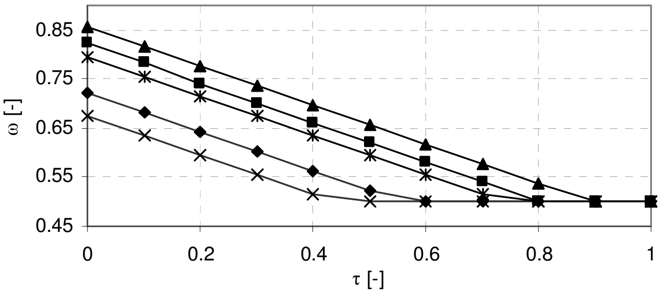

Figure 2: Linear reduction of the inertia coefficient ω to the minimum value over relative iteration number τ (number of iterations to maximum number of allowed iteration before stopping) for 5 randomly initialized particles

3.4. Curve generation and validation

The maximum change of the radius of a control point in each iteration is fixed at an order of magnitude higher than $\Delta z_{\max}$. This affects the particle velocities during the random initialization, as well as during the position updates in the PSO loop.

The initial curve population should be as diverse, and simultaneously as computable as possible. The computability of a variant depends on the fluid-mechanical plausibility of the generated geometry, since the quasi-3D CFD calculation of tip contours with abrupt changes in curvature usually does not converge.

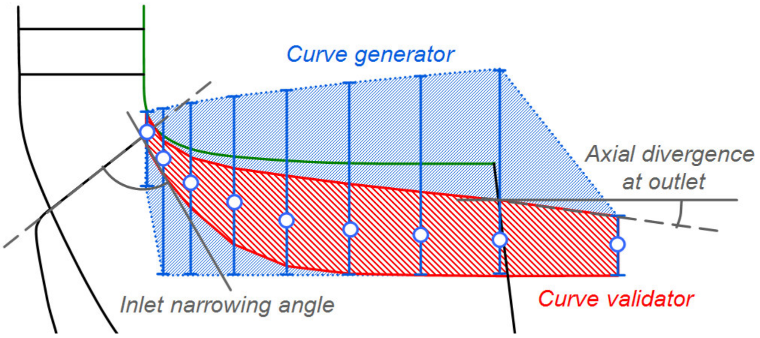

Figure 3: Parameter space of the curve generator and its subset, constrained by the curve validation function

The implemented curve generation schema is summarized in Eq.(3). The radii of the control points are determined recursively and are randomly perturbed by up to $\Delta r_{\max} = 0.5{\cdot}\Delta z_{\max}$. The radius of the control point at the turbine outlet is randomly selected from a value range [$r_{outlet,min}$, $r_{outlet,max}$] that results in a reduction of the reference turbine outlet surface area by the requested percentage (cf. Eq.(1)) with a tolerance of ±5%.

$$\begin{cases} r_1 = U(r_{inlet,min}, r_{inlet,max}) \\ r_k = U(r_{outlet,min}, r_{outlet,max}) \\ r_i = U(\min(r_k, r_{i-1} + \Delta r_{random}), \max(r_k, r_{i-1} + \Delta r_{random})) \mid \Delta r_{random} = U(0, \Delta r_{max}), \quad 1 < i < k \end{cases} \tag{3}$$

The radius of the inlet control point is constrained in a specified value range [$r_{inlet,min}$, $r_{inlet,max}$], as excessive inlet narrowing results in an acute angle between the leading edge and the blade tip, which causes the blade to be sensitive to excitation by the flow past the nozzle ring. The parameter $r_{inlet,min}$ can either be determined heuristically by hand before the optimization, or be left to influence the initial population. The first measure leads to fewer discarded curve variants, and the second method usually leads to useful designs, as the designs with a narrowed inlet feature disadvantageous flow conditions and are eventually abandoned during optimization anyhow. In the former case, a maximum reduction of the nominal r_{inlet} of the reference design by half the percentage of the mass flow rate reduction leads to a good compromise between the number of regenerated (during initialization) and abandoned (during optimization) variants. A weakly weighted component of the objective function also penalizes variants with r_{inlet} outside the initially allowed range.

The proposed curve generation schema, although effective, does not guarantee acceptable (computable) curves. Thus, the generated curves are checked by a "curve validation" function. This function applies the same geometry checks (inlet and outlet narrowing, saddle points, maximum axial divergence at outlet) which are applied as component objective functions during the evaluation phase of the PSO loop, but with a binary "accept/regenerate" behavior instead of a sigmoid function. The validation process also forces the trimmed curve to lie beneath the blade tip curve of the reference turbine.

The validated curve population is diverse and computable. An example initialization for a run with 15 particles and 8 control points with imposed axial outlet flow ($r_7 = r_8$) is presented in Fig. 4.

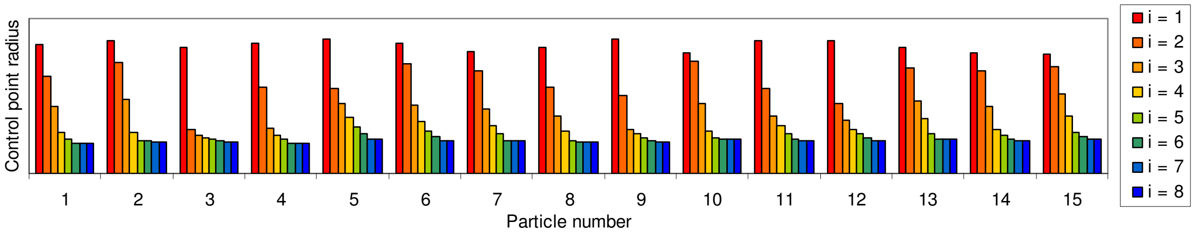

Figure 4: Sample of 15 generated and validated control point radius sets after the particle initialization

3.5. Transitional designs

During optimization, the change of the radial coordinates of the Bezier control points sometimes leads to designs which are infeasible according to fluid mechanics and established experience in turbomachinery design. Although these designs would be rejected by the curve validation process, they are accepted for three reasons.

Firstly, through them the exploration of diverse solutions is facilitated by not forcing the variants to strictly adhere to rules about the contour shape during optimization; rather, those variants will eventually be abandoned due to penalties imposed through weakly weighted components in the component f_{shape} of the composite fitness function concerning the curve geometry.

Secondly, it enables relatively rapid transitions between “solution clusters” by not “waiting” for all parameters to be improbably changed in the same “sensible” way between iterations.

Thirdly, strong penalties are imposed on estimated performance characteristics of the turbine stage by f_{ed}, which means that those designs which look drastically unconventional based on engineering knowledge and experience are only transitional and have few chances of “survival” as the swarm abandons them and converges to the global optimum.

4. Evaluation and convergence

4.1. Non-convergent CFD calculations

Depending on the initial particle population, and mostly during the first iterations, some of the transitional variants are infeasible enough to pose convergence difficulties to the quasi-3D flow calculation. Since the variants are evaluated serially this represents a significant bottleneck in the optimization loop, which is overcome by employing a “CFD convergence watchdog” shell script running in the background, parallel to the optimization Python script.

The watchdog parses the file-redirected standard output (stdout) stream of the quasi-3D calculation for the string “NaN” (“Not a Number”) with a period comparable to the duration of a normal HT017 run (2.0 seconds). Presence of “NaN” indicates convergence difficulties; as soon as this is detected, the watchdog terminates the HT017 process and creates a dummy file. As soon as the Python script regains control from the last HT017 execution in the shell it immediately checks for the presence of the dummy file; in which case it deletes it, forgoes evaluation of the current variant and overrides the value of the composite objective function by setting it to a predetermined maximum value.

The large change in particle velocity and subsequent position caused by the large differences between the current non-convergent position and the personal best and global best positions is avoided by imposing the velocity bounds on the particle between iterations. Despite that measure, it is possible for a particle to move into a completely unfit part of the parameter space, where its calculation will not converge for many consecutive iterations. For that reason, a “non-convergence list” of all particles is maintained. As soon as a “problematic” particle crosses the limit of 5 consecutive non-convergent iterations, it is reset by being newly initialized using the parameter generator and the curve validation functions.

4.2. Stopping criteria

“Soft”, as well as “hard” stopping criteria of the optimization loop are used. The hard stopping criterion terminates the PSO loop after a predefined number of design variants. The soft stopping criterion is implemented on two levels: a particle activity check and an overall activity check. A particle is considered active and participates in the swarm as long as its parameters (r-coordinates) rounded to 3 decimal places are different than the similarly rounded parameters of the currently known global best position. The particle that found the last global best position is excluded from this check. If that condition is not satisfied, it is assumed that the rest of the swarm is on its way to convergence and that this particle has practically "landed"; it is marked inactive and does not contribute to the swarm in subsequent iterations. The overall activity check is also implemented on two levels: if all particles have become inactive, the PSO loop stops. The loop also stops if the accumulated difference of parameter values between all active particles and the global best position falls below a certain threshold value, e.g. 1 mm.

4.3 Composite objective function

The variants are evaluated using a composite objective function, comprised of a weighted sum of three main components. The components themselves are comprised of sums of criteria, and correspond to evaluation of the mass flow rate (f_{mfr}), the energy dissipation (f_{ed}), and the curve geometry (f_{shape}). Of those, f_{shape} has the weakest contribution, as the estimated energy dissipation is coupled to the Mach number profile at the blade edge, which is strongly connected to the geometry. The geometry component is thus implemented as a weak selection mechanism, primarily to discard infeasible variants. The component objective functions have been adapted from [4], where they are described as "bounded constraint mapping functions".

4.4. Target value component objective function

Eq.(6) describes the target value constraint mapping function, bounded in [0,1]. The function parameters σ^2 and ξ are determined using Eq.(4) and Eq.(5) and the chosen location and shape parameters C_{target}, D_{feas}, $C_{feas,tol}$ and $C_{adm,tol}$.

The value of parameter C must attain the target value C_{target} in the "admissible interval" $C_{target} \pm C_{adm,tol}$. The value of D_{feas} corresponds to the requested penalty function value at the boundaries of the "feasible interval" $C_{target} \pm C_{feas,tol}$. It determines how feasible parameter values are penalized in comparison to unacceptable values, influences the slope of the function on either side of the interval, and defines the point of curvature change.

$$\sigma^2 = \frac{\left(C_{feas,tol} - C_{adm,tol}\right)^2}{-2\ln\left(1 - D_{feas}\right)} \tag{4}$$

$$\xi = \left|C - C_{target}\right| - C_{adm,tol} \tag{5}$$

$$f_{target}(C) = \left(1 - \exp\left(-\frac{\xi^2}{2\sigma^2}\right)\right) \cdot \frac{1 + \mathrm{sgn}(\xi)}{2} \tag{6}$$

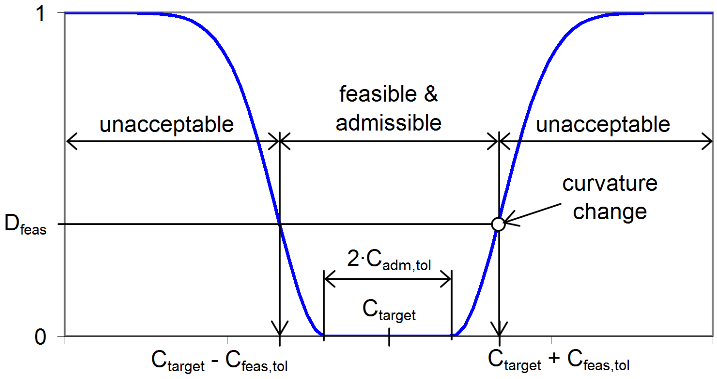

Figure 5: Description of the parameters of the target value component objective function

4.5. Upper limit component objective function

The target value constraint mapping function f_{upper} can be derived by f_{target}, so that only values above the parameter value limit $C_{limit}=C_{target}+C_{adm,tol}$ are penalized. This is achieved by "deleting" the left of the two sigmoids of which f_{target} consists by multiplication, as in Eq.(7). This function is also bounded in [0,1]. Feasible values of parameter C lie in the range [C_{limit}, $C_{target}+C_{feas,tol}$].

$$f_{upper}(C) = f_{target}(C) \cdot \frac{1 + \mathrm{sgn}\left(C - C_{target}\right)}{2} \tag{7}$$

Correspondingly, for a lower limit component objective function the right sigmoid of f_{target} can be "deleted" through multiplication by 0.5·(1-sgn(C-C_{target})). However, no such criteria are applied in the herein described application.

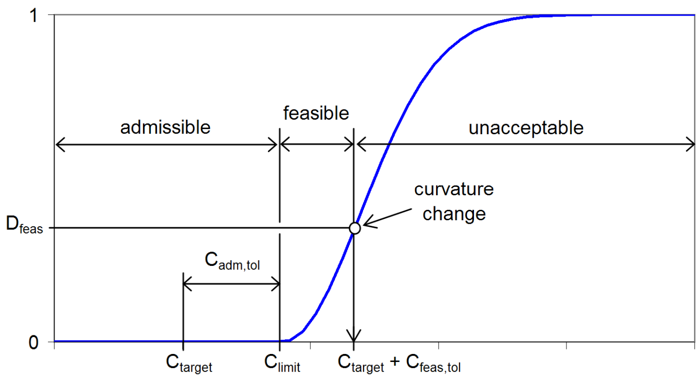

Figure 6: Description of the parameters of the upper limit component objective function

4.6. Energy dissipation evaluation

As a first step, the mass flow rate of the reference turbine is calculated. The target mass flow rate for the trimmed turbine is given as a percentage of the reference. Additionally, the maximum Mach number along the meridional blade edge of the reference design serves as an indirect performance requirement by being accounted for in the composite objective function.

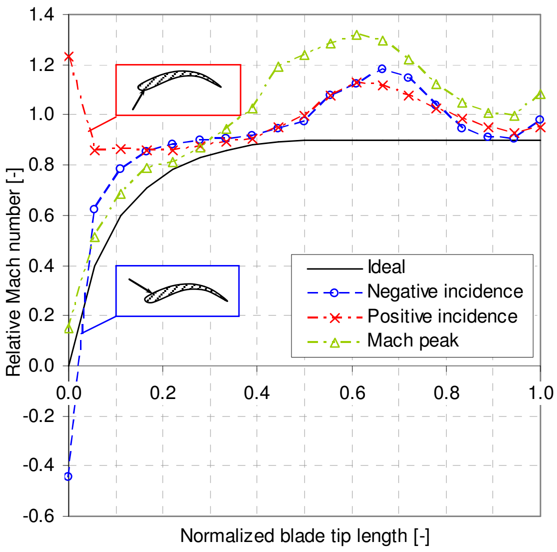

Figure 7: Example cases for the relative Mach number profile across the blade tip (suction side)

The quasi-3D flow calculation returns data for the profile of the Mach number of the relative velocity along 6 paths (blade "cuts") across the height of the blade, on the suction side. Energy dissipation due to viscous effects scales with the third power of the relative velocity of the free stream [5]. To obtain an estimate for the largest part of energy dissipation over the blade surface, and at the same time penalize designs with positive incidence, the profile of the absolute value of the relative Mach number along the tip "cut" to the third power is integrated over the normalized tip length. A more accurate measure would be the integral of this quantity over the whole blade surface. For a radial turbine however, the maximum relative velocity close to the hub is much lower (approx. 50%) than at the blade tip. Combining this property of the radial turbine with the third-power scaling of the relative velocity means that optimizing the design with the partial goal of reducing the value of the $|Ma|^3$ integral results in improving on the most significant factor contributing to loss of efficiency.

4.7. Curve shape evaluation

Initially, the inlet and outlet radii of the tip contour were the only constraints imposed on the curve form. However, the resulting optimal designs were of the shape depicted in Fig. 8, featuring an artificial narrowing of the outlet, usually preceded by a saddle point. This form still satisfies the curve validation criteria. In that case, the optimization scheme has exploited the fact that, by enlarging the available surface area between the blades at a specified axial coordinate the mean velocity of the flow can be locally reduced, and the usually observed Mach number jump to values higher than 1.0 can be weakened, or altogether avoided, depending on the rest of the turbine geometry. At the same time, the required mass flow rate will be attained by reducing the outlet diameter. Extreme cases of this exploit can be observed by loosening the criteria of the curve generation/validation, so that the curve is not forcedly convergent from inlet to outlet. Indeed, a reduction of the Mach number across the tip can be observed when the curve features divergent/convergent “bumps”. However, using a turbine with such a tip shape is prevented by a) the mounting concept of axially sliding the gas outlet flange over the turbine, and b) by the resulting great variability of the tolerated tip gap between the turbine blade edges and the gas outlet flange due to thermal expansion of the parts in operation.

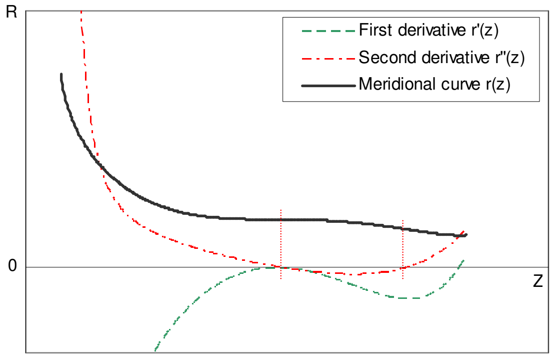

Figure 8: Example of a convergent/divergent meridional curve

In order to suppress the consideration of such variants during optimization, two additional criteria for the curve shape are formulated: the radius must be monotonically decreasing from inlet to outlet, and the first derivative of the radius must be monotonically increasing from negative values towards zero. An example of a curve that does not satisfy these criteria is shown in Fig. 8.

The monotony criteria are checked by counting the number of sign changes. The expected number of sign changes for an acceptable curve shape is zero for both derivatives. The component objective function applied on this criterion is a binary step, with which curves with sign changes of the derivatives are penalized with the value 1.

4.8. Applied criteria

Table 1: Summary of the applied criteria and the corresponding component objective functions

Code	Objective	Type	C	C_{target}	Relative weight
S1	Penalize artificial outlet narrowing	Target	$\min(r)$	r_k	0.025
S2	Reward axial outflow	Target	r_{k-1}	$1.05{\cdot}r_k$	0.025
S3	Penalize artificial inlet narrowing	Target	r_1	$r_{\text{nozzle ring}}$	0.05
S4	Reward convergent curve	Binary	# of 1st derivative sign changes	0	0.05
S5	Penalize convergent/divergent curve	Binary	# of 2nd derivative sign changes	0	0.05
M1	Reward mass flow rate	Target	$\dot{m}_T / \dot{m}_{T,req}$	1	0.5
E1	Penalize high *Ma* peak	Upper limit	max(*Ma(ℓ)*)	1.0	0.1
E2	Reward low $\|Ma\|^3$ integral	Upper limit	$\int_0^1 \|Ma(x)\|^3 d\ell$	0.78	0.1
E3	Reward monotonically increasing *Ma*	Upper limit	# of 1st derivative sign changes of *Ma(x)*	0	0.1

The 9 component mini-objectives applied during the evaluation of a design are listed in Table 1. The relative weights correspond to a weighing of the three components according to Eq.(8):

$$f_{total} = 0.5 \cdot f_{mfr} + 0.3 \cdot f_{ed} + 0.2 \cdot f_{shape} \,. \qquad (8)$$

5. Results

5.1. Run time

Prediction of the wallclock run time is straightforward, as the time required for the evaluation chain, the “housekeeping” shell commands and the Python script run time of each variant is normally shorter than 3.5 seconds. Experience gathered with the optimization program over more than 15 trims and multiple runs using different turbine geometries, target mass flow rates, initialization constraints and weightings of the component objective functions shows that a swarm of 15 particles evaluated by Eq.(8) consistently delivers plausible and “good enough” solutions within the first 2 hours and “exceptionally good and accurate enough” solutions usually at least 4 hours before the hard stopping criterion of 15000 design variants has to be enforced.

5.2. Convergence behavior

For the purpose of determining the most effective PSO parameter combination in terms of achieved results over a specified time period, batches of 7 differently initialized runs with the same limits on the parameters c_1, c_2 and ω, as well as the same number of particles have been investigated. For all batch runs, the soft limit was deactivated and the hard limit was set as the evaluation of 15000 design variants, which corresponds to a development time which can sensibly be integrated in a daily work cycle (i.e. “overnight” optimization).

In the empirically determined best setup, each of the 15 particles is initialized with a uniformly random c_2. The value of c1 is drawn randomly from the same interval and then scaled with the factor 0.75, so that the particle swarm demonstrates more social than cognitive behavior. A comparison of the evolution of the average global best objective function over the 7 runs of each batch over the normalized runtime is shown in Fig. 9. Doubling the number of particles potentially delivers similar results in less than half the time; however, no significant improvement is observed from that point on. Almost halving the number of particles brings no advantage whatsoever. With 15 particles, setting one of both c_1 and c_2 parameters as 1/3 of the other one leads to worse results.

Among the 7 runs of the best setup batch, the average percent error for the mass flow rate is equal to 0.05%, and the maximum and average percent errors among the control point radii are equal to 0.45% and 0.27%, respectively, demonstrating very good convergence to a final design.

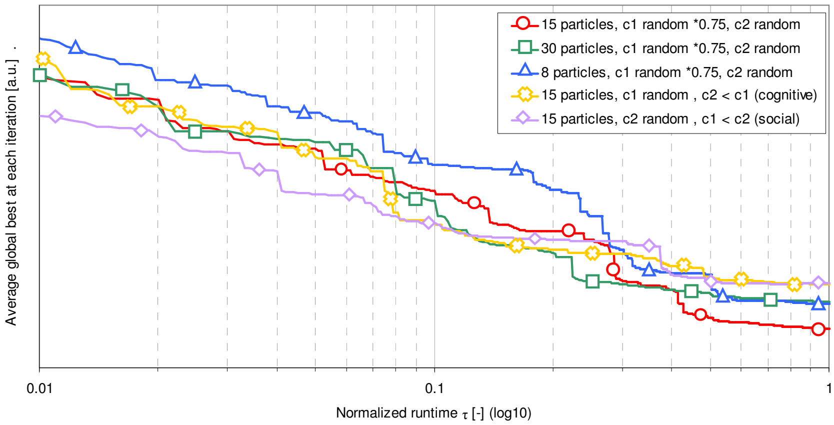

Figure 9: Average convergence (PSO “global best”) of 5 batches with 7 runs each, with different number of particles and initialization of the cognitive and social parameters

5.3. Result validation with CFD

The commercial fully-3D CFD code Numeca FINE/Turbo has been used to verify the good thermodynamic performance of the optimally trimmed turbines which is expected according to the quasi-3D calculations and the applied evaluation criteria. An example of the validated results of the optimization scheme is shown in Fig. 10. Despite modifying an already optimized 3D geometry to attain a 20% lower mass flow rate, instead of an expected decrease of the total-static turbine efficiency due to the new, potentially suboptimal blade shape, an increase by 1% relative to the reference is predicted computationally.

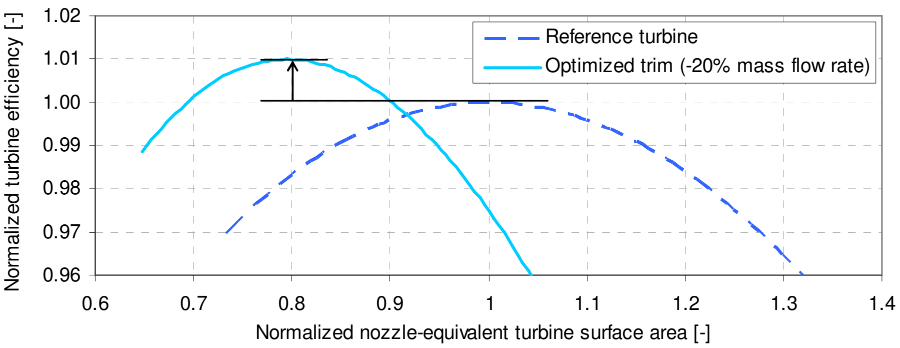

Figure 10: Comparison of the total-static turbine efficiencies calculated using fully-3D CFD between a reference turbine and an optimally trimmed turbine with 20% mass flow rate reduction

6. Summary and outlook

6.1. Possible improvements

Improvements could be undertaken on the variant initialization scheme. The necessary geometry checks and evaluation arise from the fact that, although a monotonically decreasing radius is expected, the variant parameterization scheme allows – and is in fact based on – the absolute value of each control point radius, independently of the neighboring control points. This leads to partially diverging curves, whereas a recursive parameterization of the curve could inherently forbid implausible shapes.

The current evaluation of variants in series means that massive reductions of the total run time could be achieved by object-wrapping the legacy CFD software as a Python module, multiple instances of which can be parallelized using the Python “multiprocessing” module. As such, all particles in a given iteration could be evaluated asynchronously on different processor cores of nowadays commonplace multi-core-equipped workstations.

6.2. Conclusion

An optimization scheme has been presented, which allows for efficient automated optimization of radial turbine meridional profiles using very fast quasi-3D turbomachine flow calculations as the evaluation part of a Python implementation of the standard PSO algorithm.

Due to the high computational capacity available, simulation software written more than two decades ago can nowadays be applied on engineering development tasks with almost negligible computational cost. Additionally, such in-house tools have the additional benefit of incurring no software licensing costs. Correspondingly, trading model complexity for low run times means that even if the computational accuracy shortcomings of the simpler model are taken into account, legacy in-house simulation software can still be repurposed as a fast design evaluation tool in a modern, cost-effective optimization chain.

Such an optimization chain can also be used to efficiently investigate different requirement scenarios, such as higher thermodynamic performance at different mass flow rates and/or pressure ratios, without the need to accompany the investigation with manual data input and output.

The application of the PSO algorithm on this problem is very effective, with multiple runs reproducing similar optimal designs with percent errors in order of magnitude of 1%. The algorithm itself is computationally cheap, which makes the use of an interpreted and versatile programming language, such as Python, fitting. The run time necessary for convergence to “exceptionally good” variants is low enough to be usable as part of a daily workflow, even though the highly parallelizable variant evaluation of PSO has not yet been taken advantage of.

The validation of the expected good thermodynamic performance and the requested mass flow rate reduction using fully-3D CFD indicates that the current optimization scheme could be applicable on the whole geometry of radial turbines and compressors.

References

[1] H. Hiereth and P. Prenninger, *Charging the Internal Combustion Engine*, Springer-Verlag, Wien, 2007

[2] M. Ribaut, A Full Quasi-Three-Dimensional Calculation of Flow in Turbomachines, *Trans. of ASME, JTM, v.110*, July 1988.

[3] J. Kennedy and R. Eberhart, Particle Swarm Optimization, *Proceedings of IEEE International Conference on Neural Networks*, IV, pp. 1942-1948, 1995

[4] O. König, Evolutionary Design Optimization: Tools and Applications, *Dissertation ETH No. 15486*, Swiss Federal Institute of Technology Zurich, 2004

[5] J.D. Denton, Loss Mechanisms in Turbomachines, *ASME 93-GT-435*, 1993