IAHR
24th Symposium on Hydraulic Machinery and Systems
OCTOBER 27-31, FOZ DO IGUASSU

RESERVED TO IAHR

Constrained Multi-Objective Design Optimization of Hydraulic Components Using a Hierarchical Metamodel Assisted Evolutionary Algorithm. Part 1: Theory

H.A. Georgopoulou†, S.A. Kyriacou† and K.C. Giannakoglou†, P. Grafenberger*, E. Parkinson**

† National Technical University of Athens, NTUA, Parallel CFD & Optimization Unit,
P.O. Box 64069, Athens 157 10, GREECE, Tel: (30)-210.772.16.36, Fax: (30)-210.772.37.89, e-mail: kgianna@central.ntua.gr

* Andritz VA TECH HYDRO, RD, Lunzerstrasse 78, 4031 Linz, Austria,

** Andritz VA TECH HYDRO, RD, Rue des Deux-Gares 6, 1800 Vevey, SWITZERLAND

Abstract

This paper is concerned with optimization methods which, in combination with CFD-based analysis tools, can efficiently be used for the design-optimization of hydraulic turbine blades. It particularly focuses on metamodel-assisted evolutionary algorithms (MAEAs) used as either stand-alone tools or the main components of a hierarchical optimization algorithm (hierarchical MAEAs or HMAEAs). In a HMAEA, search is carried out on regularly communicating levels using models or search tools of different complexity and CPU cost; two levels are often sufficient though this is not mandatory. Additional economy in the CPU cost required to reach a successful design can be achieved by using surrogate evaluation models (the so-called metamodels) for the major part of search on each level. The metamodels exploit the "experience" gained during the evolution to approximately pre-evaluate new offspring generated by the EA and select the most promising among them for re-evaluation on the problem-specific tool. The metamodels used herein are radial basis function networks, trained on the fly on a small number of previously evaluated individuals in the vicinity of each offspring. Basic tuning parameters of a HMAEA are the parent and offspring population sizes per level, the minimum number of previously evaluated individuals that must be available prior to the metamodel-based pre-evaluations, the size of training pattern sets, the percentage of the population selected to undergo exact evaluation, the frequency of interlevel data migrations as well as the migration policy rules (determining the migrating individuals and the individuals to be displaced by them). In this paper, the theory of HMAEAs is presented and, then, emphasis is laid to the parametric study of the most important "tuning" parameters of MAEA, which is its basic component. This study is based on two-objective designs of Francis and Kaplan runners. In the companion paper, the application of hierarchical MAEAs on the design-optimization of hydraulic turbine blades is presented.

Keywords: design optimization algorithms; evolutionary algorithms; metamodels; design of hydraulic components.

INTRODUCTION

To be competitive and deal with strictly constrained contracts, hydropower industries rely on modern methods for the design-optimization of hydraulic turbines and their components. Short delivery times for new products (runners, etc) must often be met. For this purpose, CFD tools and optimization techniques, such as evolutionary algorithms (EAs), gradient-based methods, etc, are in use. The two joint papers by the same groups present an integrated optimization procedure which can be adapted to the design of optimal blades of runners, by considering more than one objectives and/or more than one operating points. Compared to conventional methods, the present method is based on hierarchical metamodel-assisted EAs (HMAEAs) and aims at reducing the overall CPU cost of the optimization process. The presentation of HMAEAs and a few relevant parametric studies (on real problems, such as the design of Francis and Kaplan runners) helping the reader to become acquainted with them are presented in the paper in hand; their use on hydro turbine blade optimization, with discussions on the objectives and constraints used, are exposed in the companion paper [1].

EAs are popular in engineering design due to their ability to handle single- and multi-objective, constrained optimization problems by repetitively calling to the analysis software for the evaluation of candidate solutions. Multi-objective problems are solved by computing fronts of Pareto optimal (non-dominated) solutions, from which a single solution can be chosen only after employing additional decision-making criteria. EAs ask for no access to the analysis source code used as an “evaluation black-box” which, when presented by a set of values of design variables, computes the fitness or cost value(s) of the corresponding candidate solution. Thus, even commercial (such as CFD tool) evaluation software can readily be used. If, however, a single evaluation is computationally demanding, this increases noticeably the CPU cost of the optimization. To overcome this problem, the use of low cost surrogate models or metamodels (polynomial response surfaces, artificial neural networks, etc) in place of the problem-specific tool, has been proposed. This gives rise to the so-called metamodel-assisted EAs (MAEAs, [2]). There are several ways to incorporate metamodels within the evolution. The method employed herein uses local metamodels (radial basis function networks, in specific) trained on the fly on some of the neighboring previously-evaluated individuals. The metamodel-based evaluations serve to pre-screen the population members so as to select only the most promising among them, which will then undergo exact evaluations on the CFD tool. As shown in [3,4,5], by Inexactly Pre-Evaluating (IPE technique) the population in each generation, the CPU cost can be reduced by even an order of magnitude and the delivery time of new products is thus shortened.

Jointly with or separately from the IPE technique, the search for optimal solutions can be based on hierarchical schemes. A two-level structure is the easiest way to employ hierarchy during the optimization; though more levels can be used and have been used by the authors in other cases, herein we stick with only two. On the high level, which is responsible for delivering the optimal solution(s), search is based on an analysis tool with sufficient accuracy (for instance, an accurate flow solver and adequately fine meshes). On the low level, less accurate but less CPU demanding tools are used (for example, low fidelity CFD tools and/or coarser grids). So, any evaluation on the low level is less costly and search aims at exploring the design space at low cost. A two-way communication between the two levels, based on the “controlled” migration of promising individuals, is used. The hierarchical search which is based on MAEAs will be referred to as hierarchical MAEA or HMAEA, [6].

The rest of the paper is organized as follows: the multi-objective, constrained EA, MAEA and HMAEA platforms are presented. Then, a number of investigations are carried out to “tune” the MAEAs, prove that they outperform conventional EAs and shed light into the most efficient way of using them. This study is carried out on the two-objective optimization of Francis and Kaplan runners. In the companion paper [1], the so-tuned MAEA is used on each level of a hierarchical scheme (HMAEA) to design hydraulic turbine rotors.

METAMODEL-ASSISTED OPTIMIZATION – HIERARCHICAL MAEAs

Evolutionary Algorithms (EAs) are optimization methods capable of locating optimal solutions to single and multi-objective [8] optimization (SOO or MOO) problems through the evolution of populations of candidate solutions. EAs handle three populations of candidate solutions, namely offspring S^{λ}, parent S^{μ} and elite S^{a}. These interact during the application of the evolution (parent selection, crossover, mutation) and elitist operators which imitate the natural evolution mechanisms. The elite set is populated by the best individuals found thus far. In SOO, S^{a} degenerates to a single solution. In MOO, it includes all or some of the non-dominated solutions; by definition a solution A dominates B if A is no worse than B for all objectives and wholly better for at least one objective. Once the non-dominated solutions are available, extra decision criteria are required for choosing among them. This final S^{a} is also referred to as the Pareto front of optimal solutions. EAs for MOO are based on EAs for SOO with the additional use of a process that transforms the objective function vectors to scalar utilities (equivalent to the scalar objective function in a SOO). This is often referred to as fitness assignment and incorporates dominance and proximity based criteria. In constrained optimization problems, candidate solutions are penalized accordingly. Herein, all penalties are computed using exponential functions that multiply the elements of the cost function vector.

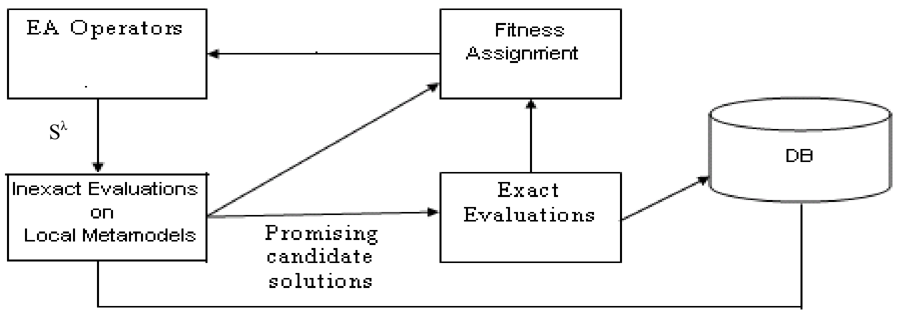
EA Operators
Fitness Assignment
S^λ
DB
Inexact Evaluations on Local Metamodels
Exact Evaluations
Promising candidate solutions

Figure 2. Schematic presentation of a MAEA. During the first few generations, all offspring are evaluated using the exact evaluation tool and recorded into the DB. Then, the Inexact Pre-Evaluation (IPE) phase begins, during which all exactly (re-) evaluated individuals are recorded in the DB, to be used to train the metamodels in the forthcoming generations.

EAs can also benefit from the use of metamodels in order to reduce their computational burden (MAEAs); herein, we make use of radial basis function networks (RBFNs) as metamodel [9]. They act as low cost approximation tools and require the existence of a database (DB) of exactly evaluated individuals to be trained on. The optimization algorithm starts as a conventional EA (without using metamodels) and, once the minimum number of evaluations has been carried out, the offspring pre-evaluation on the metamodel starts. The DB is populated by the exactly evaluated individuals. Locally trained metamodels are used; i.e. for each offspring, its closest DB entries are identified and an RBFN is trained on them. The λ_e most promising population members (out of the λ offspring), according to the metamodel-based fitness, are selected to undergo exact (re-)evaluation using the costly evaluation tool. All exactly evaluated individual are archived in the DB. This algorithm has been proposed in [3,4,5] where this was referred to as IPE (Inexact Pre-Evaluation) method; the relevant flowchart is shown in figure 2.

HMAEAs are enhanced MAEA variants which outperform conventional EAs or single-level MAEAs in many engineering applications, including CFD cases. Conceptually, a HMAEA establishes a multilevel search mechanism and splits the computational burden among the levels. On the low level, a low cost exploration of the search space is carried out through global search methods, less demanding or less accurate evaluation tools or by even using reduced design variable sets, etc. The higher levels make use of accurate and, thus, more expensive tools, enhanced parameterizations global or local optimization tools. These levels mainly serve to refine immigrants from the lower levels. Intercommunication and two-way migrations of individuals between adjacent levels are necessary. The number of levels, the frequency of migration and the number of immigrants are user-defined parameters.

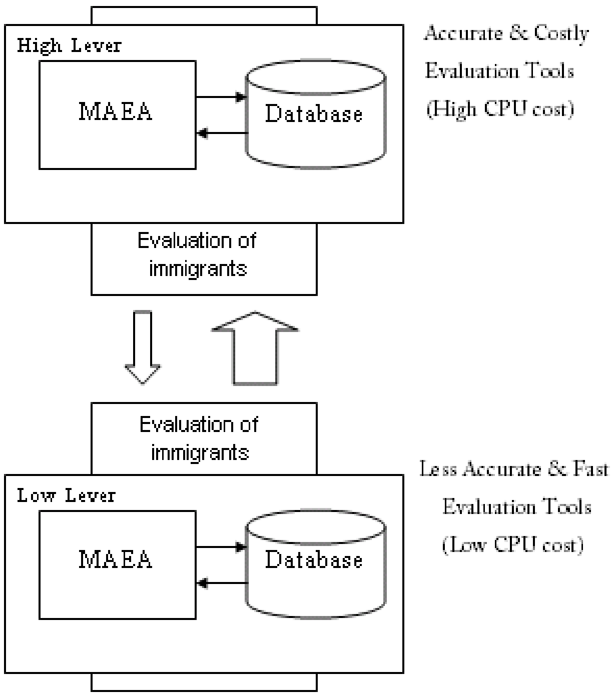

Figure 4. Illustration of a two-level hierarchical optimization scheme employing different MAEAs on each level [7]. Note that each level must maintain its own DB since different evaluation tools are in use. All migrated solutions must be re-evaluated on the destination level.

ASSESSMENT OF MAEAs IN RUNNER DESIGN PROBLEMS

Design optimization of a Francis Runner - Parametric study on MAEAs

The first test case is concerned with the design optimization of a Francis runner, figure 5, aiming at achieving a pressure coefficient c_p distribution with appropriate features while minimizing cavitation. The runner has 17 blades and its external diameter is 0.34 m. It is designed to operate at 40 m head, volume flow rate at discharge equal to 0.38 m^3/s and rotating speed equal to 58.84 rad/s. The analysis of the flow field is based on an Euler flow solver which, along with the grid generation tool, determines the CPU cost per evaluation.

The first objective function F_1 is defined after post-processing the numerically predicted c_p distributions at the crown, mid-span and band and expresses how monotonic and constantly decreasing the load profiles along the blade cuts are. The second objective F_2 is equal to the opposite of the minimum pressure on the blade which should be minimized to reduce or avoid cavitation. For further details on the objectives, the reader should turn to the companion paper [1] where parameterization issues are also discussed. In brief, Bezier polynomials are used to model the blade hydrofoil cuts and their control points constitute the optimization variables

[10]. A number of constraints related to pressure, mass and swirl distributions along the blade and/or the outlet and discharge of the runner are imposed.

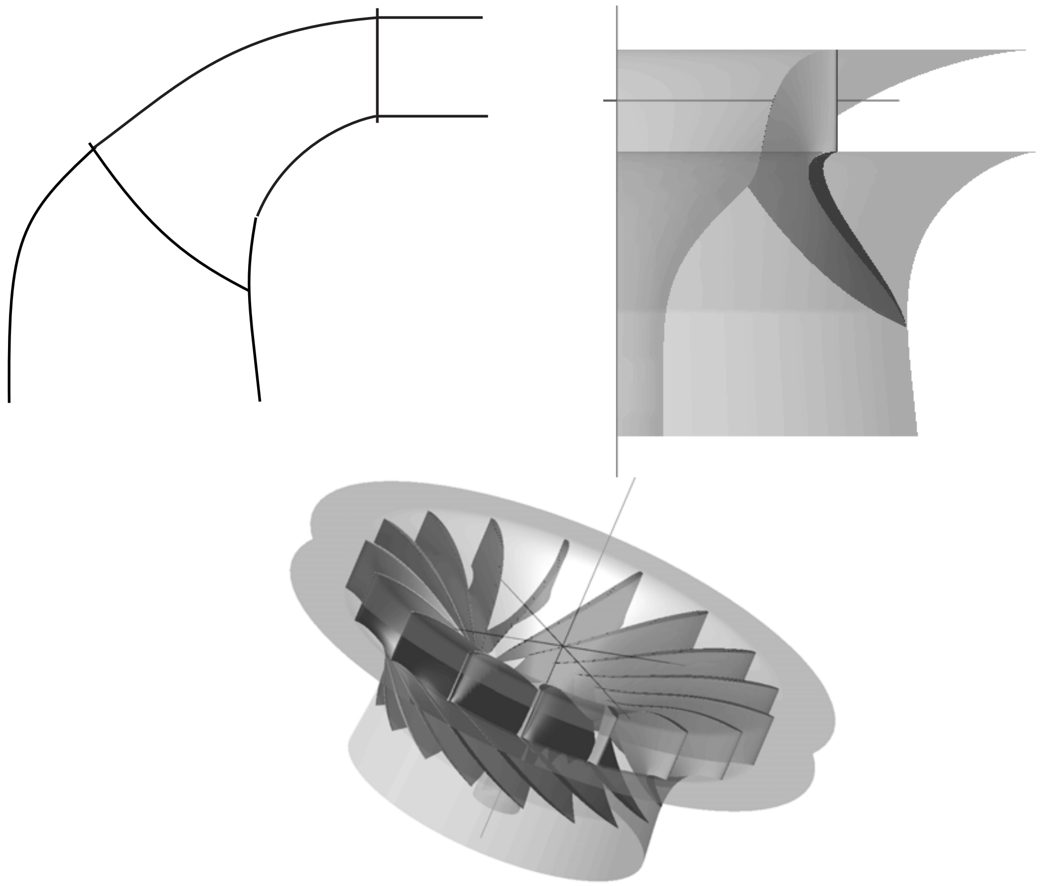

Figure 5. Francis runner: view of the flow channel on the meridian plane (top-left), 3D view of a single blade (top-right) and 3D view of the entire turbine (bottom).

In this case, the optimal use of metamodels within the IPE technique is investigated. The performance of three MAEA configurations with the same parent, offspring and elite populations (μ=20, λ=40 and a=40, respectively) but different numbers of exactly re-evaluated members per generation (λ_e=3, 6 and 10 candidate solutions) are compared. For the sake of comparison, a conventional EA (λ_e=λ) is also used. We abbreviate each method variant to (μ,λ,a;λ_e). In figure 6, the three fronts of non-dominated solutions computed at the cost of 400 calls to the Euler code are shown. The presented fronts of non-dominated solutions are approaching the real Pareto front but, after 400 evaluations, the algorithm is not considered to be fully converged. However, the comparison of performances can better be seen at this stage of the evolution. The best performing variant is (20,40,40;6) and will, finally, capture the sought Pareto front earlier than any other variant tested (not shown here). As shown in figure 7, the MAEA(20,40,40;6) variant noticeably outperforms conventional EAs (denoted by MAEA(20,40,40;40) or, merely, MAEA(20,40,40)).

We next investigate two important parameters affecting the metamodel-based IPE of population members during the evolution. These are the size of the training pattern set used for each one of the new population members and the size of the starting DB, i.e. the minimum number of DB entries that must be archived before activating metamodels and the IPE screening. We tested two different configurations for each one of these parameters and the results are shown in figures 8 and 9. It should become clear that the training pool size, formed by the closest already evaluated solutions to any new individual, is not fixed; instead, the user defines a lower and upper acceptable size. Several criteria are used to determine the most appropriate size and guarantee that there is “adequate” data around the new individual, in all directions in the search space. So, in figure 8, the performances of local RBF networks as metamodels, trained with [40,50] and [80,90] neighbouring database entries, are compared. With the same number of Euler-based evaluations, the RBF networks trained on fewer patterns perform better. From a certain point of view, this reconfirms the decision of using local metamodels. By even neglecting the (slightly) higher training cost when more training patterns are used, figure 8 demonstrates that it is more advantageous to use a limited number of training data in the close neighbourhood of the new individual. On the other hand, figure 9 serves to show the most appropriate “timing” for starting the use of the metamodel-based filter. Two computations, where the metamodel action started after 200 and 100 Euler-based evaluations are compared. By comparing the two fronts of non-dominated solutions, computed at the same CPU cost, we conclude that there is no clear superiority of any of them. Practically, this figure (and some other runs, not shown here) dictates that the IPE should start once the database gets between 100 and 200 entries.

For the same case, we also tested the possibility of improving the prediction capabilities of metamodels by using enhanced RBF networks. As such, we tested the RBF networks enhanced by the autocatalytic action of the so-called importance factors, as described in the Appendix. As shown in figure 10, the use of the enhanced RBF networks greatly improves the performance of MAEA(20,40,40;6).

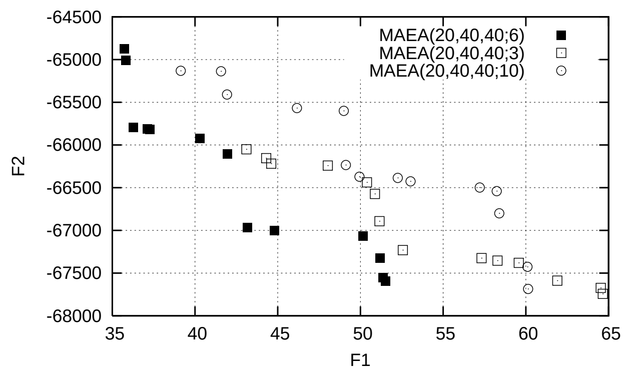

Figure 6. Francis runner optimization: Comparison of the fronts of non-dominated solutions computed after 400 Euler-based evaluations using three MAEA variants. All the other parameters, used to adjust the use of metamodels, were the same.

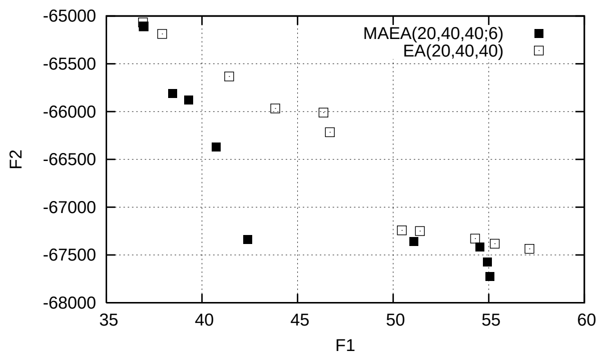

Figure 7. Francis runner optimization: Comparison the fronts of non-dominated solutions computed by the best performing MAEA(20,40,40;6) and a conventional EA after 500 Euler-based evaluations. All empty boxes are clearly dominated by the filled ones and this shows the advantages of using metamodels.

Use of MAEAs on the design optimization of a Kaplan Runner

This test case shows how MAEAs can be also implemented in the design optimization of Kaplan runners. The runner has 3 blades and an external diameter of 0.34 m. The head and volume flow rate, at the design point, are 7 m and 0.66 m^3/s, respectively. The rotating speed is 126 rad/s. Targets are related to the optimality of the c_p profiles (at crown, mid-span and band, F_1) and the minimization of the cavitation effect (F_2).

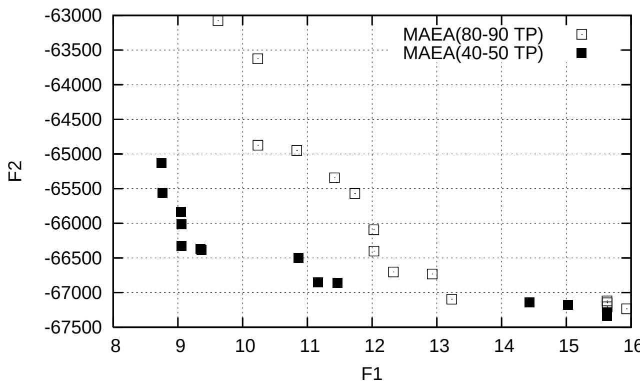

Figure 8. Francis runner optimization: Study of the role of the size of the training pattern pool for the local metamodels. Non-dominated solutions computed after 400 Euler-based evaluations using MAEA(20,40,40;6) based on RBF networks trained on (a) 80 to 90 and (b) 40 to 50 training patterns per new individual.

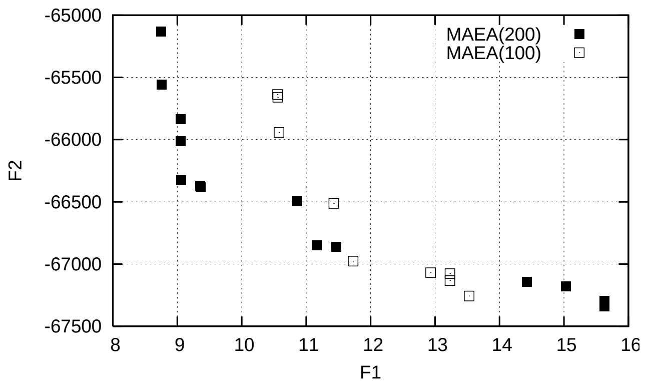

Figure 9. Francis runner optimization: Non-dominated solutions for two different database sizes at the starting generation of the IPE phase, namely for 100 and 200 archived members. Both runs correspond to the same CPU cost (400 exact evaluations).

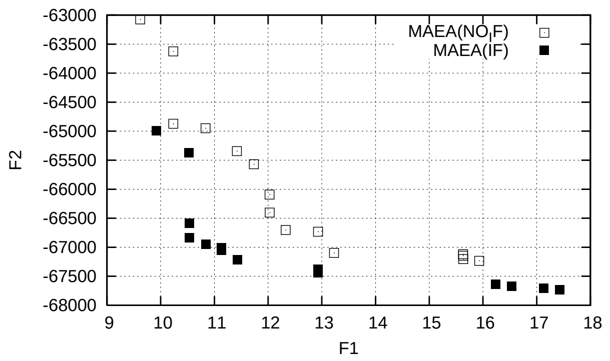

Figure 10. Francis runner optimization: Comparison of the Pareto fronts of optimal solutions found after 400 exact evaluations for a MAEA with and without the use of Importance Factors.

Each design is parameterized using Bezier polynomials, as in the previous case. The control point coordinates constitute the optimization variables and, thus, they were allowed to vary among user-defined bounds. All candidate solutions were forced to respect the list of constraints described in the problem formulation section of the companion paper [1].

The optimization was carried out using conventional EAs (60,40,60) and MAEAs (60,40,60;10); the IPE technique is used with λ_e=0.15λ. Figure 11 presents the fronts of non-dominated solutions after 2800 and 3500 exact evaluations. After 3500 evaluations, the optimization is adequately converged. It is obvious that MAEA outperforms EA at both instants during the optimization. Two extreme members of the MAEA front have been post-processed and their performance is compared and contrasted in figure 13.

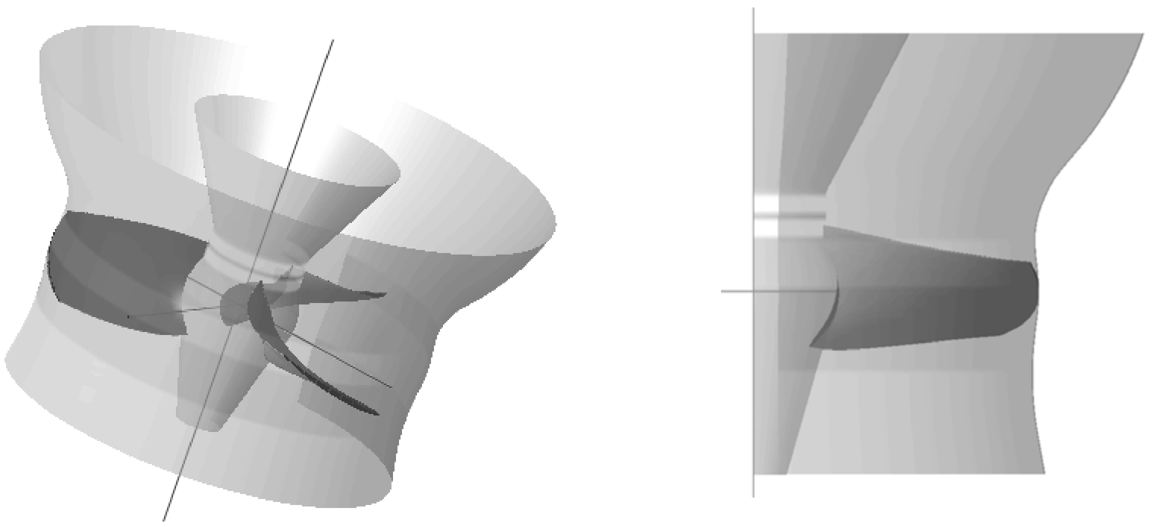

Figure 11. Kaplan runner optimization: 3-dimensional view of the Kaplan runner under investigation.

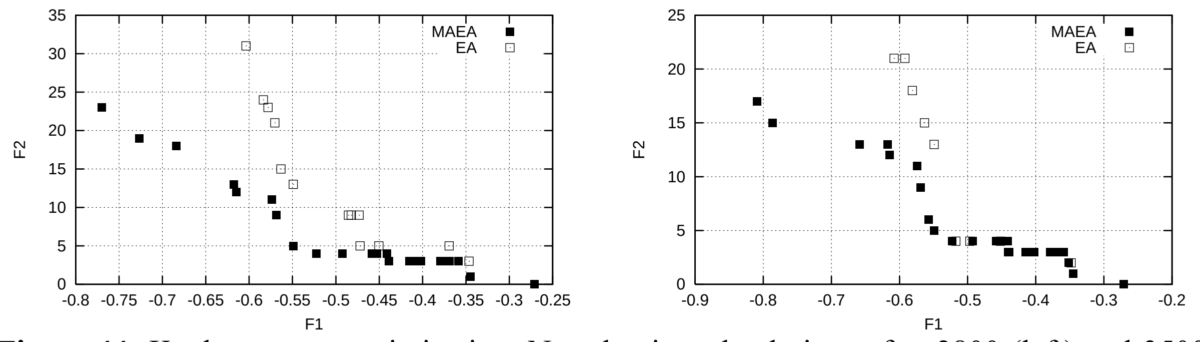

Figure 11. Kaplan runner optimization: Non-dominated solutions after 2800 (left) and 3500 (right) exact evaluations.

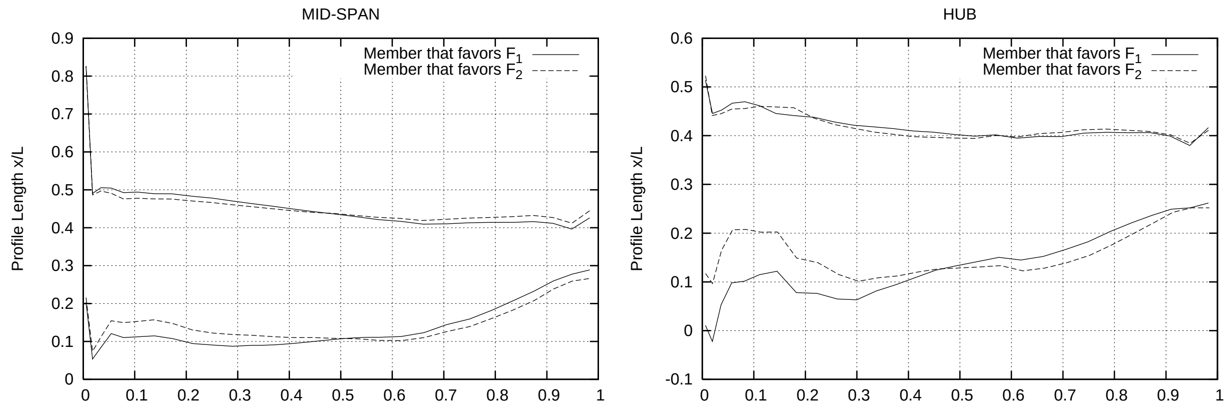

Figure 13. Kaplan runner optimization: Pressure coefficient (c_p) profiles at hub for the "extreme" elite members on the non-dominated front (computed by the MAEA; figure 11, right). The continuous line corresponds to the member with the best c_p profiles (21 cavitation loci on the blade), whereas the dashed line corresponds to the one that minimizes the cavitation effect (2 cavitation loci, only). For the first one, the difference in c_p between pressure and suction side is decreasing "more constantly" than for the second one, resulting to decreasing load production along the blade,.

CONCLUSIONS

An optimization platform that combines surrogate models and evolutionary search, in order to reduce the computational burden of engineering optimization applications such as the design of hydraulic turbine components, was presented and tested. The use of metamodels in the framework of MAEAs, improves significantly the efficiency of conventional EAs. Moreover, further CPU gain can be achieved by implementing MAEAs in a two-level (hierarchical) structure. An investigation on the optimal calibration of MAEAs was presented on the design of a Francis runner, aiming at both (a) cavitation reduction and (b) c_p profile improvement. During this study, we tried to optimally adjust the number of promising offspring members that undergo exact evaluation at each generation of the MAEA. Additionally, optimal MAEA results were compared to those of a conventional EA so as to prove the superiority of the former. Comparison of MAEAs and EAs was also done through a second hydraulic Kaplan runner design case, with similar objectives. In the companion paper [1], an extended discussion on the problem formulation and the proposed two-level – multi operating optimization process, that makes use of the optimization tools presented and tested herein, is given. The optimization tool used herein is software EASY [6], developed and brought to market by the National Technical University of Athens.

BIBLIOGRAPHICAL REFERENCES

1. P. GRAFENBERGER, E. PARKINSON, H. GEORGOPOULOU, S. KYRIACOU, K. GIANNAKOGLOU, October 2008. *Constrained Multi-Objective Design Optimization of Hydraulic Components Using a Hierarchical Metamodel Assisted Evolutionary Algorithm. Part 2: Applications*. 24th IAHR Symposium Foz do Iguassu, Brazil.
2. K.C. GIANNAKOGLOU: '*Design of Optimal Aerodynamic Shapes using Stochastic Optimization Methods and Computational Intelligence*', Int. Review Journal Progress in Aerospace Sciences, Vol. 38, pp. 43-76, 2002.
3. K.C. GIANNAKOGLOU, A.P. GIOTIS and M.K. KARAKASIS, 2001. *'Low-Cost Genetic Optimization based on Inexact Pre-evaluations and the Sensitivity Analysis of Design Parameters*. Inverse Problems in Engineering, Vol. 9, pp. 389-412.
4. M.K. KARAKASIS, A.P. GIOTIS and K.C. GIANNAKOGLOU, 2003. *Inexact Information Aided, Low-cost, Distributed Genetic Algorithms for Aerodynamic Shape Optimization.* Int. Journal for Numerical Methods in Fluids, Vol. 43, pp. 1149-1166.
5. M. KARAKASIS, K.C. GIANNAKOGLOU, 2006. *On the Use of Metamodel-Assisted Multi-Objective Evolutionary Algorithms*. Eng. Optimization, Vol. 38(8), pp. 941-957.
6. I.C. KAMPOLIS, K.C. GIANNAKOGLOU, 2008. *A Multilevel Approach to Single- and Multiobjective Aerodynamic Optimization*. Computer Methods in Applied Mechanics and Engineering, to appear.
7. EASY. Evolutionary Algorithms SYstem, 2006. URL http://velos0.ltt.mech.ntua.gr/EASY.
8. E. ZITZLER , M. LAUMANS, L. THIELE , Zurich 2001. *SPEA2 Improving the strengthPareto evolutionary algorithm for multiobjective optimization*, Eurogen 2001, Conference, CIMNE, Barcelona,2002, pp. 19–26.
9. S. HAYKIN. *Neural Networks, a comprehensive foundation,* Prentice Hall ISBN 0-13-273350-1.
10. E. PARKINSON, 1995. *Test Case 8: Francis turbine.* Turbomachinery Workshop ERCOFTAC II.

APPENDIX

RBF Networks Enhanced by Importance Factors

An RBF network comprises three layers, namely the input, hidden and output layer, [9], as shown in figure A.1 (with a single output node for sake of simplicity). Signals propagate through the network in the forward direction, from the input to the output layer, by performing a nonlinear mapping followed by a linear one. The latter is related to the weight coefficients w_k that must be computed during the training on a number of available patterns. An RBF network to be used within a MAEA should have N input units, i.e. as many as the design variables. The hidden layer includes K nodes, associated with the so–called RBF centers $c^{(k)}$. At each hidden neuron a nonlinear mapping of the input signals to a single value is carried out using the radial-basis activation function $G:R^N \rightarrow R$, acting on the distance of input x from the corresponding center $c^{(k)} \in R^N$. A widely used activation function is the Gaussian function, $G(h,r) = \exp\left(h^2 / r^2\right)$, where $h = \left\| x - c^{(k)} \right\|_2$

The values that the radii or widths r take on may considerably affect the prediction abilities of the network; these are computed using heuristics, [9]. The output layer includes as many nodes as the responses of the network. The single response we are dealing with is expressed by the sum of the weighted output signals from the hidden neurons, as follows

$$o(x) = \sum\nolimits_{k=1}^{K} w_k G\left(\left\| x - c^{(k)} \right\|_2, r \right)$$

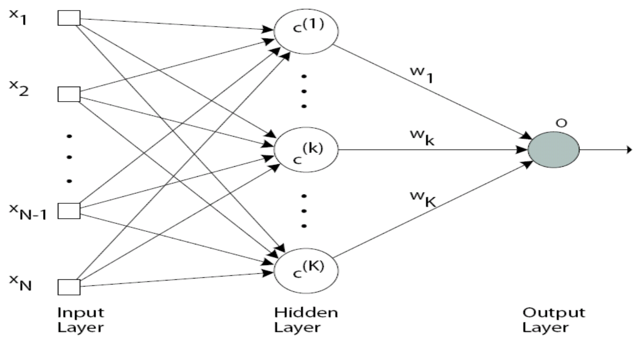

Figure A.1 RBF network with N inputs, K hidden neurons and a single output.

The idea of RBF networks with Importance Factors (IFs), as proposed in [3], was to make the last expression sensitive to the effect each design variable has on the response. So, instead of the "isotropic" norm $\left\| x - c^{(k)} \right\|_2$, a weighted norm is used, which is defined as $\|x-c^{(k)}\|_{wei} = \sqrt{\sum\nolimits_{n=1}^{N} I_n (x_n - c_n^{(k)})^2}$ by introducing the importance factors I_n, n = 1,N. By definition, a high I_n value denotes high sensitivity of the response (i.e. the cost function) with respect to the n–th input variable, whereas low I_n values denote the opposite. The computation of I_n is based on the RBF network predictions; it is important that these are computed from and used by the RBF networks. Their computation is based on the current best solution (superscript (b)) and the I_n values are updated whenever a better solution is found. For the new best solution, the corresponding local RBF network is built and N partial derivatives $\partial o^{(b)} / \partial x_n$ are computed using the network. Using these derivatives, the importance factors values are computed as

$$I_n = \left| \frac{\partial o^{(b)}}{\partial x_n} \right| \Bigg/ \left(\sum\nolimits_{i=1}^{N} \left| \frac{\partial o^{(b)}}{\partial x_i} \right| \right).$$