INTELLIGENT PERFORMANCE CFD OPTIMISATION OF A CENTRIFUGAL IMPELLER

Simone Pazzi, Francesco Martelli
Department of Energetics "Sergio Stecco" – University of Florence – Florence - Italy
E-mail: s.pazzi@ing.unifi.it, martelli@ing.unifi.it

Vittorio Michelassi
DIMI, University of Roma Tre, Rome – Italy
E-mail: michelas@uniroma3.it

Marco Giachi
Nuovo Pignone – GE, Florence – Italy
E-mail: marco.giachi@np.ge.com

Frank Van den Berghen, Hugues Bersini
IRIDIA Labs, ULB, Brussels – Belgium
E-mail: fvandenb@iridia2.ulb.ac.be, bersini@ulb.ac.be

ABSTRACT

A typical centrifugal impeller characterised by a low flow coefficient and cylindrical blades is optimised by means of an intelligent automatic search program. The procedure consists of a Feasible Sequential Quadratic Programming (FSQP) algorithm [6] coupled to a Lazy Learning (LL) interpolator [1] to speed-up the process. The program is able to handle geometrical constraints to reduce the computational effort devoted to the analysis of non-physical configurations. The objective function evaluator is an in-house developed structured CFD code. The LL approximator is called each time the stored database can provide a sufficiently accurate performance estimate for a given geometry, thus reducing the effective CFD computations.

The impeller is represented by 25 geometrical parameters describing the vane in the meridional and s-θ planes, the blade thickness and the leading edge shape. The optimisation is carried out on the impeller design point maximising the polytropic efficiency with more or less constant flow coefficient and polytropic head. As a preliminary test the optimisation algorithm takes into account only eight geometrical parameters describing the impeller in the meridional plane. The optimisation, carried out on a cluster of eight PCs, is self-learning and leads to a geometry presenting an increased design point efficiency. The program is completely general and can be applied to any component which can be described by a finite number of geometrical parameters and computed by any numerical instrument to provide performance indices.

The work presented in this paper has been developed inside the METHOD EC funded project for the implementation of new technologies for optimisation of centrifugal compressors.

INTRODUCTION

Centrifugal compressors are nowadays utilised for many applications. At industrial level they can be used in oil and gas and chemical industry from extraction, gas liquefaction and transportation to reforming and cracking in refinery, from gas synthesis to air fractionation, in chemical and pharmaceutical processes. Other applications of these machines can be found in power engines, aeronautics and helicopter engines. Power and dimensions of these machines cover a range from very small configurations for micro applications to huge multistage groups for heavy industrial plants. It is clear that, in case of large industrial applications in which the power consumption of a multistage centrifugal compressor can reach 70 MW, the search for a good efficiency of the machine has to be a must for designers.

In order to improve the compressor efficiency, much effort has been devoted to the impeller, considered the “heart” of the compressor, and thus deserving a special attention on part of designers. Many works can be found in the technical literature dealing with the optimal design of centrifugal impellers, ranging from the classical one-dimensional methods providing loss correlations to more sophisticated ones exploiting CFD codes which tend to be used mostly as instruments for the verification of performance rather than as actual design tools. It is a matter of fact that new and powerful instruments and techniques can be used today to improve a component’s performance. With the growth of innovative methods of optimisation, like Genetic Algorithms (GA) and Evolutionary Strategies (ES), activities in many research fields have moved a considerable step further. The development of Artificial Intelligence has exerted a positive influence also in aerodynamics where it has been demonstrated that new configurations of wings, blades and channels can be obtained, starting the optimisation process from existing geometries, with a remarkable increase in performance. In the cases of Mosetti et al. [10] and Quagliarella et al. [13], for example, Genetic Algorithms have been coupled to CFD solvers. The authors show how, in a relatively small time, the CFD code is able to perform all the computations after the changes of input parameters given by the GA and find the geometry giving the optimum solution in terms of a previously defined objective function representing one or more performance parameters. Cosentino et al. [4] used a Genetic Algorithm (GA) coupled to an Artificial Neural Network (ANN) to optimise a three-dimensional impeller described by fifteen geometrical parameters and giving a target Mach number distribution on the blade surfaces.

Following this approach, although exploiting different algorithms, it was decided to begin the implementation of a procedure to automatically optimise an impeller given a general target function. The method, exploiting a Feasible Sequential Quadratic Programming (FSQP) optimisation algorithm (Fletcher [6]) coupled to a Lazy Learning interpolator (Aha [1]) to speed-up the process, is able to automatically search for those geometrical parameters which describe an impeller with performance improved with respect to the starting configuration. It is clear that this expert approach required an accurate analysis of existing impeller configurations in order to identify the most useful set of both geometrical parameters, which are changed directly by the FSQP algorithm, and performance indices whose combination defines the target function to be maximised or minimised. In the present work the optimisation is carried out only on the design point, although several studies (Michelassi and Pazzi [9]) have demonstrated that impeller performance, in terms of characteristic curve, evaluated in steady ideal conditions can significantly differ from the ones computed in unsteady regimes taking into account the time-variable flow distortions induced by the upstream stator components. In the present work some preliminary results obtained on a test impeller will be shown to highlight the efficiency and flexibility of this method representing, in our opinion, that class of engineering design tools which will be more and more utilised in the very next years.

THE CFD CODE

The flow inside the impeller is computed by means of an implicit Navier-Stokes (N-S) solver, XFLOS, developed in-house by the Energetics Department of Florence. The equations enforce the balance of conservative variables as density, momentum, and total specific energy. The variables are made non-dimensional with respect to the inlet total pressure P_0 and inlet total temperature T_0. Viscosity and diffusion coefficients are made non-dimensional with respect to the inlet laminar viscosity. The equations are discretised by centred finite differences in a curvilinear non-orthogonal three-dimensional co-ordinate system. The algorithm is based on the scalar approximate factorisation by Pulliam and Chauseè [12] and implemented for internal flow aerodynamics by Michelassi and Belardini [8]. The solver is based on the ADI sequence stemming from the scalar approximate factorisation method.

The algorithm includes second and fourth order non linear artificial damping in both the right hand side and the implicit side of the operator in order to achieve the large degree of robustness required in such complex flow fields. To take full advantage of the implicit time marching formulation the solver implements a local time step strategy. The solver is designed for memory intensive simulations and it requires approximately 60 real numbers per grid node.

In view of the wide engineering applicability of the solver, the effect of turbulence on the mean flow field is modelled by a formulation based on the Boussinesq assumption. Among these the k-ω model (k is the turbulence kinetic energy and ω is the turbulence frequency) proposed by Wilcox [14] was selected for the present set of calculations because of its superior robustness and accuracy as compared to other similar formulations. The overproduction of turbulence in stagnation points, or in flow regions with adverse pressure gradients, is a potential source of inaccuracy. To overcome this problem, which is common to all two-equation turbulence models, the solver introduces the realizability constraint proposed by Durbin [5] which limits the turbulence time scale to limit the turbulence level in stagnation regions.

The code can use relative or absolute variables in the rotating frame of reference, and absolute or relative boundary conditions. All the present runs are performed by using absolute variables and boundary conditions.

The structured mesh generator is an in-house developed code creating I-type grids. The I structure allows easy treatment of tip clearance and, although the tested impeller is shrouded, the I structure was retained for this set of calculations. The grid is generated once the impeller camber lines at hub and shroud are given by points in a cylindrical reference frame, together with local blade thickness. The distribution of grid nodes along the geometry can be easily controlled by the user. Particular attention was devoted to the leading edge discretisation. The code is able to mesh both shrouded and open impellers.

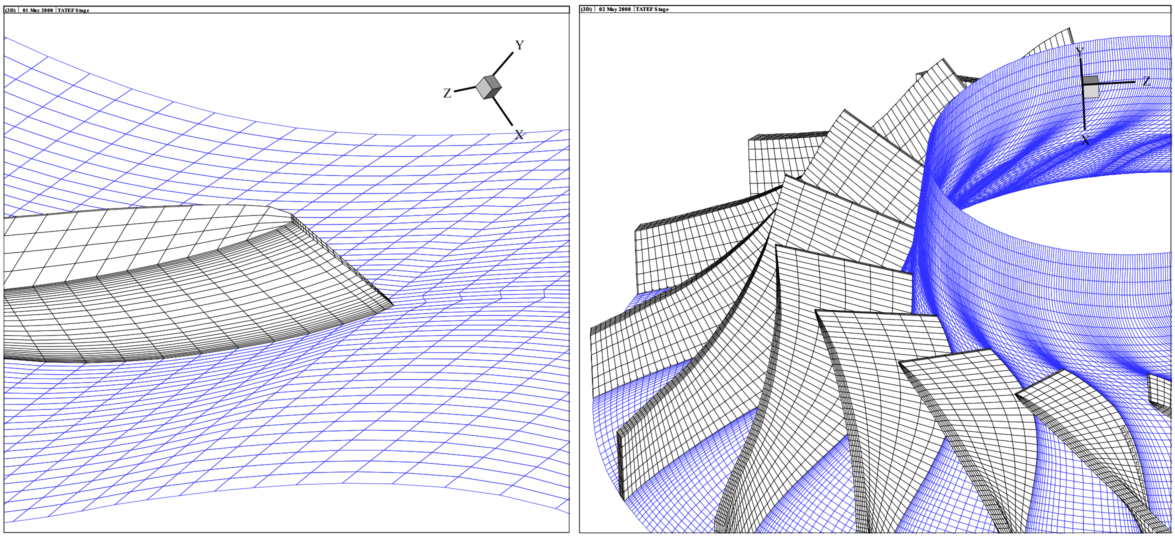


Figure 1. Particular of the blade leading edge discretization (left) and view of a coarse grid on an open impeller (right).

The code was developed in order to be easily used for massive and repeated computations during which small changes to the blade geometry are carried out in sequence thus allowing the utilisation of a standard input file for all configurations. In view of the adoption of the structured mesh generator for sequential computations, some automatic check features for grid quality improvement were implemented.

THE OPTIMISATION ALGORITHM

A general optimisation problem consists in finding the values of some variables describing a model that minimize or maximise a determined objective function while satisfying some established constraints. Optimisation problems are made up of three basic ingredients:

- ✓ An objective function which has to be minimised or maximised.
- ✓ A set of unknowns or variables which affect the value of the objective function.
- ✓ A set of constraints allowing only certain values of the design variables.

The general constrained optimisation problem is to minimise a non-linear function subject to non-linear constraints. A mathematical formulation can be stated as follows:

$$\min\{f(x): c_i(x) \leq 0, i \in \Gamma, c_i(x) = 0, i \in \Im\}$$

where each c_i is a mapping from $\Re^n$ to $\Re$, and Γ and $\Im$ are index sets for inequality and equality constraints, respectively.

The main techniques that have been proposed for solving constrained optimisation problems are reduced-gradient methods, sequential linear and quadratic programming methods, and methods based on augmented Lagrangians and exact penalty functions. Other algorithms, like Evolutionary Strategies or Genetic Algorithms, can as well fit these problems although not intrinsically constrained.

Many algorithms satisfying these requirements are nowadays available and employed at many levels in engineering processes.

The choice of the optimisation algorithm for the impeller was quiet delicate because of the many aspects involved in the analysis. The algorithm was required to be fast, reliable, sophisticated enough to find an optimum in a complex model, and constrained.

The initial choice of a Genetic Algorithm (see Goldberg [7]) was soon discarded because it seemed not possible to use this technique with such a big search-space. With this in mind it was decided to use a modified version of the *Feasible Sequential Quadratic Programming* (FSQP) algorithm which has been implemented and tested by IRIDIA labs at the Université Libre de Bruxelles. The sequential quadratic programming (sequential QP) algorithm is a generalisation of Newton's method for unconstrained optimisation in that it finds a step away from the current point by minimising a quadratic model of the problem. In its purest form, the sequential QP algorithm replaces the objective function with the quadratic approximation:

$$q_k(d) = \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla_{xx}^2 L(x_k, \lambda_k) d \, ,$$

where $\nabla_{xx}^2 L$ is an approximation of a part of the Hessian matrix of the objective function Lagrangian. The constraint functions are replaced by linear approximations. The step d_k is then calculated by solving the quadratic subprogram

$$\min\left\{q_k(d): c_i(x_k) + \nabla c_i(x_k)^T d \leq 0, i \in \Gamma, \; c_i(x_k) + \nabla c_i(x_k)^T d = 0, i \in \Im\right\}$$

Feasible sequential quadratic programming algorithms, as their name suggests, constrain all iterates to be feasible. They are more expensive than standard sequential QP algorithms, but they are useful when the objective function, as in our case, is difficult or impossible to calculate outside the feasible set, or when termination of the algorithm at an infeasible point (which may happen with most algorithms) is undesirable. The FSQP code solves problems of the form

$$\min\{f(x): c(x) \leq 0, Ax = b\}$$

In this algorithm, the step is defined as a combination of the sequential QP direction, a strictly feasible direction (which points into the interior of the feasible set) and a second-order correction direction. This mix of directions is adjusted to ensure feasibility while retaining fast local convergence properties.

Feasible algorithms have the additional advantage that the objective function can be used as a merit function, since, by definition, the constraints are always satisfied. In order to reduce the time necessary for the optimisation, the FSQP algorithm was coupled to a data modeller able to substitute the CFD code giving an approximation of the performance indices once a database of geometrical parameters and corresponding results are provided. For this purpose a local modeller was preferred to a global one, as an Artificial Neural Network. The first one, in fact, provides a description of the system by combining several models pertaining to different operating regimes. Each of the models is obtained giving full attention to a reduced portion of the space of the possible solutions, yielding a more accurate description even when simple approximators (for example linear models) are used. It is precisely the simple form of the local models, and consequently the possibility of handling them using standard and well-known tools from linear statistics, that makes the local approach appealing. A second reason for the growing popularity of local methods is that the decomposition of the learning task into sub-tasks makes the whole process easier to manage, allowing for instance the integration of physical models into the black-box description. The modelling process, adopting such methods, starts with a training phase during which the examples available are used both to extract the local descriptions of the system and to define a partition of the space of the operating regimes. Any request for information is fulfilled by interpolating the answers of different local models.

Lazy learning (Atkeson et al. [2]), also known as just-in-time learning, is inspired by nearest-neighbour techniques and by non-parametric statistics. It defers processing of the examples until an explicit request for information is received. When this happens, the database available is searched for those examples that, according to some measure of distance, are considered most relevant to answer the query. These examples are used to extract a local description of the system, for example through a local linear model, and finally to fulfil the request. Both the answer and any intermediate results are then discarded and each following request for information will make the full process start again.

The lazy approach is able to deal effectively with situations in which the examples are not evenly distributed or when the noise affecting the data is described by different distributions for different operating regimes. Moreover, since the training phase is computationally inexpensive and simply amounts to a storage of the available examples into a database, the lazy approach is particularly suitable when the examples are not all available from the beginning but are collected on-line. This makes Lazy Learning a very well fitting technique for our purposes while the adoption of a global modeller like ANN would have required continuous training loops for each estimation. In this case, on the contrary, a new example observed requires only an update to the database. It is worth noticing that, contrary to global approximators, lazy learning does not suffer from data interference. That is, acquiring examples about an operating regime does not degrade modelling performance about others. The drawbacks of lazy learning are mainly associated with the necessity of a possibly large amount of memory to store the data, and with the fact that each request for information involves starting the identification of a local model from scratch. Nevertheless the evolution of computer hardware has already partially eased these problems.

Birattari et al. [3] test the performance of the Lazy Learning algorithm, showing how, compared to others, it shows to be one of the best regression algorithms, especially when the number of dimensions (geometrical parameters plus performance indices, in our case) is high.

As mentioned before, the LL regressor has been coupled to the optimisation algorithm in order to speed-up the process. Each time an objective function evaluation is required, in fact, the LL performs an evaluation of the Quality of the Prediction (called ‘QP’) using leave-one out cross-validation. This evaluation is available at no costs as an internal result of the calculation of the prediction. QP is used (with other small heuristics) to decide if the LL prediction should be used instead of the CFD code result. In this case the computation time is clearly shortened since the CPU time for an LL prediction is almost zero if compared to the one required by a full 3D CFD analysis.

THE IMPELLER PARAMETERISATION

Similarly to what was done by Cosentino et al. [4] a detailed set of geometrical parameters able to fully and accurately describe the impeller was chosen. The parameters, which can be used both for mixed and radial flow impellers, can be divided into three main groups. The first one collects the parameters describing hub and shroud camber lines, including the inlet vaneless channel, in the meridional plane. The second one is the group containing the parameters describing hub and shroud camber lines in the s-θ plane, where s and θ are respectively the curvilinear and the tangential co-ordinate. The third group consists of those parameters necessary to completely describe the blade once its camber line has been defined.

The first group gathers the eight control points of the third order Bezier curves representing hub and shroud in the meridional plane (see Figure 2). This kind of curves, as suggested by Poloni [11], were chosen both for their capacity of accurately representing a curve with a restricted number of parameters, and for the flexibility of control points for geometrical constraints handling. The first group also contains the data representing the linear radial extension of the blade and the eventual slope of the linear shroud segment on the meridional plane.

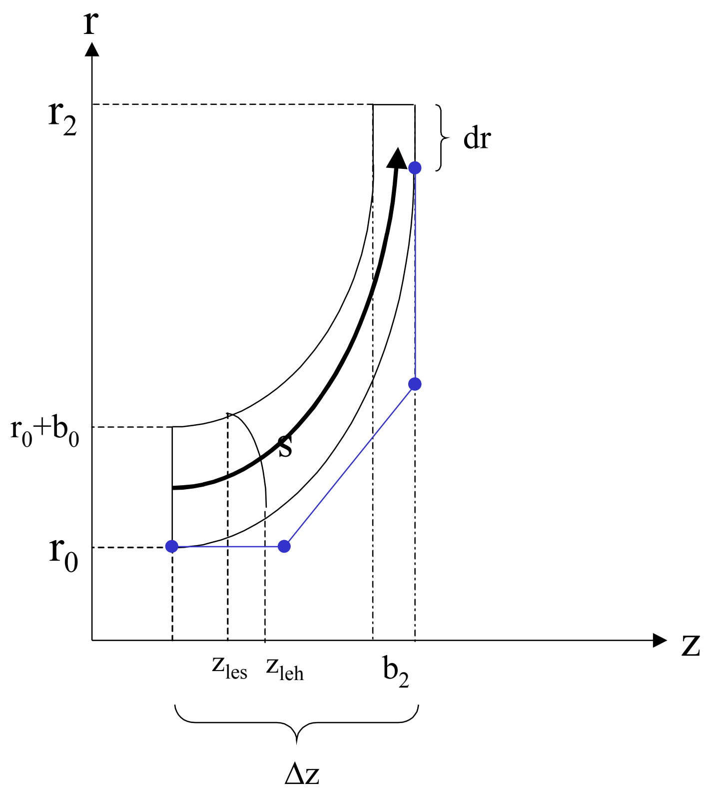


Figure 2. Sketch of the impeller parameterisation on the meridional plane.

Also for the description of hub and shroud on the s-θ plane, third order Bezier curves were adopted.

Once the blade camber lines have been completely described, the parameters belonging to the third group define the thickness at hub and shroud and the shape (elliptic) of the leading edge.

It's important to highlight that this impeller parameterisation is able to represent also innovative blade shapes characterised by double curvature surfaces, leaned leading and trailing edges, variable thickness distribution and other features which, for the test performed in this work, have not been considered.

PERFORMANCE INDICES

The necessity to implement an easy to use tool for an automatic optimisation of the impeller required the choice of simple but meaningful performance parameters. The choice made by Cosentino et al. [4] to impose as target function a determined Mach number distribution on the blade surfaces seemed not to match our requirements, both because this would imply the necessity of an expert user who, given a certain impeller typology, has in mind the optimal blade loading distribution, and because this approach cannot be fully generalised to innovative impeller geometries.

With this in mind we decided to adopt as performance indices those non-dimensional parameters which are normally used in industry to classify impellers (or stages) for the common applications. In this way we also would meet the expectations on part of industry to deal with a program which can be exploited at several levels of technical knowledge.

The first performance index is the inlet flow coefficient φ_1 defined as follows:

$$\varphi_1 = \frac{4 \cdot Q_1}{\pi \cdot D_1^2 \cdot U_1}$$

where Q_1 is the volumetric flow rate and D_1 and U_1 are the inlet tip impeller diameter and velocity respectively.

The second coefficient is the load factor τ defined as the total enthalpy rise across the impeller divided by the inlet peripheral blade velocity:

$$\tau = \frac{\Delta H}{U_1^2} = \frac{Cp \cdot (T_{02} - T_{01})}{U_1^2}$$

The third performance parameter considered for our optimisation procedure is the polytropic efficiency η_p:

$$\eta_p = \frac{K-1}{K} \cdot \frac{\log\left(\frac{P_{02}}{P_{01}}\right)}{\log\left(\frac{T_{02}}{T_{01}}\right)}$$

These coefficients were sufficient to characterise the impeller performance. Below it will be shown how, combining these parameters into a target function, it was possible to perform an optimisation procedure which provided a new impeller characterised by a higher efficiency at the same flow coefficient, rotational speed and pressure ratio.

COMPUTATIONS AND RESULTS

The program has been tested on a two-dimensional impeller whose meridional view is sketched in Figure 3. This is a low flow coefficient shrouded impeller with seventeen cylindrical backswept blades. As it can be seen in figure the impeller is characterised by totally radial blades while the axial part of the inlet channel is vaneless.

The objective function was stated in order to maximise the polytropic efficiency keeping the flow coefficient and the polytropic head as close as possible to the initial (design) ones. The target function is then defined as:

$$f=-(\eta_p)^2+10((\tau\eta_p)_{des}-\tau\eta_p)^2$$

where $(\tau\eta_p)_{des}$ represents the required polytropic head. Since the function f has to be minimised, it has been defined in order to increase significantly each time the computed polytropic head differs from the required one.

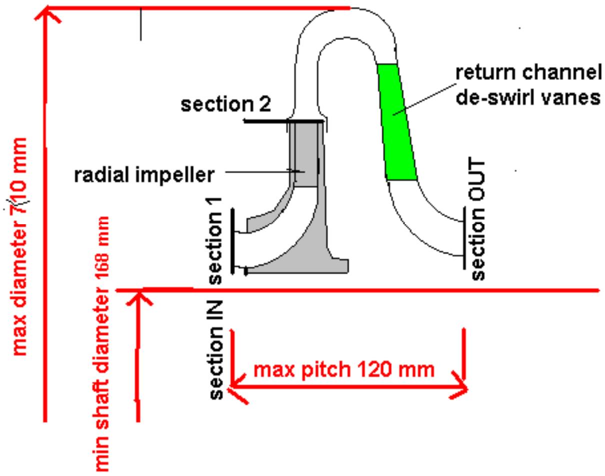


Figure 3. Meridional view of the test impeller.

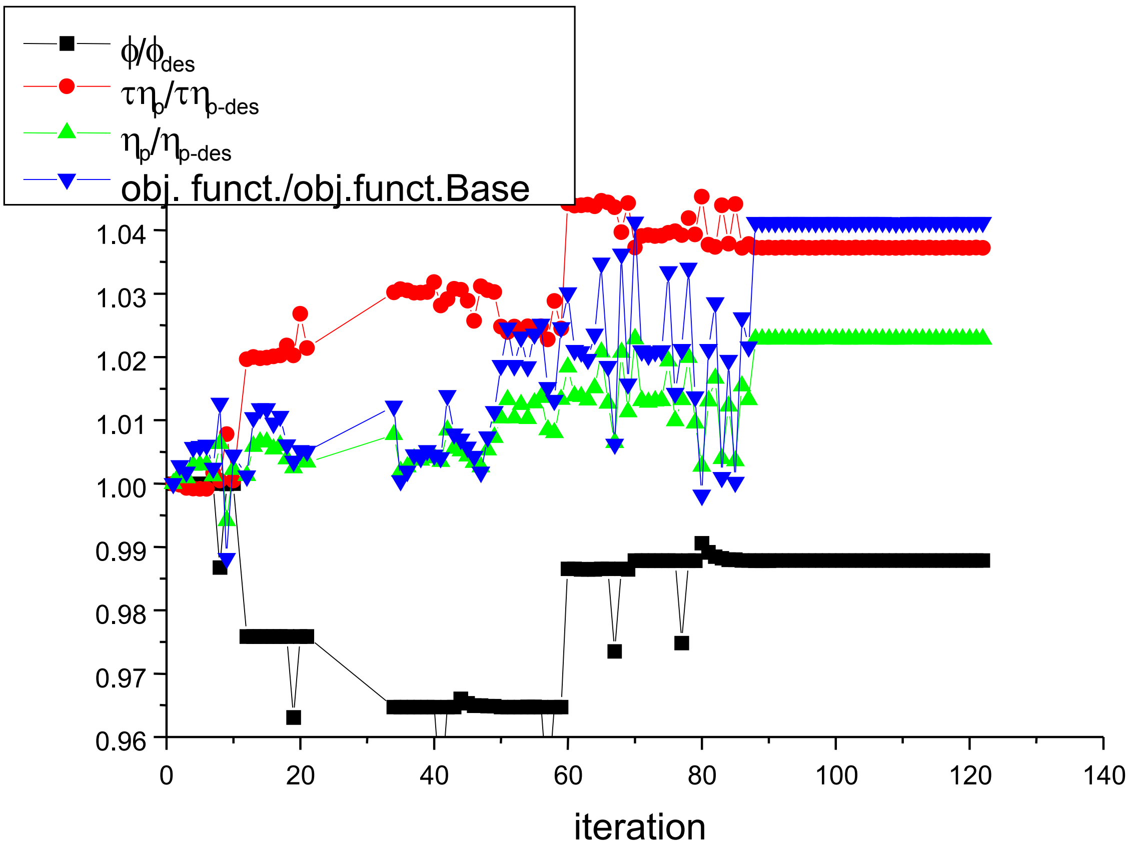


Figure 4. Convergence history for the test impeller optimisation.

The computations are performed imposing a non-dimensional inlet flow velocity and the impeller rotational speed. The mesh used has 89 nodes in the streamwise direction, 69 nodes in the blade to blade direction and 51 in the spanwise direction.

For the performed test only 8 out of the whole 25 geometrical parameters were left free to change during the optimisation process. These 8 parameters are the ones which control the shape of the impeller in the meridional plane, thus letting unaltered the blade curvature in the s-θ plane.

Figure 4 shows the convergence history of the performed test. The values of the performance indices reported are normalised with respect to their value for the base impeller. It can be seen that the flow coefficient and the polytropic head (black squares and red dots respectively) are varying within a range of 4% with respect to their initial value, thus confirming the correct choice of the objective function. The diagram shows how the polytropic efficiency of the impeller (green triangles) improves of 2% with respect to the initial value in about 90 computations including function and gradient evaluations for an amount of about 100 hours of CPU time.

Figure 5 shows a comparison between the original meridional channel (black line) and the optimised one (red line). The new geometry is characterised by a smoother curvature radius in the axial-to-radial bend of the impeller. Also the exit radius and blade height are slightly increased with respect to the initial ones.

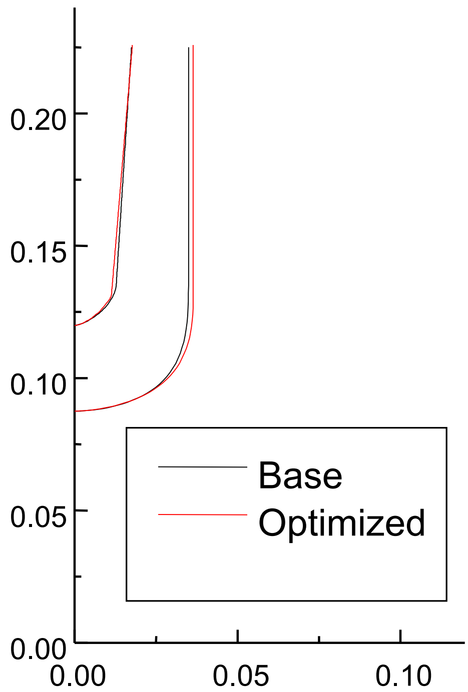


Figure 5. Comparison between base (black) and optimised (red) impeller shapes on the meridional plane.

Figure 6 shows the trend of two performance indices (load factor and polytropic efficiency) along the meridional channel curvilinear abscissa between leading and trailing edge. As for the following plots, for each value of the s co-ordinate (see Figure 2) the variables are mass averaged on a section perpendicular to the curvilinear abscissa itself. These first two plots show how the optimised impeller presents a much higher polytropic efficiency already close to the impeller leading edge (about 40% of the meridional length). This is due to the lower losses taking place in the upstream axial to radial bend of the impeller thanks to the modifications to the wall curvature radius with respect to the base geometry. Along the blade the efficiency distribution for the optimised configuration remains almost constant while the base impeller presents a sudden rise of the η_p value just downstream the leading edge and then an almost flat trend from 55 to 100% of the meridional length. In the right part of the figure the load factor distributions for the two impellers are almost parallel, with the optimised one which is somehow translated to slightly lower values with respect to the base one. Thanks to the lower loading on the optimised blade it was possible to increase the final polytropic efficiency keeping the same head value of the base geometry.

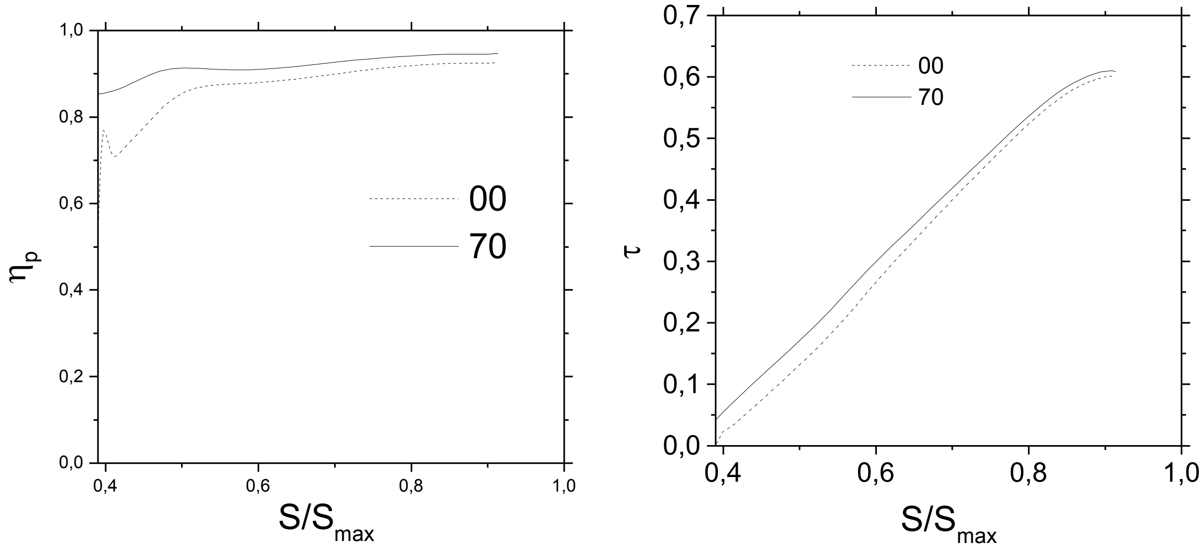


Figure 6. Polytropic efficiency (left) and load factor (right) distributions along the meridional curvilinear abscissa for the base (00) and optimised (70) impeller.

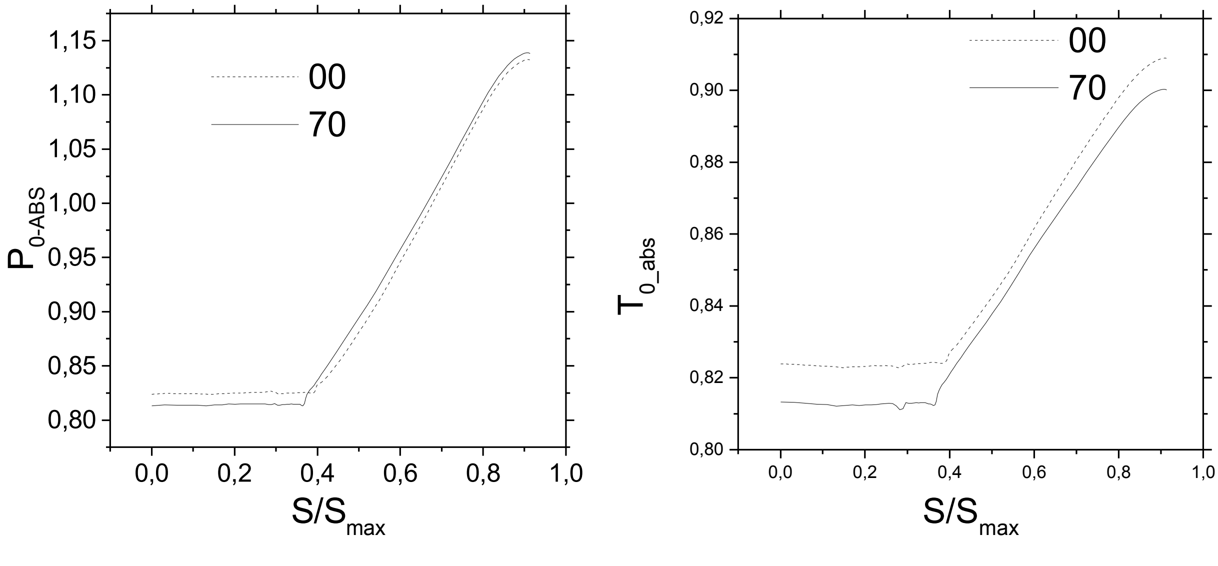
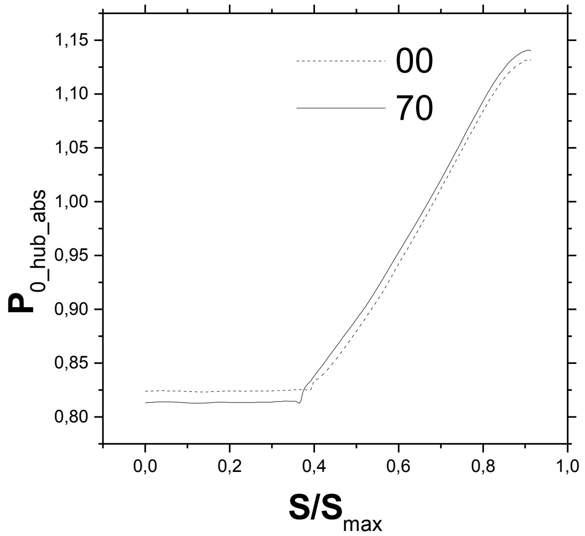


Figure 7. Absolute total pressure (left) and total temperature (right) distributions along the meridional curvilinear abscissa for the base (00) and optimised (70) impeller.

Figure 7 shows how the total temperature distributions for the two impellers are quite similar, with the optimised one translated to lower values with respect to the base one. The main contribution to the polytropic efficiency increase for the optimised impeller comes from the total pressure distribution, as shown in the left plot of the same figure. Here it can be seen that, close to the blade leading edge (about 40% of meridional length) both impellers present a sudden total pressure rise which is remarkably higher for the optimised configuration. From 40 to 100% of the s abscissa both the distributions remain parallel. The total pressure trend is quite similar, for both impellers, close to hub and to shroud walls (see Figure 8) although close to shroud the curve slope for the optimised impeller is slightly worse than the one corresponding to the base geometry. The result, anyway, consists in a higher exit to inlet ratio of total pressure for the optimised geometry leading, from the definition itself of the performance index, to a higher value of the polytropic efficiency with respect to the base impeller.



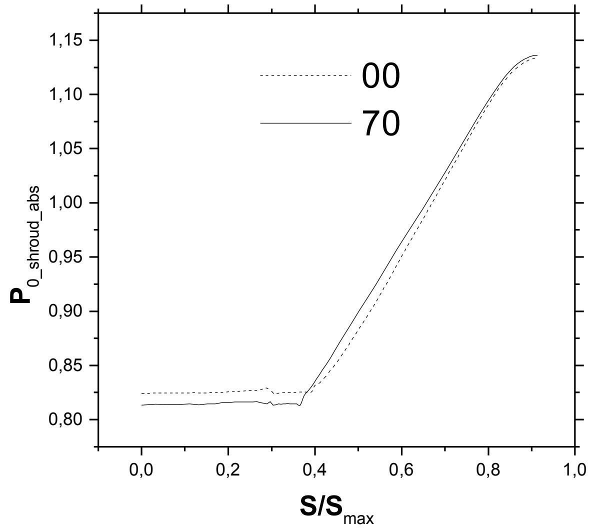


Figure 8. Absolute total pressure close to hub (left) and shroud (right) distributions along the meridional curvilinear abscissa for the base (00) and optimised (70) impeller.

CONCLUSIONS

An optimisation procedure based on a FSQP search algorithm coupled to a Lazy Learning approximator exploiting a structured CFD code has been applied to a test radial impeller for centrifugal compressors. Eight geometrical parameters controlling the shape of the meridional channel, among the 25 necessary to completely describe the blade, have been utilised as free variables for the optimisation process. The target function has been defined in order to improve the compressor's efficiency keeping almost constant the flow coefficient and the polytropic head.

In less than hundred computations, including function and gradient evaluations, the program has been able to increase of 2% the impeller polytropic efficiency keeping the inlet flow coefficient and the polytropic within an acceptable range. The new impeller is characterised by a smoother curvature radius in the axial-to-radial bend of the impeller. Also the exit radius and blade height are slightly increased with respect to the initial ones.

The preliminary results shown in this work highlight the capacity of the algorithm to perform an optimisation procedure in a fast and reliable way without any kind of human intervention, if we except the definition of the target function. Hence the procedure meets the requirements on part of industry of an instrument able to automatically design a component without the necessity to employ highly specialised personnel.

ACKNOWLEDGEMENTS

The authors wish to thank the EC for supporting the project to which this work belongs and for having allowed the publication of this paper.

All the partners of the METHOD EC project are gratefully acknowledged for their help.

Miss Michela Testa is thanked for the accurate results analysis.

REFERENCES

[1] Aha D.W., Editorial, *"Artificial Intelligence Review, Special Issue on Lazy Learning"*, 1997.

[2] Atkeson, C.G., Moore, A.W., Schaal, S., *"Locally weighted learning"*, Artificial Intelligence Review, 11(1-5), 11-73, 1997.

[3] Birattari, M., Bontempi, G., Bersini, H., *"Lazy learning meets the recursive least squares algorithm"*, in M.S. Kearns, S.A. Solla, and D.A. Cohn (Eds.) NIPS'98: Advances in Neural Information Processing Systems 11. pp. 375-381. MIT Press, Cambridge, MA, USA. (Conference held in Denver, CO, USA. December 1-3, 1998)

[4] Cosentino, R., Alsalihi, Z., Van Den Braembussche, R., *"Expert System for Radial Impeller Optimisation"*, ATI-CST-039/01, Fourth European Conference on Turbomachinery, March 2001, Florence, Italy.

[5] Durbin, P.A., *"On the k-ω Stagnation Point Anomaly"*. To appear in the Int. J. Heat and Fluid Flow, 1997.

[6] Fletcher, R., *"Practical Methods of Optimisation"*, Wiley, 2000.

[7] Goldberg, D.E., *"Genetic Algorithms in Search, Optimization and Machine Learning"*, 1989, Addison-Wesley.

[8] Michelassi, V., Belardini, E., *"Numerical Simulation of Three-Dimensional Inlet Guide Vanes"*, Proceedings, 3-rd European Conference on Turbomachinery Fluid Dynamics and Thermodynamics, 2-5 March 1999, London (UK).

[9] Michelassi, V., Pazzi, S., Echtner, S., Giangiacomo, P., Martelli, F., Giachi, M., *"Performances of Centrifugal Compressor Impellers in Steady and Unsteady Flow Regimes under Inlet Flow Distortions"*, ASME-2001-GT-0325.

[10] Mosetti, G., Poloni, C., (1993), *"Aerodynamic Shape Optimisation by means of a Genetic Algorithm"*, 5th International Symposium on Computational Fluid Dynamics, Sendai, Japan.

[11] Poloni, C., *"Multi Objective Optimisation Examples: Design of a Laminar Airfoil and of a Composite Rectangular Wing"*, Genetic Algorithms for Optimisation in Aeronautics and Turbomachinery, von Karman Institute for Fluid Dynamics Lecture Series 2000.

[12] Pulliam, T.H., Chaussée, D.S., *"A Diagonal Form of an Implicit Approximate Factorization Algorithm"*, Journal of Computational Physics, n° 39, 1981.

[13] Quagliarella, D., Della Cioppa, A., (1994), *"Genetic Algorithms Applied to the Aerodynamic Design of Transonic Airfoils"*, AIAA-94-1896, 12th AIAA Applied Aerodynamic Conference, Colorado Springs, CO USA.

[14] Wilcox, D.C., *"Reassessment of the Scale-Determining Equation for Advanced Turbulence Models"*, AIAA Journal Vol. 26, 1988.