Design Optimization International Conference,
March 31- April 2 2004, Athens, Greece

STOCHASTIC AND DETERMINISTIC OPTIMIZATION FOR TURBOMACHINERY APPLICATIONS BASED ON THE ADJOINT FORMULATION

Dimitrios I. Papadimitriou[⋆], **Ioannis C. Kampolis**[⋆], **Kyriakos C. Giannakoglou**[⋆]

[⋆]National Technical University of Athens,
Lab. of Thermal Turbomachines,
P.O. Box 64069, Athens 157 10, GREECE,
e-mail:design04@mail.ntua.gr,

Key words: Design, Optimization, Evolutionary Algorithms, Adjoint Formulation, Aeronautics, Turbomachinery, Fluid Mechanics

Abstract. *This paper investigates strategies for coupling adjoint methods, evolutionary algorithms and artificial neural networks, in the context of shape optimization or inverse design problems in the turbomachinery field. In the method this paper is mainly dealing with, the three aforementioned components take on discrete roles. The evolutionary algorithm is in the leading role, acting as the search method, seeking for the optimal solution. Surrogate models, in the form of new radial basis function networks that have been recently proposed by the authors, are employed for the (approximate) evaluation of candidate solutions. The adjoint equations solver computes the gradient of the objective function at a number of design points which, together with the objective function values at the same points, are used to train the surrogate evaluation model. On– and off–line versions of the aforementioned synthetic optimization method are presented and discussed. Besides, optimization using EAs with standard surrogate models, acting at the so–called Inexact Pre–Evaluation phase (see previous works by the authors) as well as stand–alone adjoint methods are shown. Two inverse design problems for turbine and compressor blades are used for the demonstration of the proposed methods.*

1 INTRODUCTION

Pros and cons of the most widely used shape optimization and inverse design methods for airfoils, wings or turbomachinery blades, such as Adjoint Methods (AMs) and Evolutionary Algorithms (EAs), are well known. Formulating and solving the adjoint to the flow problem equations, in order to compute functional gradients that will then guide any ascent or descent search algorithm, is an elaborate and often cumbersome task. The inability of AMs to cope with the so-called inadmissible objectives and the risk of being trapped in local rather than global optima, depending on the starting search point, are their major weaknesses. In contrast, the indisputable advantage of AMs is that they are much less computationally expensive than other rival optimization methods. On the other hand, the implementation of EAs with any CFD analysis software and any objective function is straightforward; however, they are blamed for high computing cost due to the great number of objective function evaluations (herein, many calls to the CFD tool) they usually need. A known remedy to the latter is to replace as many exact and costly CFD evaluations as possible by less expensive, approximate evaluations, through surrogate modeling. This can be done in different manners, as this will be discussed below.

Our group started working on the development and use of surrogate models long ago, [1],[2]. In [3], the authors proposed the use of EAs with local surrogate evaluation models, constructed separately for each candidate solution by exploiting previously evaluated individuals. Surrogate modeling through Artificial Neural Networks (ANNs) and, in particular, Radial Basis Function (RBF) networks was preferably used. Though, in the past, the same group successfully tried other surrogate modeling techniques such as multilayer perceptrons, [1], or kriging methods, [4], there are good reasons for working with RBF networks. These reasons will become clear as the paper develops. It is not difficult to differentiate between the method proposed in [3] and other contemporary methods since the latter usually construct a single surrogate model for the entire search space, prior to the evolution. Detailed comments on the above methods will be given below, where different algorithmic variants will be used for the purpose of comparison.

Among the optimization strategies discussed in this paper, emphasis will be given to the one which brings together AMs, EAs and RBF networks. The new method was first proposed in [5], where this was tested in single– and multi–objective problems; here, this method is applied to turbomachinery inverse design problems and compared with other optimization algorithms. The key features of the new method are listed below: (a) the search for the optimal solution is carried out through EAs, (b) for the evaluation of candidate solutions, the EAs use exclusively RBF networks, (c) the RBF networks have a non-conventional formulation, constructed separately using a number of points in the search space for which the objective function value and its gradient have been computed, (d) the adjoint equations solver undertakes the computation of the objective function gradient and (e) a small number of cycles is needed for locating the optimal solution. The computationally expensive part of the method is the evaluation of the training patterns and the computation of functional gradients. The EA cost is negligible since it relies upon the computationally inexpensive surrogate model. In what follows, we will refer to the new surrogate model as Gradient-Assisted RBF networks, abbreviated to GARBF, [6].

Both cases examined are 3D inverse design problems, governed by the inviscid flow equations. The analysis (CFD) tool is a 3D Euler equations solver for structured grids, based on the finite-volume technique and a time-marching upwind formulation, employing the Roe's approximate Riemann solver [7] with variables extrapolation to account for second-order accuracy.

2 EAs WITH SURROGATE MODELING

Nowadays, EAs with cheap surrogate evaluation models (known, also, as "metamodels") are in widespread use. Regardless of how surrogate modeling is employed, the concept is the same: create and use inexact, inexpensive models instead of the costly, exact evaluation tool. Methods proposed in the literature can be categorized depending on: (a) the type of metamodel used, (b) the definition space of the metamodel (global or local one), (c) the way training data have been collected and (d) the strategy for coupling EAs with the surrogate model.

A first important classification of metamodel–assisted EAs is to those using *on–line* or *off–*

line trained surrogate models. In EAs with off–line trained metamodels, the latter are built at the initial stage of the optimization and, for this purpose, more or less sophisticated Design–of–Experiment (DoE) techniques are used. Under the DoE abbreviation one might consider even grid techniques or the generation of the training set at random. In multivariate problems, the former might lead to huge (2^d) training sets whereas the latter is often inadequate. The evaluation of training patterns takes place separately from the EA and is the computationally expensive part of the algorithm. The EA seeks for the optimal solution using the previously trained, global surrogate model. The almost negligible cost of the evaluation makes the optimization phase extremely fast. The outcome of the optimization is usually referred to as the "optimal" solution (the quotation marks show that this solution was evaluated using the metamodel). The "optimal" solution needs to be exactly evaluated and, depending on a termination criterion, this point is added to the training set. Other points can be added too, according to various selection schemes. The surrogate model is built anew (full or incremental training can be used) and the EA–based optimization is repeated. This algorithm is illustrated in fig. 2, where the inner cycle is the metamodel assisted EA that computes the "optimal" solution.

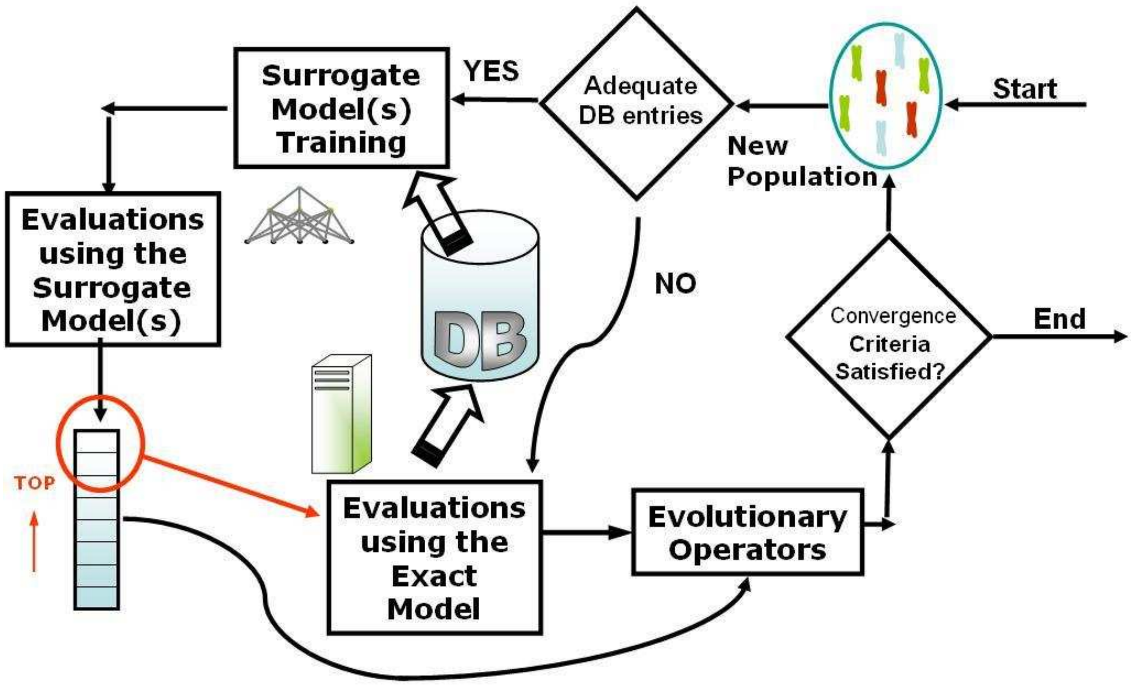
Surrogate Model(s) Training
YES
Adequate DB entries
Start
New Population
NO
Evaluations using the Surrogate Model(s)
DB
Convergence Criteria Satisfied?
End
TOP
Evaluations using the Exact Model
Evolutionary Operators

Figure 1: EAs coupled with on-line trained surrogate models.

Optimization based on EAs supported by on–line trained surrogate models, fig. 1, has been proposed by the third author [1]. Recently, similar works appeared in the literature, [8],[9],[10].

Distinguishing figures of all of them are: (a) the construction of a different metamodel valid in the vicinity of each new candidate solution, (b) the use of a subset only of the available patterns for the metamodel training and (c) the way the approximate fitness values computed by the surrogate models are used by the EA.

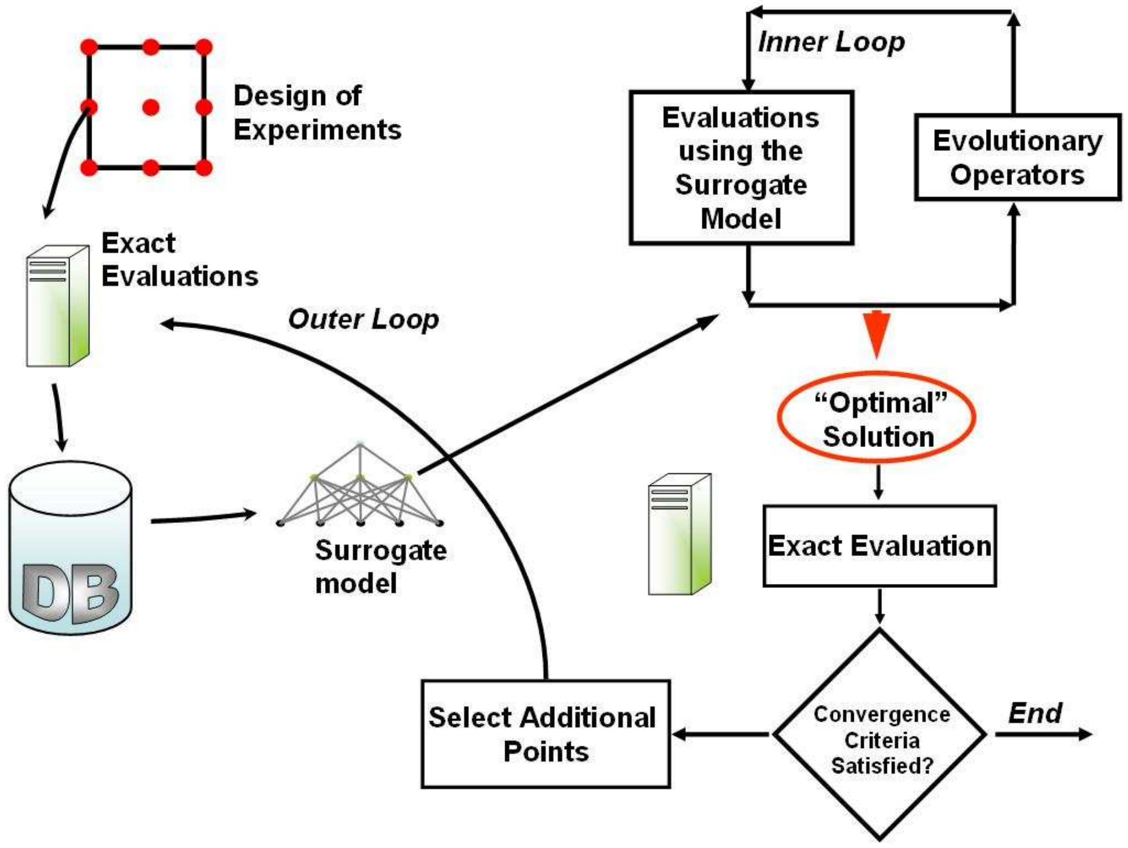

Figure 2: EAs coupled with off-line trained surrogate models.

At this point, we should make clean that the terms *on–line training* and *local metamodeling* are not necessarily identical. For instance, the algorithm illustrated in fig. 2 can be used with local metamodels by merely subdividing the search space into subspaces and associating each one of them with a different local metamodel. However, this seems to be of minor interest. In what follows, on–line trained metamodels supporting EAs will necessarily be local ones. In the sake of fairness, let us note that the on–line metamodel used in [1] was global.

In EAs assisted by on–line trained metamodels, at each generation of the EA and for each individual separately, the neighboring entries in the training set are selected for the metamodel training. Practically, as many metamodels as the population size are needed. However, the total training cost is low since, each training sub–set is small. Each new population member is evaluated using its own metamodel and the population is rank sorted using approximate fitness values. Then, the top small percentage of the population is evaluated using the exact evaluation tool. By doing so, the CPU cost per generation is considerably cut down. Practically, the algorithm just described (shown in fig. 1) trains and uses local metamodels for screening out less promising evaluations and restricts exact evaluations only to the most promising among them.

In [5], the authors proposed to use new RBF networks trained on both objective function values and gradients. To accommodate the additional information, the mathematical formulation of standard RBF networks should be changed by introducing new tuning parameters. The adjoint equations solver is used sequentially after the flow equations solver for each candidate solution. Since the CPU cost of the adjoint equations' solver is almost equal to the CPU cost of the flow equations, the new algorithm has twice as much computing cost per individual. On the other hand, the trained network is expected to be much more accurate than the one constructed using exclusively known responses. Or, for the same prediction error, much less training patterns are needed.

It should become clear that figs 1 and 2 may also describe the new method. It suffices to replace *Evaluations using the Exact Model* with *Evaluations and Gradient Computation using the Exact Model and the Adjoint System.*

3 THE ADJOINT FORMULATION

This section presents the continuous adjoint method developed for the calculation of the objective function gradient, [11]. The development was based on an existing Euler equations solver for rotating cascades with constant rotational speed ω, as briefly described in the introduction. Starting point for the formulation of the adjoint equations is the functional which should be minimized. Dealing with inverse design problems, where the target is to capture a given pressure distribution p_{tar} over the blade surface (w), the objective function reads,

$$min\ F \ = \ \iint_{(w)} (p - p_{tar})^2 ds \tag{1}$$

On a Cartesian (x,y,z) frame, where z coincides with the rotation axis, the Euler equations are written in vector form as

$$\frac{\partial \overrightarrow{U}}{\partial t} + \frac{\partial \overrightarrow{f}}{\partial x} + \frac{\partial \overrightarrow{g}}{\partial y} + \frac{\partial \overrightarrow{j}}{\partial z} + \overrightarrow{q} \ = \ \overrightarrow{0} \tag{2}$$

with $\overrightarrow{U} = [\rho\ \rho u\ \rho v\ \rho w\ E]^T$, $E = \rho e + \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right)$ being the total energy per unit volume. For the sake of completeness, we just repeat the source term

$$\overrightarrow{q} = \Big[0, -\rho(2\omega v + \omega^2 x), \rho(2\omega u - \omega^2 y), 0, 0\Big]^T \tag{3}$$

By multiplying the Euler equations with the adjoint variables $\overrightarrow{\Psi}$ and integrating over the flow domain, the augmented cost functional F_{aug} is defined as

$$F_{aug} = F - \iiint_{\Omega} \overrightarrow{\Psi}^T \left(\frac{\partial \overrightarrow{f}}{\partial x} + \frac{\partial \overrightarrow{g}}{\partial y} + \frac{\partial \overrightarrow{j}}{\partial z} + \overrightarrow{q} \right) d\Omega \tag{4}$$

After some rearrangements, the variation of the augmented cost functional gives

$$\begin{aligned}
\delta F_{aug} &= \underbrace{\frac{1}{2}\iint_{(w)}(p-p_{tar})^2\delta(ds)}_{S.D.} + \underbrace{\iint_{(w)}(p-p_{tar})\delta p ds}_{C.R.} \\
&+ \underbrace{\iiint_{\Omega}\delta\overrightarrow{U}\left(A\frac{\partial\overrightarrow{\Psi}}{\partial x}+B\frac{\partial\overrightarrow{\Psi}}{\partial y}+C\frac{\partial\overrightarrow{\Psi}}{\partial z}-\overrightarrow{q}_a\right)^T d\Omega}_{A.E.} \\
&- \underbrace{\iint_{(in)}\overrightarrow{\Psi}^T A_n\delta\overrightarrow{U}ds - \iint_{(out)}\overrightarrow{\Psi}^T A_n\delta\overrightarrow{U}ds}_{A.E.B.C} \\
&- \underbrace{\iint_{(w)}\overrightarrow{\Psi}^T(\delta\overrightarrow{f}n_x+\delta\overrightarrow{g}n_y+\delta\overrightarrow{j}n_z)ds}_{C.R.} \\
&- \underbrace{\iint_{(w)}\overrightarrow{\Psi}^T(\overrightarrow{f}\delta(n_xds)+\overrightarrow{g}\delta(n_yds)+\overrightarrow{j}\delta(n_zds))}_{S.D} \qquad (5)
\end{aligned}$$

where

$$\overrightarrow{q}_a = \Big[0,-(2\omega\Psi_3+\omega^2x\Psi_1),(2\omega\Psi_2-\omega^2y\Psi_1),0,0\Big]^T \qquad (6)$$

and (n_x,n_y,n_z) are the components of the outward normal to the domain boundary vector. Flow variable variations are eliminated by enforcing $\overrightarrow{\Psi}$ to satisfy the field adjoint equations

$$\frac{\partial\overrightarrow{\Psi}}{\partial t}-A\frac{\partial\overrightarrow{\Psi}}{\partial x}-B\frac{\partial\overrightarrow{\Psi}}{\partial y}-C\frac{\partial\overrightarrow{\Psi}}{\partial z}+\overrightarrow{q}_a = \overrightarrow{0} \qquad (7)$$

the boundary conditions for the inlet/outlet

$$\delta\overrightarrow{U}^T(A_n{}^T\overrightarrow{\Psi}) = \overrightarrow{0} \qquad (8)$$

and the compatibility relations over the blade surface

$$(p-p_{tar})-(\Psi_2n_x+\Psi_3n_y+\Psi_4n_z) = 0 \qquad (9)$$

What remain in δF_{aug}, are the terms depending on the variation of geometrical quantities

$$\begin{aligned}
\delta F_{aug} &= \frac{1}{2}\iint_{(w)}(p-p_{tar})^2\delta(ds) \\
&+ \iint_{(w)}\overrightarrow{\Psi}^T(\overrightarrow{f}\delta(n_xds)+\overrightarrow{g}\delta(n_yds)+\overrightarrow{j}\delta(n_zds)) \qquad (10)
\end{aligned}$$

Further rearrangement of the terms appearing in eq. 10 depends on the blade parameterization. Though eq. 2 was written in the Cartesian frame, the control points are defined using cylindrical coordinates. The reason for doing this is that, quite often, the (z_i, r_i) coordinates for each control point are kept constant and only their peripheral position $r\theta_i$ is allowed to vary. Here, a fixed grid of control points on the (z_i, r_i) plane is assumed, which corresponds to their projection on the meridional plane.

The gradient computed using the adjoint technique is employed in an iterative steepest descent method updating the control variables

$$\overrightarrow{b}^{n+1} \; = \; \overrightarrow{b}^{n} \; - \; \eta \nabla F(\overrightarrow{b}^{n}) \tag{11}$$

With different targets, such as a desirable velocity distribution V_{tar} on (w), different compatibility relations can be derived, namely

$$(V - V_{tar}) + \rho V(\Psi_2 n_x + \Psi_3 n_y + \Psi_4 n_z) \; = \; 0 \tag{12}$$

by assuming that $\delta p \; = \; -\rho V \delta V$.

4 THE NEW GARBF NETWORK

In the new *GARBF* network [5], the response is given by

$$\zeta \; = \; \sum_{i=1}^{N} (b_i \; + \; \sum_{m=1}^{M} a_{i,m}[1 + (x_m - c_{i,m})]) \; exp(-\sigma_i) \tag{13}$$

where

$$\sigma_i = M \sum_{m=1}^{M} I_{i,m}(x_m - c_{i,m})^2 \tag{14}$$

The N coefficients b_i and the $N \times M$ coefficients $a_{i,m}$ are the unknown parameters that should be computed using responses and M gradient components known at the M training patterns.

5 METHOD APPLICATIONS AND CONCLUSIONS

In this paper, we first focused on evolutionary and deterministic optimization methods and, then, on their combined use (hybridization), in particular those supported by new surrogate models. Two turbomachinery inverse design problems have been analyzed and will be demonstrated below using four of the proposed algorithms. In the comparisons which follow, we do not intend to the direct (quantitative) comparison of the four methods. Such a comparison might lead to conclusions that are strongly case–dependent. For instance, the pure steepest descent method (driven by the adjoint technique) is expected to be (and is, in fact) faster than the evolutionary tools but its success depends on the starting point. Also, any meaningful conclusion about the performance of EAs using off–line trained metamodels depends on the patterns used to train the metamodel (this important issue is beyond the scope of this paper).

5.1 Inverse Design of 3D Turbine Peripheral Cascade

For the inverse design of a turbine peripheral cascade blade, pressure distributions over the pressure and suction surfaces of an existing blade were first computed and then used as targets.

The target shape was built using the same parameterization, through two Bezier surfaces with 33 control points on each of them. Both pressure and suction surfaces used 3×11 control points (in the radial and axial direction, respectively), as shown in fig. 3. Among them, only the $r\theta$ coordinates of the control points per blade side were allowed to vary.

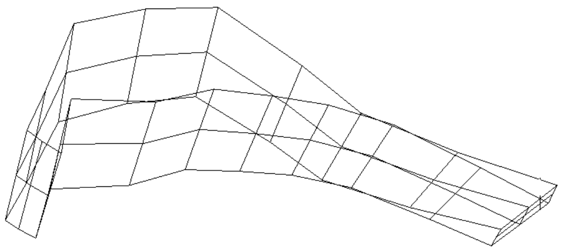

Figure 3: 3D Turbine Peripheral Cascade: Parameterization of the pressure and suction blade surfaces using Bezier surfaces. A total number of $3 \times 11 = 33$ control points are defined on each surface.

The evaluation tool and the adjoint equations solver were based on the Euler equations; for each blade–candidate solution, a 3D structured grid of about 12000 nodes was generated before solving the flow equations on it. The cost for the grid generation and numerical solution of the flow equations was about 5 minutes on an Intel P4 2.4GHz processor. The cost for solving the adjoint equations is considered to be something less than 5 minutes; however, in what follows, direct and adjoint equations solvers will be considered to be equally expensive; the cost for each one of them will be referred to as the cost for an "equivalent flow solution".

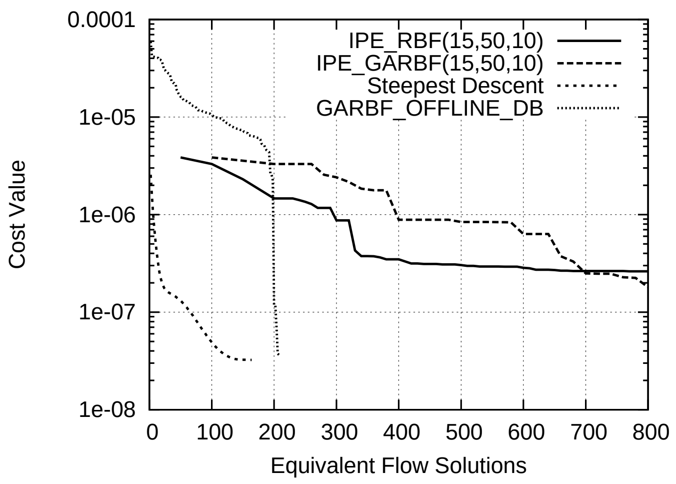

Figure 4: Inverse Design of a 3D Turbine Peripheral Cascade: Convergence of the four optimization methods.

Fig. 4 summarizes the convergence behaviour of the four tools used. The pure steepest descent method, driven by the objective function gradient as computed by the adjoint method yields a monotone convergence within 80 cycles (i.e. 160 equivalent flow solutions). A decrease of about two orders of magnitude is observed. The second design method was EA supported by off–line trained GARBF. Its convergence plot is marked by GARBF–OFFLINE–DB. To initiate the run, a database with 100 solutions, randomly selected within the search space, was formed. For these training patterns, objective function values and gradient components have been computed; the corresponding cost was about 200 equivalent flow solutions. In the convergence plot, the 200 first units in the abscissa have been paired with the rank sorted cost function values of the database entries. As previously described, this method requires a small number of outer loops for the cross–checking of the GARBF–based evaluations at the so–called "optimal" solutions. Each time the "optimal" solution was exactly evaluated, this point was added to the database together with one more point, placed at the most unexplored area of the search space. A simple algorithm was devised to locate this point using a simplified distance–based criterion and a non–time–consuming search tool. The last two methods used, plotted also in fig. 4, are EAs using the IPE technique with conventional RBF and gradient–assisted RBF networks as the inexact pre–evaluation tool, i.e. the surrogate model. These are marked by IPE–RBF and IPE–GARBF, respectively. The population size was $\mu = 15$ and $\lambda = 50$. A two order of magnitude convergence was obtained in about 500 equivalent flow solutions with both tools. However, it should be noted that this CPU cost corresponds to 500 evaluated individuals in IPE–RBF and only half of them in IPE–GARBF. The different correspondence between number of evaluations and CPU cost (or equivalent flow solutions), is reflected upon the convergence. A careful examination of the convergence behaviour shows that IPE–GARBF is more costly during the starting phase, since a minimum number of individuals should be evaluated for building a database being representative of the search space. But, later on, thanks to the additional information (gradient) the network is using, better convergence for the IPE–GARBF is observed.

5.2 Inverse Design of 3D Compressor Peripheral Cascade

The second case examined is that of the inverse design of a compressor rotor blade. The corresponding stator blade was analysed by the authors in [5]. Here, the blade surfaces were parameterized using two Bezier surfaces which meet along the leading and trailing edges. Each blade side was parameterized using $4 \times 5 = 20$ control points (4 in the radial and 5 in the axial direction). Only 6 per surface out of them were allowed to vary. A view of control point

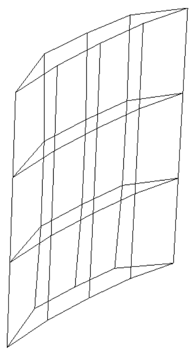

Figure 5: 3D Compressor Peripheral Cascade: Parameterization of the pressure and suction blade surfaces using Bezier surfaces. $5 \times 4 = 20$ control points were defined on each surface.

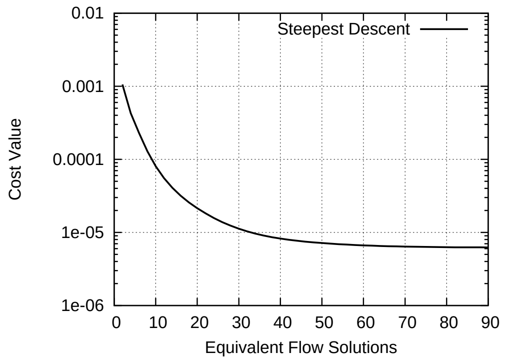

Figure 6: 3D Compressor Peripheral Cascade: Convergence history of the steepest–descent method supported by the adjoint method for the objective function gradient computation.

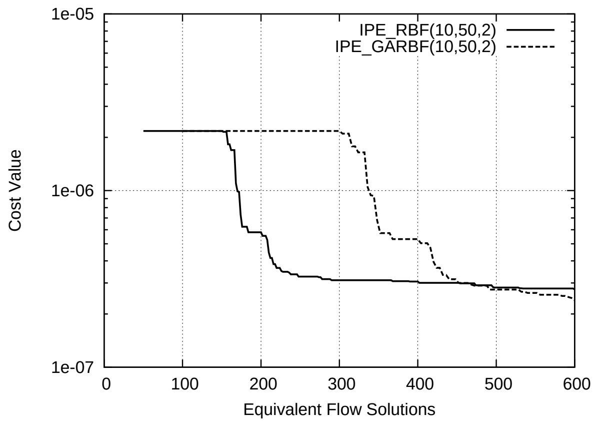

Figure 7: 3D Compressor Peripheral Cascade: Convergence history of two EAs, both using the IPE technique: Conventional RBF and Gradient–Assisted RBF networks were used as surrogate models, in each one of them.

locations is shown in fig. 5.

The convergence plot of the adjoint–steepest descent method is show in fig. 6. The final cost value is not that low as in the EA–based designs (fig. 7) due to the step size η values which did not allow the refinement of the finally computed solution. The off–line trained GARBF network converged in about 250 equivalent flow solutions, starting with a database with 100 entries (as in the turbine problem). Its convergence is not included in these figures. Fig. 7 shows the convergence plots for EAs with IPE–RBF and IPE–GARBF, used with $(\mu, \lambda) = (10, 50)$ population sizes. Like the turbine case, the IPE–GARBF needs twice as many equivalent flow solutions to start up; later, however, it becomes faster convergent, for the reasons already exposed in the previous case.

The initial, target and converged $Mach$ number distribution over the turbine and compressor blades (computed through steepest descent) are shown in fig. 8.

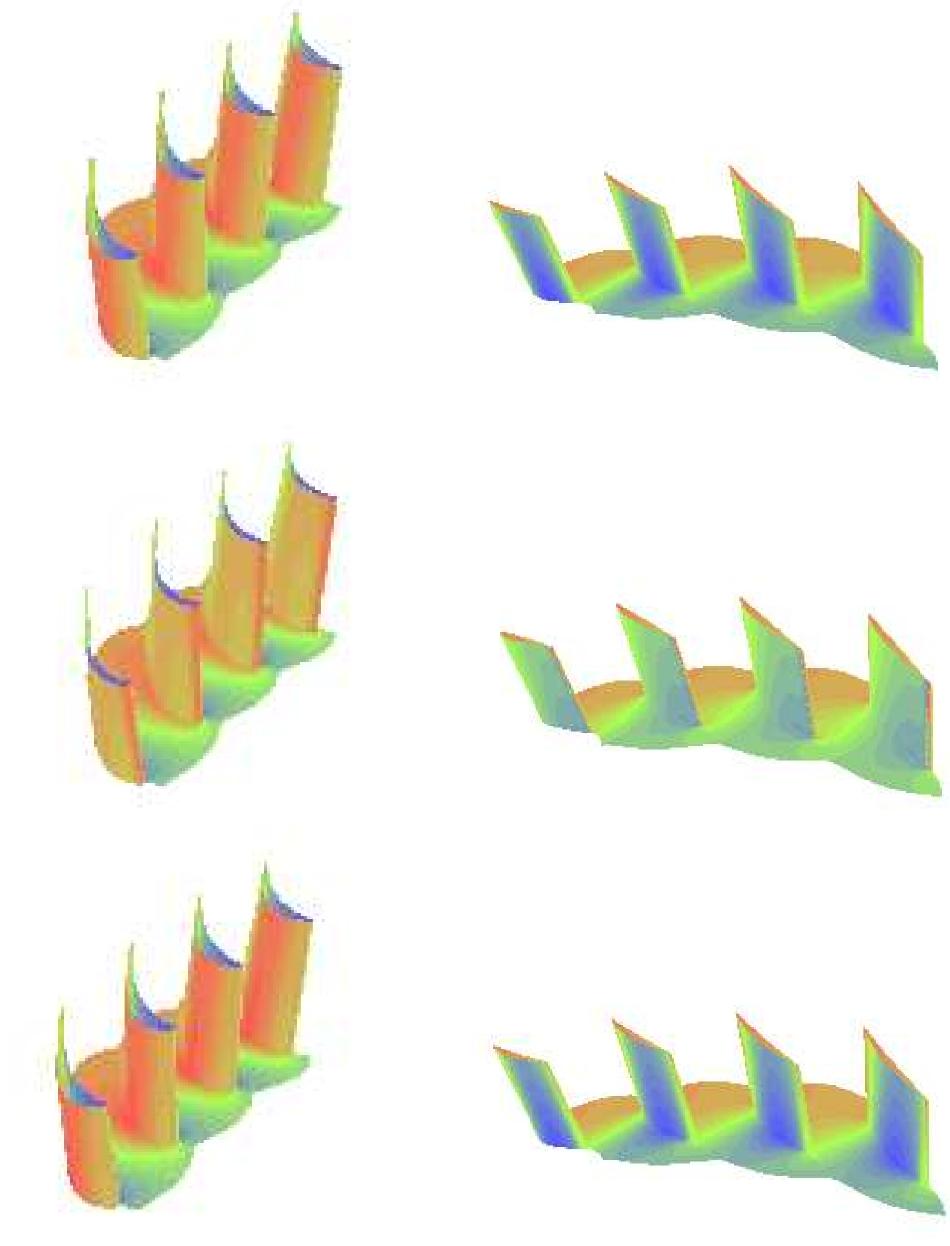

Figure 8: Inverse Design of 3D Turbomachinery Bladings (Left:Turbine Stator, Right: Compressor Rotor). Mach number distribution over the cascade. Top: Target, Center: Initial, Bottom: Converged solutions (using steepest descent).

5.3 Conclusions–Discussion

The above studies didn't intend to lead to definite conclusions about the superiority of any method. We rather intended to make clear that the objective function gradient is an information which can be exploited in many different ways (other than the standard descent algorithms). EAs with the Inexact Pre–Evaluation (IPE) technique can be driven by gradient–assisted metamodels (such as GARBF networks). The capabilities of the new algorithms, which bring together many traditionally "rival" methods (EAs, adjoint methods, computational intelligence), should be further investigated using more test problems with complex, multimodal solution landscapes.

Support from Program Thales of the National Technical University of Athens is acknowledged.

REFERENCES

[1] Giannakoglou, K.C., Designing Turbomachinery Blades Using Evolutionary Methods, ASME Paper 99-GT-181, 44th ASME Gas Turbine and Aeroengine Congress, Indianapolis, IN, USA, June 7-10, 1999.

[2] Giannakoglou, K.C., Design of Optimal Aerodynamic Shapes using Stochastic Optimization Methods and Computational Intelligence, Progress in Aerospace Sciences, 38, pp. 43-76, 2002.

[3] Giotis, A.P., Giannakoglou, K.C., Periaux, J., A Reduced-Cost Multi-Objective Optimization Method Based on the Pareto Front Technique, Neural Networks and PVM, ECCOMAS 2000, European Congress on Computational Methods in Applied Sciences and Engineering, Barcelona, 2000.

[4] Emmerich, M., Giotis, A., Ozdemir, M., Back, T., Giannakoglou, K.C., Metamodel-Assisted Evolution Strategies, 7th Intern. Conf. on Parallel Problem Solving from Nature, PPSN 2002, Granada, Spain, 2002.

[5] Kampolis, I.C., Papadimitriou, D.I., Giannakoglou, K.C., Evolutionary Optimization using a new Radial Basis Function Network and the Adjoint Formulation, Inverse Problems, Design and Optimization (IPDO) Symposium, Rio de Janeiro. Brazil, 2004.

[6] Kampolis, I.C., Karangelos, E.I., Giannakoglou, K.C., Gradient–assisted Radial Basis Function Networks: Theory and Applications, Applied Mathematical Modelling, 28, pp. 197-209, 2004.

[7] Roe, P., Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes, Journal of Computational Physics, 43, pp. 357-371, 1981.

[8] Ong, Y.S., Nair, P.B., Keane, A.J., Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling, AIAA Journal, Vol. 41, No. 4, April 2003.

[9] Ong, Y.S., Lum, K.Y., Nair, P.B., Shi, D.M. Zhang, Z.K., Global Convergence of Unconstrained and Bound Constrained Surrogate–Assisted Evolutionary Search in Aerodynamic Shape Design, CEC 2003, Australia, 2003.

[10] Ulmer, H., Streichert, F., Zell, A., Evolution Strategies assisted by Gaussian Processes with improved Pre–Selection Criterion, CEC 2003, Australia, 2003.

[11] Jameson, A., Optimum Aerodynamic Design using CFD and Control Theory, AIAA Paper 95-1729-CP, 1995.