9th International Conference on Numerical Ship Hydrodynamics
Ann Arbor, Michigan, August 5-8, 2007

Nonlinear Programming Approaches in the Multidisciplinary Design Optimization of a Sailing Yacht Keel Fin

Emilio F. Campana*, Giovanni Fasano*†, Daniele Peri* and Antonio Pinto*

(* *INSEAN* - Italian Ship Model Basin, Rome, Italy,
†*IASI-CNR* - Istituto di Analisi dei Sistemi ed Informatica "A. Ruberti")

I. Abstract

This paper discusses certain shape optimization problems in which the solution depends on more *disciplines*, presenting results for the optimal shape of the keel fin of a sailing yacht, simultaneously accounting for hydrodynamics and elasticity. A *pure* fluid dynamic approach would simply consider the hull, the keel fin and the bulb as rigid, connected bodies. In the Multidisciplinary Design Optimization (MDO) framework these are considered instead as elastic and the shape of the keel fin is hence modified by the hydrodynamic loads. As a result, the final performances of the yacht are also influenced by the structural behavior of the fin.

For this problem we study and compare a suite of different MDO formulations. The optimal design problem is finally tackled considering a Global Optimization (GO) problem within a MDO framework. MDO both includes difficult theoretical and computational aspects, determined by the simultaneous solution of different *disciplines* affecting each other, and by the related solvers involved all at once in the convergence of the overall optimization framework. We first introduce and describe some MDO approaches from the literature. Then, we consider our MDO scheme where we deal with the GO box-constrained problem

$$\min_{a \le x \le b} f(x), \qquad f : \mathbb{R}^n \to \mathbb{R}.$$

In general, the objective function $f(x)$ is non-linear and non-convex and, in simulation based design approaches, also costly. We also assume that the solution of the problem requires the use of a derivative-free method since the derivatives of $f(x)$ are unavailable and/or the function must be treated as a 'black-box' (see [30] and [23], [40]). Within this framework we study some globally convergent modifications of the evolutionary Particle Swarm Optimization (PSO) algorithm [24], suitably adapted for box-constrained optimization. To this purpose we adopt both the theory described in [27] for exact methods, and the generalized PSO scheme [10], which includes the standard PSO scheme. Finally, we report our numerical experience for the design of the sailing yacht elastic keel fin.

Keywords : Multidisciplinary Design Optimization, Global optimization, Particle Swarm Optimization, Derivative-free methods, Global Convergence.

II. Introduction

Reliable Computational Fluid Dynamics (CFD) solvers, as long as design databases, are a modern way to reduce the number of experimental tests on models. However, due to the increasing interest on the design optimization, mature CFD analysis should also be used in a larger context, where the simultaneous effectiveness of different solvers is sought. The problem complexity has so far prevented from assessing a satisfactory reformulation of the overall design problem, into a unique mathematical programming formulation. In fact, traditional approaches to shape design have often focused on the satisfaction of feasibility constraints of the problem in hand, rather than tackling optimal solutions.

As a result, when several disciplines are involved in the design problem, different heuristic methods have been used, which address individual disciplinary optimization. The growing complexity of modern engineering systems has spurred designers to provide more efficient heuristics. Unfortunately, the latter are often based on designers personal skills *on the specific problem treated*, instead of relying on exact and self-adaptive techniques. Thus, the application of heuristics to new instances of general real problems, may be unsatisfactory. As a result, new designs are often desirable, but one cannot rely on historical databases, which are mainly the result of human experience rather than quantitative methods.

These reasons motivate our interest for the systematic numerical approach to MDO. In our case the *multidisciplinarity* refers to the design of a ship, which encompasses interacting physical phenomena as hydrodynamics, structural mechanics, and control.

Recently a larger number of real industrial applications have included complex optimization approaches, where efficient solutions were claimed. Aircraft and spacecraft engines design are among the latter applications, which intrinsically yield challenging MD formulations (see e.g. [1]). Observe that in most of the cases, MDO methodologies substantially

imply a process of parallelization and coupling of different independent optimization schemes (*disciplines*). Moreover, a distinguishing feature of MDO formulations is that the interaction among the standard optimization approaches, each related to a discipline, is non-trivial.

Furthermore, the accurate coupling of disciplines might be essential to guarantee the convergence properties of the overall framework. Thus, a specific care should be paid to provide the *correctness* of the MDO formulation, for the problem in hand. On the other hand, both the theoretical results and the methods provided by nonlinear programming, for standard optimization problems, must be coupled accordingly. This suggests that those MDO formulations, which strongly rely on the abilities of optimization methods, may efficiently gain advantage from conventional optimization [3], [4].
On this guideline, observe that most of the typical issues considered for nonlinear programming formulations (e.g. feasibility, optimality conditions, sensitivity analysis, duality theory, etc.), require a suitable adaptation when considered in an MDO framework.

The first attempts to give a taxonomy of MDO formulations, simply relied on managing standard nonlinear programming schemes in a sequential fashion. I.e., no real coupling among the disciplines was considered, and the interaction among optimization codes was often non-significative. The latter scenario was essentially the consequence of the early incapability to match several numerical codes, independently studied for each discipline. In addition, large scale MDO problems were even tougher, so that coarse solutions had to be allowed and the coupling among disciplines was possibly weakened. As a consequence, the early MDO formulations often described quite poorly several non-convex real challenging problems.

This work briefly reviews the main results of MDO literature, and details some more recent MDO formulations based on multilevel programming. In the latter schemes, the overall MDO formulation is decomposed into a *master* level problem and a set of optimization *subproblems*. The master (i.e. the system level) problem depends on the optimal solutions of subproblems; conversely each subproblem includes a set of unknowns provided by the master level. Then, we study and solve the MDO formulation of a sailing yacht keel fin design problem, where the derivatives of the objective functions are unavailable. The latter problem is a hydroelastic design optimization problem for a race yacht, where the fin is used to sustain the bulb, adopted to increase the stability of the sailing yachts during a competition (e.g. the *America's Cup*).

Unlike to a pure fluid dynamic approach, in a multidisciplinary framework, the shape of the keel fin is influenced by both the weight of the bulb and the hydrodynamic forces arising from the different sailing positions. Therefore, the final performances of the yacht are undoubtedly affected by the structural behavior of the fin. We study and compare a suite of different MDO formulations underlying the latter problem. The interaction between the two disciplines is approached with the application of derivative-free optimization methods. We highlight that for several MDO problems the real functionals to be minimized are described by expensive simulations and the derivatives are unavailable. This strongly motivates the interest for effective derivative-free techniques.
We apply a modified Particle Swarm Optimization (PSO, [11]) method, which belongs to the family of evolutionary algorithms. PSO [24] owes its popularity to the reasonable balance between its overall computational cost and the quality of the final solution it provides. More specifically, the PSO algorithm is an *iterative method*, which is tailored to detect a global minimum for the unconstrained optimization problem

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1}$$

i.e., a point $x^* \in \mathbb{R}^n$ such that $f(x^*) \leq f(x)$, for any $x \in \mathbb{R}^n$. For a computationally costly function $f(x)$, exact iterative methods are possibly too expensive or they may not provide a current satisfactory approximation of the solution, after a finite number of iterations. In these scenarios heuristics may be fruitfully used, whenever the computational resources and/or the time allowed for the computation are severely bounded. On this guideline, PSO proved to be both effective and efficient on several practical applications from real life [35].
Since the algorithms as PSO are mainly heuristics (see [9], [41] for exceptions), their progress may be eventually very slow. The latter drawback can be explained by observing that these techniques completely disregard any condition related to first order information on the objective function. Thus, eventually they possibly have poor performance and hardly retain theoretical convergence properties. In this regard, we propose a suitable modification of PSO. Our proposal both preserves asymptotic convergence properties, and after a finite number of iterations it provides a satisfactory approximation of the solution. We focus here on a modification of the PSO algorithm, where converging subsequences of iterates are generated. The modifications we propose guarantee that the generated subsequences of iterates converge to stationary points, satisfying the first order optimality conditions for the box-constrained problem

$$\min_{a\leq x\leq b} f(x), \qquad x \in \mathbb{R}^n \tag{2}$$

(see also [20], [21]). In particular, we prove the global convergence to first order points under very mild assumptions, in a linesearch framework. Therefore, in this paper we both focus on the theoretical properties of PSO, and propose a numerical experience with PSO on the sailing yacht design problem described above. Definitions and the general mathematical framework about optimization are summarized in the Appendix.

III. Introduction to MDO formulations

As we introduced in Section II, some peculiar aspects should be identified when addressing an MDO formulation. In particular we can consider the following:

- the overall problem includes *several* disciplines;

- each discipline is *essential* to describe the overall problem;
- the overall problem can be *hardly* formulated as a *nonlinear mathematical program*;
- each discipline is an independent problem with its own formulation. The latter formulation and its solution rely on theoretical results (e.g. optimality conditions, sensitivity analysis, convergence analysis), solution techniques (e.g. solution methods, heuristics, etc.), and possibly codes which may not be worth for the other disciplines;
- there exists a procedure (either iterative or direct) such that
  (a) it collects, in a finite number of steps, the results provided by each discipline,
  (b) it yields a partial (intermediate) result of the MDO problem (i.e. the results from the independent disciplines are suitably gathered and coordinated).

Of course the steps (a) and (b) are specifically critical, in as much as they do not indicate a unique nonlinear formulation (and the related feasibility/optimality conditions), for the overall MDO problem.

Some unknowns of the overall formulation, included in the formulations of the disciplines, will be addressed as *design unknowns*, since they have a physical meaning. On the other hand, the formulation associated with each discipline often includes also *state unknowns* (e.g. discipline auxiliary parameters, state unknowns of control systems, variables generated by the discretization of a PDE solver, etc.). The latter variables do not play a direct role in the design problem, i.e. a specific physical meaning is not usually associated with them.

Let us now formally describe the formulation of an MDO problem. Consider the disciplines D_i, $i = 1, \ldots, p$, and let the pair of vectors $(x_i^T, s_i^T)^T \in \mathbb{R}^{n_i+m_i}$ be associated with D_i. Here, $s_i \in \mathbb{R}^{m_i}$ represents the *state* of the i-th discipline D_i, while $x_i \in \mathbb{R}^{n_i}$ are the design unknowns arising in the formulation of D_i. With these positions, and including the subvector $x_0 \in \mathbb{R}^{n_0}$ of design unknowns shared by the p disciplines, we formally introduce the definition for MDO formulations, by considering the overall vectors: $x^T = (x_0^T \; x_1^T \; \cdots \; x_p^T) \in \mathbb{R}^n$, $n = n_0 + n_1 + \cdots + n_p$ (design variables), and $s^T = (s_1^T \; \cdots \; s_p^T) \in \mathbb{R}^m$, $m = m_1 + \cdots + m_p$ (state vector).
We are now ready to give the following general definition for MDO formulations, which will be adopted in this paper, unless differently specified.

Assumption 3.1: Consider a real problem and suppose it involves the disciplines D_i, $i = 1, \ldots, p$. Let $x^T = (x_0^T \; x_1^T \; \cdots \; x_p^T) \in \mathbb{R}^n$, $n = n_0 + n_1 + \cdots + n_p$, and $s^T = (s_1^T \; \cdots \; s_p^T) \in \mathbb{R}^m$, $m = m_1 + \cdots + m_p$, suppose that

1) we can introduce the set $B_i \subseteq \mathbb{R}^{n_0 \times n_i \times m}$ (possibly empty) for the discipline D_i, such that

$$B_i = \{(x_0, x_i, s),\ x_0 \in \mathbb{R}^{n_0},\ x_i \in \mathbb{R}^{n_i},\ s \in \mathbb{R}^m : \\ g_i(x_0, x_i, s) \geq 0, \quad A_i(x_0, x_i, s) = 0\};$$

2) the nonlinear function $f_i(x_0, x_i, s)$ exists (possibly constant), with $f_i : \mathbb{R}^{n_0} \times \mathbb{R}^{n_i} \times \mathbb{R}^m \to \mathbb{R}^{q_i}$, such that the formulation associated with the discipline D_i is

$$\min_{(x_0, x_i, s) \in B_i} f_i(x_0, x_i, s_i), \qquad i = 1, \ldots, p;$$

3) there exist $f(x, s) = \varphi[f_1(x_0, x_1, s), \ldots, f_p(x_0, x_p, s)]$ and $g_0(x, s)$ (either explicitly or implicitly defined), with $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^q$ and $g_0 : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, such that if $B = \{(x, s),\ x \in \mathbb{R}^n,\ s \in \mathbb{R}^m : g_0(x, s) \geq 0,\ (x_0, x_i, s) \in B_i,\ i = 1, \ldots, p\}$ the real problem may be formulated as

$$\min_{(x,s) \in B} f(x, s). \tag{3}$$

♢

Definition 3.1: Let the Assumption 3.1 hold; then, we say that (3) is a `nonlinear MDO formulation` for the MDO problem.

♢

We remark that with the latter definition we intend to distinguish between *tractable* MDO problems (i.e. those problems for which a nonlinear MDO formulation exists), and *intractable* MDO problems, whose formulation is misleading or it cannot be described, either explicitly or implicitly, by a mathematical program.
The formulation (3) is only apparently a standard nonlinear program, so that the usual techniques from numerical optimization cannot in general be used to solve it. Indeed, the specific difficulty of (3) is that the feasible set B also includes the so called *MultiDisciplinary Analysis (MDA)*, defined as

$$MDA = \begin{cases} A_1(x_0, x_1, s) = 0 \\ \vdots \\ A_p(x_0, x_p, s) = 0. \end{cases}$$

MDA only implicitly describes a nonlinear system of equation, in as much as the i-th block of equalities $A_i(x_0, x_i, s) = 0$ may not correspond exactly to a set of nonlinear equations. It may be a black-box, which implicitly defines a map among the variables x_0, x_i and s. Moreover, it often corresponds to the discretization of PDE systems, so that the *implicit function theorem* cannot be exploited to retrieve $s = s(x)$. As a result, apart from specific cases, (3) can be hardly solved as a nonlinear program uniquely dependent on the design vector x.

As mentioned in the Appendix, the Karush-Kuhn-Tacker (KKT) conditions (a set of *optimality conditions* for a general mathematical programming problem) require some assumptions on both the objective function (differentiability) and the feasible set in (3) (e.g. constraint qualification). It is not difficult to find simple examples of MDO problems where the latter assumptions do not hold at all. Hence, this proves the intrinsic difficulty of providing both a complete convergence analysis and effective algorithms for the formulation (3).

As we will describe in Section VII, our real MDO problem also imposes box constraints on a subset of the design

unknowns. The latter constraints are implicitly included in the block of inequalities $g_0(x,s) \geq 0$ of (3).

A. A classification for MDO formulations

The definition of a general classification for nonlinear MDO formulations is an intriguing issue (see for instance [4] and therein references). On this purpose we partially rewrite (3) as

$$\min_{(x,s,t)\in\hat{B}} \hat{f}(x,s,t), \qquad \hat{B} = \Gamma_1 \cap \Gamma_2 \cap \Gamma_3, \tag{4}$$

where the sets Γ_1, Γ_2, Γ_3 are respectively given by

Design Constraints:

$$\Gamma_1 = \begin{cases} g_0(x,s) \geq 0 \\ g_1(x_0,x_1,s) \geq 0 \\ \vdots \\ g_p(x_0,x_p,s) \geq 0 \end{cases}$$

Disciplinary Analysis Constraints (MDA):

$$\Gamma_2 = \begin{cases} A_1(x_0,x_1,s_1,t_2,\ldots,t_p) = 0 \\ \vdots \\ A_p(x_0,x_p,s_p,t_1,\ldots,t_{p-1}) = 0 \end{cases}$$

Interdisciplinary Consistency Constraints:

$$\Gamma_3 = \begin{cases} t_1 = \mathcal{C}_1(s_1) \\ \vdots \\ t_p = \mathcal{C}_p(s_p). \end{cases}$$

The main difference with respect to (3) is that the Interdisciplinary Consistency Constraints introduce the new set of unknowns $t^T = (t_1^T \cdots t_p^T)$. For each block $A_i(x_0,x_i,s)$, $i = 1,\ldots,p$ in (3), the vector t_i simply weaken the dependency of $A_i(x_0,x_i,s)$ from the vector s, i.e. the state unknowns of the i-th discipline are de-coupled. This implies that any two blocks i and j among the MDA constraints, i.e.

$$A_i(x_0,x_i,s_i,t_1,\ldots,t_{i-1},t_{i+1},\ldots,t_p) = 0$$

$$A_j(x_0,x_j,s_j,t_1,\ldots,t_{j-1},t_{j+1},\ldots,t_p) = 0,$$

only share the sub-vector of unknowns x_0 and not also the sub-vector s. The latter structure of the formulation could be fruitfully used to apply either *decomposition techniques* or *multilevel methods* to the nonlinear program (4). With a common terminology within the MDO literature, we say that the auxiliary variables t *de-couple the interdisciplinarity* among disciplines.

In spite of the considerations above, the formulation (4) can be hardly solved with a direct application of standard mathematical programming techniques. Thus, we prefer to introduce the following concept of *reformulation* for an MDO formulation, in order to have a *more* tractable alternative scheme.

Definition 3.2: Let the set $\hat{B}$ in (4) be nonempty, let Z^* be the solution set of the MDO formulation (4). We say that $\hat{\mathcal{F}}$ is a *reformulation* of (4) if

- $\hat{\mathcal{F}}$ is a formulation for the MDO problem;
- a smooth nonlinear function $\varphi_{\hat{\mathcal{F}}}$ exists such that $\varphi_{\hat{\mathcal{F}}}(\hat{z}^*) \in Z^*$, for any $\hat{z}^* \in \hat{Z}^*$, where $\hat{Z}^*$ is the solution set of $\hat{\mathcal{F}}$.

$\diamondsuit$

We remark that by the latter definition any MDO formulation is also an MDO reformulation with φ given by the identity. Furthermore, the solution(s) of the reformulated problem *may not be* in general optimal solutions of (4). We also introduce the following definition of *equivalence among reformulations*, in order give criteria for evaluating the solutions provided by different reformulations.

Definition 3.3: Let $\tilde{\mathcal{F}}$ and $\bar{\mathcal{F}}$ be reformulations of (4). Let $\tilde{Z}^*$ and $\bar{Z}^*$ be the solutions sets of $\tilde{\mathcal{F}}$ and $\bar{\mathcal{F}}$ respectively. We say that $\tilde{\mathcal{F}}$ is *equivalent* to $\bar{\mathcal{F}}$ (i.e. $\tilde{\mathcal{F}} \sim \bar{\mathcal{F}}$) if the smooth nonlinear functions $\varphi_{\tilde{\mathcal{F}}}$, $\varphi_{\bar{\mathcal{F}}}$ exist, such that $\varphi_{\tilde{\mathcal{F}}}(\tilde{x}^*,\tilde{s}^*,\tilde{t}^*) \in \bar{Z}^*$ and $\varphi_{\bar{\mathcal{F}}}(\bar{x}^*,\bar{s}^*,\bar{t}^*) \in \tilde{Z}^*$, for any $(\tilde{x}^*,\tilde{s}^*,\tilde{t}^*) \in \tilde{Z}^*$ and $(\bar{x}^*,\bar{s}^*,\bar{t}^*) \in \bar{Z}^*$.

$\diamondsuit$

The latter definition is evidently of difficult application since $\tilde{Z}^*$ and $\bar{Z}^*$ are in general unavailable.

This partially explains why in the MDO literature the authors prefer to handle more qualitative classifications for the reformulations of (4). One of these classifications introduces the following *naive* distinction [3].

- *Structural (or Analytical) Perspective*. A general reformulation of (4) is proposed which meets a suitable *nice* structure. In particular we say that a nice structure is available if the formulation associated to each discipline may be treated independently, by using standard nonlinear programming techniques. Consequently, the MDO reformulation results into a decomposed problem with respect to the disciplines (an overall *globalization strategy* is still necessary but discipline specific codes may be used).
- *Algorithmic Perspective*. The reformulation of (4) must be aimed at using as many results, algorithms and packages as possible, from nonlinear programming. For instance, convex or continuously differentiable reformulations would be in general preferable to respectively nonconvex or nonsmooth ones.

By reasoning from a different perspective, another possible criterion to classify MDO reformulations may be given according to the following definition [2].

Definition 3.4: Consider the reformulation $\hat{\mathcal{F}}$ of the nonlinear MDO formulation (4). We say that $\hat{\mathcal{F}}$ is **closed [open]** with respect to the block Γ_i, $i = 1,2,3$, of constraints, if regardless of the optimization algorithm(s) used to solve $\hat{\mathcal{F}}$, the block Γ_i is always **satisfied [not satisfied]**.

$\diamondsuit$

Observe that in the previous definition, the classification criterion does not rely on the optimization technique(s) used to tackle the solutions of the MDO reformulation. From Definition 3.4 we associate to the MDO reformulation $\hat{\mathcal{F}}$ the label

$$\hat{\alpha}\mathrm{D} \ / \ \hat{\beta}\mathrm{DA} \ / \ \hat{\gamma}\mathrm{IC}, \qquad \hat{\alpha}, \hat{\beta}, \hat{\gamma} \in \{\mathrm{O}, \mathrm{C}\}, \tag{5}$$

where the possible entries $\{\mathrm{O}, \mathrm{C}\}$ for $\hat{\alpha}$, $\hat{\beta}$ and $\hat{\gamma}$ stand respectively for '*OPEN*' and '*CLOSE*'.

B. Examples of MDO reformulations from the literature

In this section we describe some MDO reformulations of (4), which are widely adopted in the literature (we will consider examples of them in Section VII). We urge to remark that the structure of both the objective function and the constraints in (4) is clearly dependent on the application in hand. This suggests that a specific reformulation might be suitable for a real problem, while it can be completely inadequate for another application. The following MDO reformulations of (4) are based on the classification induced by Definition 3.4. We highlight that the reformulations (*R4*)-(*R5*) correspond to a bilevel optimization formulation.

(*R1*) *MultiDisciplinary Feasible (MDF).* It is an MDO reformulation of (4) also known as *FIO* or *AIO* [8], which represents the most trivial approach to the solution. It consists of using the implicit function theorem to explicit the vectors $s = s(x)$ and $t = t(x)$ from the Disciplinary Analysis and the Interdisciplinary Constraints. Then, the resulting MDO reformulation reduces to

$$\begin{aligned} \min_{x} \quad & \hat{f}(x, s(x), t(x)) \\ & g_0(x, s(x)) \geq 0 \\ & g_1(x_0, x_1, s(x)) \geq 0 \\ & \qquad \vdots \\ & g_p(x_0, x_p, s(x)) \geq 0, \end{aligned} \tag{6}$$

which may be treated as a nonlinear program depending on the vector of unknowns $x \in \mathbb{R}^n$. As we said, the equality constraints in (4) can be hardly inverted to provide $s = s(x)$, so that the reformulation (6) turns to be quite unusual. According with the pattern (5), the MDF scheme is an OD/CDA/CIC reformulation.

(*R2*) *Simultaneous Analysis and Design (SAD).* Also known with the acronyms *AAO* or *SAND* [22], is the counterpart of MDF. Indeed, now x, s and t must be treated as independent unknowns, so that the overall reformulation to be solved is

$$\begin{aligned} \min_{x,s,t} \quad & \hat{f}(x, s, t) \\ & g_0(x, s) \geq 0 \\ & g_1(x_0, x_1, s) \geq 0 \\ & \qquad \vdots \\ & g_p(x_0, x_p, s) \geq 0 \\ & A_1(x_0, x_1, s_1, t_2, \ldots, t_p) = 0 \\ & \qquad \vdots \\ & A_p(x_0, x_p, s_p, t_1, \ldots, t_{p-1}) = 0 \\ & t_1 = \mathcal{C}_1(s_1) \\ & \qquad \vdots \\ & t_p = \mathcal{C}_p(s_p). \end{aligned} \tag{7}$$

Observe that the number of unknowns for SAD is relatively larger with respect to MDF. From (5) and the Definition 3.4, the SAD scheme is an OD/ODA/OIC reformulation.

(*R3*) *Distribute Analysis Optimization (DAO).* This is an intermediate approach (see Section VII) between the previous two; that is why it is often addressed as the *In Between* [8], [26] reformulation, or alternatively it is the *IDF* approach [8]. Here, a subset of the equality constraints is used to explicit a sub-vector of the unknowns in terms of the remaining variables (i.e., the implicit function theorem may be partially applied). Considering the following partition of vectors $s^T = (\tilde{s}^T \quad \hat{s}^T)$ and $t^T = (\tilde{t}^T \quad \hat{t}^T)$, the resulting optimization problem becomes (for simplicity we have compounded the Disciplinary Analysis and the Interdisciplinary Consistency constraints)

$$\begin{aligned} \min_{x,\tilde{s},\tilde{t}} \quad & \hat{f}\left[x, (\tilde{s}^T \quad \hat{s}^T(x, \tilde{s}))^T, (\tilde{t}^T \quad \hat{t}^T(x, \tilde{s}))^T\right] \\ & g_0\left[x, (\tilde{s}^T \quad \hat{s}^T(x, \tilde{s}))^T\right] \geq 0 \\ & g_1\left[x_0, x_1, (\tilde{s}^T \quad \hat{s}^T(x, \tilde{s}))^T\right] \geq 0 \\ & \qquad \vdots \\ & g_p\left[x_0, x_p, (\tilde{s}^T \quad \hat{s}^T(x, \tilde{s}))^T\right] \geq 0 \\ & A\left[x, (\tilde{s}^T \quad \hat{s}^T(x, \tilde{s}))^T, (\tilde{t}^T \quad \hat{t}^T(x, \tilde{s}))^T\right] = 0 \\ & \tilde{t} = \tilde{\mathcal{C}}(\tilde{s}). \end{aligned} \tag{8}$$

Finally observe that the DAO scheme is an OD/CDA/OIC reformulation.

(*R4*) *Optimization by Linear Decomposition (OLD).* This is a true *bilevel reformulation* of (4) [14]. Indeed, the first (*upper*) level of minimization (the master level) has the role of coordinating the results coming from the second (*lower*) level of minimization, which is the disciplines

level. The resulting overall nonlinear program is

$$\begin{aligned}
\min_{x_0,t} \quad & \hat{f}\left[x_0, x_1, .., x_p, s_1(x_0, x_1, t), .., s_p(x_0, x_p, t)\right] \\
& g_0\left[x_0, x_1, .., x_p, s_1(x_0, x_1, t), .., s_p(x_0, x_p, t)\right] \geq 0 \\
& m_i(x_0, x_i, t) \leq 0, \qquad i \leq p \\
& \min_{x_i} \quad m_i(x_0, x_i, t) \\
& \qquad t_i = C_i\left[s_i(x_0, x_i, t)\right], \qquad i \leq p
\end{aligned} \tag{9}$$

where

$$m_i(x_0, x_i, t) = \left\| g_i^+\left[x_0, x_i, s_i(x_0, x_i, t)\right]\right\|^2 \tag{10}$$

and

$$g_i^+\left[x_0, x_i, s_i(x_0, x_i, t)\right] \quad = \quad \min\left\{0, g_i\left[x_0, x_i, s_i(x_0, x_i, t)\right]\right\},$$

and the last equality is intended componentwise. Observe that here the sub-vector $s_i(x_0, x_i, t)$ is supposed to be computed by the implicit function theorem, applied to the i-th block of MDA constraints (i.e., $A_i(x_0, x_i, s_i, t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_p) = 0$). The function $m_i()$ (the so called *discrepancy function* [4]) in both the upper and lower level of (9) substantially measures a penalization for infeasible solutions. Note that the exponent 2 in (10) is introduced in order to yield a continuously differentiable objective function, for the lower level.

(*R5*) *Collaborative Optimization (CO).* Similarly to OLD, CO is a multilevel optimization reformulation [7]. Here, the different role played by the system level and the disciplines level is strongly remarked. In particular, we allow the dependency of the Interdisciplinary Constraints from both the design variables and the state variables. The overall nonlinear program is described by introducing the so called *surrogates* $y_1, \ldots, y_p$ of vector x_0. Observe that for each discipline, the latter unknowns are used to de-couple the upper level and the lower level, i.e. they play a role similar to that of vector t.

$$\begin{aligned}
\min_{x_0,t} \quad & \hat{f}(x_0, x_1, \ldots, x_p, t) \\
& \left\| t_i - C_i\left(y_i - x_0, s_i(y_i, x_i, t)\right)\right\|_* = 0, \quad i \leq p \\
& \min_{y_i, x_i} \quad \frac{1}{2}\left[\|y_i - x_0\|^2 + \|s_i(y_i, x_i, t) - t_i\|^2\right] \\
& \qquad g_i\left(y_i, x_i, s_i(y_i, x_i, t)\right) \geq 0, \quad i \leq p.
\end{aligned} \tag{11}$$

The reformulation (11) strongly requires that the sub-vector $s_i = s_i(y_i, x_i, t)$ can be computed, by applying the implicit function theorem to the i-th block of MDA constraints $A_i(y_i, x_i, s_i, t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_p) = 0$. As reported above, the bilevel structure of CO may be fruitfully exploited by using suitable nonlinear programming techniques [14]. Finally, the choice of the norm '$*$' is substantially arbitrary; however, common choices are '$*$'$= 1$ (CO_1) and '$*$'$= 2$ (CO_2).

With the choice '$*$'$= 1$ in (11), the constraints of the upper level are not differentiable. This implies that the Lagrange multiplier rule may fail [39]. On the other hand, The choice '$*$'$= 2$ is appealing because it gives a smooth feasible region of the upper level in (11). Unfortunately, in case the feasible region of the upper level is open, the KKT optimality conditions may fail, since the Jacobian matrix (of the upper level constraints) vanishes in any feasible point. Thus, the Lagrange multiplier rule [6] may fail as well.

IV. A GENERALIZED PSO SCHEME FOR GO

As described in Section II, PSO is an iterative heuristics for the solution of (1). It generates subsequences of points in $\mathbb{R}^n$ which possibly converge eventually to a stationary point of $f(x)$ (see also Section VIII).

At the current iteration k the PSO algorithm generates the P sequences $\{x_j^k\}$, $j = 1, \ldots, P$, according with (see [10]):

$$\begin{aligned}
v_j^{k+1} &= \chi\left[w^k v_j^k + c_j r_j (p_j^k - x_j^k) + c_g r_g (p_g^k - x_j^k)\right], \\
x_j^{k+1} &= x_j^k + v_j^{k+1}.
\end{aligned} \tag{12}$$

PSO is in the wide class of *evolutionary algorithms* and follows the natural paradigm of a bird flock, where the trajectories of the birds (so called *particles*) are represented by the P sequences $\{x_j^k\}$. On the other hand, the vector $v_j^k \in \mathbb{R}^n$ represents the so called *speed* of the j-th particle at iteration k. Finally, the n-real vectors p_j^k and p_g^k, for any k, satisfy the conditions

$$\begin{aligned}
& 1) \quad p_j^k \in \{x_j^\ell\} && \ell \leq k,\ j = 1, \ldots, P, \\
& 2) \quad f(p_j^k) \leq f(x_j^\ell) && \forall \ell \leq k,\ j = 1, \ldots, P,
\end{aligned} \tag{13}$$

and

$$\begin{aligned}
& 1) \quad p_g^k \in \{x_1^\ell, \ldots, x_P^\ell\} && \ell \leq k, \\
& 2) \quad f(p_g^k) \leq f(x_j^\ell) && \forall \ell \leq k,\ \forall j = 1, \ldots, P.
\end{aligned} \tag{14}$$

Furthermore, $\chi, w^k, c_j, r_j, c_g, r_g$ are real bounded coefficients. Observe that we use the subscript j to indicate the subsequence, while the superscript k indicates the iterate in the subsequences $\{x_j^k\}$. Also note that p_j^k represents the 'best position' in the j-th subsequence, while p_g^k is the 'best position' among all the subsequences. The choice of the coefficients is often problem dependent; however, several values for them were proposed in the literature [13], [36], [42], [10]. In particular, the parameters r_j and r_g are often random parameters with uniform distribution in $[0, 1]$.

Observe that in relation (12) the speed v_j^{k+1} depends only on the vectors $p_j^k - x_j^k$, $p_g^k - x_j^k$. However, for the j-th particle an

obvious generalization of (12) could be thee following [10]

$$\begin{aligned} v_j^{k+1} &= \chi_j^k \left[w_j^k v_j^k + \sum_{h=1}^{P} c_{h,j} r_{h,j} (p_h^k - x_j^k) \right], \\ x_j^{k+1} &= x_j^k + v_j^{k+1}, \end{aligned} \tag{15}$$

where the speed v_j^{k+1} depends on the P vectors $p_h^k - x_j^k$ (see also [31]), $h = 1, \ldots, P$.
Now, assuming $\chi_j^k = \chi_j$ and $w_j^k = w_j$, for any $k \geq 0$, the iteration (15) is equivalent to the *discrete stationary (time-invariant) system*

$$\begin{aligned} X_j(k+1) &= \begin{pmatrix} a_j I & -\omega_j I \\ a_j I & (1-\omega_j) I \end{pmatrix} X_j(k) \\ &+ \begin{pmatrix} \sum_{h=1}^{P} \chi_j c_{h,j} r_{h,j} p_h^k \\ \sum_{h=1}^{P} \chi_j c_{h,j} r_{h,j} p_h^k \end{pmatrix}, \end{aligned} \tag{16}$$

where $a_j = \chi_j w_j$, $\omega_j = \sum_{h=1}^{P} \chi_j c_{h,j} r_{h,j}$ and

$$X_j(k) = \begin{pmatrix} v_j^k \\ x_j^k \end{pmatrix} \in \mathbb{R}^{2n}, \qquad k \geq 0. \tag{17}$$

The sequence $\{X_j(k)\}$ identifies the trajectory of the j-th particle in the real space $\mathbb{R}^{2n}$. In addition, this trajectory can be split into the *free response* $X_{j\mathcal{L}}(k)$ and the *forced response* $X_{j\mathcal{F}}(k)$ (see also [34]). In other words, for any $k \geq 0$, $X_j(k)$ may be rewritten according with

$$X_j(k) = X_{j\mathcal{L}}(k) + X_{j\mathcal{F}}(k), \tag{18}$$

where

$$X_{j\mathcal{L}}(k) = \Phi_j(k) X_j(0), \qquad X_{j\mathcal{F}}(k) = \sum_{\tau=0}^{k-1} H_j(k-\tau) U_j(\tau), \tag{19}$$

and (after few calculations [10])

$$\Phi_j(k) = \begin{pmatrix} a_j I & -\omega_j I \\ a_j I & (1-\omega_j) I \end{pmatrix}^k, \tag{20}$$

$$H_j(k-\tau) = \begin{pmatrix} a_j I & -\omega_j I \\ a_j I & (1-\omega_j) I \end{pmatrix}^{k-\tau-1}, \tag{21}$$

$$U_j(\tau) = \begin{pmatrix} \sum_{h=1}^{P} \chi_j c_{h,j} r_{h,j} p_h^\tau \\ \sum_{h=1}^{P} \chi_j c_{h,j} r_{h,j} p_h^\tau \end{pmatrix}. \tag{22}$$

V. Issues on the convergence of PSO

In this section we partially study conditions to ensure the convergence of PSO algorithm. On this guideline, according with a well known result for discrete linear systems (see [34]), we have the following

Proposition 5.1: Suppose $X_j(k) \in \mathcal{A}$, for any j and k, where $\mathcal{A}$ is a compact set. Then,

$$\lim_{k\to\infty} X_{j\mathcal{L}}(k) = 0, \text{ for any } X_j(0) \in \mathbb{R}^{2n}, \qquad j = 1, \ldots, P, \tag{23}$$

is a *necessary* condition to have

$$\lim_{k\to\infty} X_j(k) = \lim_{k\to\infty} X_{j\mathcal{F}}(k), \qquad j = 1, \ldots, P.$$

♢

In few words, in order to get convergent sequences $\{X_j(k)\}$, regardless of the choice for the initial points $X_j(0)$, $j = 1, \ldots, P$, relation (23) imposes the free response $X_{j\mathcal{L}}(k)$ to be bounded away from zero only for finite values of the index k. It is easy to verify that equivalently, (23) requires that the eigenvalues of the unsymmetric matrix $\Phi_j(1)$ satisfy suitable conditions, which limit the values of the coefficients in (15). Indeed, after few calculations [11] we have that the unsymmetric matrix $\Phi_j(1)$ has at most the two real distinct eigenvalues λ_{j1} and λ_{j2} with

$$\begin{aligned} \lambda_{j1} &= \frac{1 - \omega_j + a_j - \left[(1-\omega_j+a_j)^2 - 4a_j\right]^{1/2}}{2} \\ \lambda_{j2} &= \frac{1 - \omega_j + a_j + \left[(1-\omega_j+a_j)^2 - 4a_j\right]^{1/2}}{2}. \end{aligned} \tag{24}$$

The following result yields (23):

Proposition 5.2: Consider the PSO iteration (15) for any $j \in \{1, \ldots, P\}$, and let $\chi_j^k = \chi_j$, $w_j^k = w_j$, $k \geq 0$. Suppose that for any $j \in \{1, \ldots, P\}$ the eigenvalues λ_{j1} and λ_{j2} in (24) satisfy

$$\begin{aligned} |\lambda_{j1}| &< 1 \\ |\lambda_{j2}| &< 1. \end{aligned} \tag{25}$$

Then, (23) holds.

♢

Unfortunately the latter proposition does not provide results for the following relevant convergence issues:

1) The hypothesis (25) in Proposition 5.2 neither ensures that $\{X_j(k)\}$ is a converging sequence, nor it guarantees that $\{X_j(k)\}$ admits limit points. Indeed, (25) does not avoid possible diverging subsequences of $\{X_j(k)\}$ (i.e. we are not guaranteed that in Proposition 5.1 $X_j(k) \in \mathcal{A}$ for any k).
2) In case the subsequence $\{X_j(k)\}_{\mathcal{K}}$ converges, i.e. $\{X_j(k)\}_{\mathcal{K}} \to X_j^*$, with (see (17))

$$X_j^* = \begin{pmatrix} v_j^* \\ x_j^* \end{pmatrix},$$

the property

$$f(x_j^*) \leq f(x), \qquad \forall x \text{ s.t. } \|x - x_j^*\| \leq \epsilon, \quad \epsilon > 0,$$

may not be satisfied. I.e., x_j^* may fail to be a local minimum of $f(x)$.

The first issue was partially investigated in [25]; here we focus on the second issue. In particular, in the next section we will consider the PSO algorithm for problem (2), then we will describe some derivative-free approaches in order to compute a stationary point for (2).

VI. Issues on box-constrained minimization

Consider the KKT conditions for the box-constrained minimization problem (2), where $f(x)$ is assumed to be continuously differentiable. Observe that now, at a stationary point x^*, the first order optimality conditions for the unconstrained problem (1), i.e.

$$\nabla f(x^*) = 0, \tag{26}$$

may fail [6]. Indeed, the *convexity* of the feasible set $\mathcal{F} = \{x \in \mathbb{R}^n : \ a \leq x \leq b\}$ yields the new optimality conditions [6]

$$\nabla f(x^*)^T (x - x^*) \geq 0, \qquad \forall x \in \mathcal{F}. \tag{27}$$

In particular, setting $a^T = (a_1, \ldots, a_n)$ and $b^T = (b_1, \ldots, b_n)$, the condition (27) may be rephrased into the component-wise conditions

$$\nabla_r f(x^*) = 0, \tag{28}$$

where $\nabla_r f(x^*) \in \mathbb{R}^n$ is the so called *reduced gradient*, whose i-th entry is given by

$$[\nabla_r f(x^*)]_i = \begin{cases} \max\left\{\frac{\partial f(x^*)}{\partial x_i}, 0\right\} & \text{if } x_i = b_i, \\ \min\left\{\frac{\partial f(x^*)}{\partial x_i}, 0\right\} & \text{if } x_i = a_i, \\ \frac{\partial f(x^*)}{\partial x_i} & \text{if } a_i < x_i < b_i. \end{cases}$$

Observe that if $a_i = -\infty$, $i = 1, \ldots, n$, and $b_i = +\infty$, $i = 1, \ldots, n$, the box-constrained problem (2) reduces to (1), so that (28) simply reduces to (26). Finally, we remark that in (2) the constraints are linear, so that the *constraint qualification* conditions are always satisfied. This also implies that a straightforward application of the KKT conditions directly yields (28).

We want to solve (2) by adopting a *globally convergent* algorithm (see Section VIII) which does not use derivatives. Thus, we have to guarantee the satisfaction of condition (28) by means of a suitable derivative-free scheme. We can find several algorithms satisfying (28) in the literature, based on different approaches. In particular we consider algorithms showing both a mature convergence analysis along with satisfactory performance: the *direct search methods*.

According with [38], [25], in the latter class we include derivative-free methods which are simply based on "the ranks of a countable set of function values". We report here below a brief description of some of these methods. In particular we consider iterative methods in the class of *Generating Set Search* (GSS), where at each iteration a suitable set of search directions is considered, in order to guarantee a decrease of the function $f(x)$.

- *Pattern Search-based methods*. Observe that the partial derivatives $\partial f(x^*)/\partial x_i$ substantially provide information on $f(x)$ along the coordinate axes. When the derivatives are unavailable, a suitable alternative is to consider a GSS. The directions of the generating set are cyclically exploited, in order to choose descent directions. Then, suitable samplings of the function $f(x)$ along the latter directions ensure that either the conditions

$$\liminf_{k\to\infty} \|\nabla_r f(x^*)\| = 0, \qquad \text{or} \qquad \lim_{k\to\infty} \|\nabla_r f(x^*)\| = 0,$$

eventually holds (see [37], [29]). In particular, the presence of the convex feasible set $\mathcal{F} \subset \mathbb{R}^n$ in (2), imposes that if the current point x_k does not satisfy the condition $\nabla_r f(x^*) = 0$ (i.e. finite convergence is not achieved), then the set of search directions must contain a *feasible* descent direction d. The latter condition equivalently imposes that moving from x_k along d, with sufficiently small stepsizes, then the function $f(x)$ decreases and the points generated are *feasible*. This ensures that the global convergence for pattern search methods can be proved. The local convergence analysis of pattern search methods has also been fruitfully combined with evolutionary techniques, in order to provide globally convergent algorithms. On this purpose, examples of combined methods where evolutionary strategies and pattern search schemes yield globally convergent algorithms, can be found in [19], [20], [21], [41].

- *Linesearch-based derivative-free methods*. These methods are based on the results reported in [27], [29], and have the following distinguishing features:
  1) they use the unit vectors of the coordinate axes as a GSS;
  2) if a suitable feasible descent direction $\hat{d}$ is selected in the GSS, then a linesearch procedure is performed along $\hat{d}$ (see the **Expansion Step** described in Table II);
  3) at the current iteration k, a local model of $f(x)$ is implicitly considered using the information collected by the algorithm, both in the previous iterations and at the current one.

  The theoretical results proved for the linesearch-based methods are similar to those provided by pattern search methods. However, linesearch-based schemes turn to be more flexible in iteratively generating the points. The latter result is a direct consequence of the fact that both the direction and the steplength are computed by pursuing a 'sufficient decrease' of $f(x)$ (instead of relying on a specific grid of points as in pattern search approaches). The theoretical results of linesearch-based methods for the solution of (2) are summarized in [27]. The latter paper proposes the globally convergent algorithm PLA,

for the box-constrained problem (2).

In this paper we suitably modify PLA, in order to combine that approach with PSO. The final result is the algorithm PLA-PSO reported in Table I. As we described in Section V, the scheme PLA is used within PLA-PSO to guarantee the global convergence properties (i.e. the *effectiveness*), which are not provided by PSO. On the other hand, PSO is used (see Table III) to improve the *efficiency* of PLA-PSO, and to avoid a slow convergence.

At **Step 0.** (see Table I) some initializations are introduced, including the choice of the steplengths $\tilde{\alpha}_1^i$, which provisionally estimate the steplength along the search directions. The latter initialization is necessary since the algorithm PLA-PSO is tailored for the solution of (2), where possibly for some $i \in \{1, \ldots, n\}$, $a^i = -\infty$ or $b^i = \infty$.

At **Step 1.** we indicate $x_k = (x_k^1, \ldots, x_k^n)^T$, and we refine the value of the provisional steplength along the current direction d_i, so that also feasibility is preserved.

At **Step 2.** we substantially check whether the choice of the steplength α is *worth* for convergence. Moreover, at **Step 3.** the procedure Expansion Step() attempts to increase the value of the steplength, while preserving a reduction of the function value. Finally, at **Step 5.** the procedure PSO_procedure() is called in order to perform an improvement of the current point $\tilde{x}_{k+1}$. In particular, the latter procedure aims to integrate a *global search* in the *local scheme* developed at **Steps 1.-4.** We remark that the algorithm PLA-PSO, as long as the procedure PSO_procedure(), does not require the feasible set $\mathcal{F}$ to be bounded (i.e., it may happen that for some $i \in \{1, \ldots, n\}$, either $a^i = -\infty$ or $b^i = \infty$). As we said, this gives reasons for the initialization of the coefficients $\tilde{\alpha}_1^i$, $i \in \{1, \ldots, n\}$, at **Step 0.**, which otherwise could be set as $\tilde{\alpha}_1^i = b^i - a^i$, $i \in \{1, \ldots, n\}$.

The Theorem 6.1 summarizes the theoretical results for PLA-PSO.

Theorem 6.1: Let the function $f(x)$ in (2) be bounded from below on the feasible set $\mathcal{F} = \{x \in \mathbb{R}^n : a \leq x \leq b\}$. Let $\{x_k\}$ be the sequence produced by algorithm PLA-PSO. Then, every limit point of $\{x_k\}$ is a stationary point for PLA-PSO, i.e.

$$\liminf_{k \to \infty} \|\nabla_r f(x^*)\| = 0.$$

Proof

The proof straightforwardly follows from Proposition 3.2 in [27].

$\diamondsuit$

Step 0. Choose $x^0 \in \mathbb{R}^n$, $\theta \in (0,1)$, $\tilde{\alpha}_1^i \leq b^i - a^i$, $d_i = e_i$, $i = 1, \ldots, n$. Set $k = 1$, $i = 1$, $h_k = 1$.

Step 1. If $d_i = e_i$ then set $\alpha_{\max} = b^i - x_k^i$, $\alpha_{\max}^- = x_k^i - a^i$, Else set $\alpha_{\max} = x_k^i - a^i$, $\alpha_{\max}^- = b^i - x_k^i$. If $\alpha_{\max}\alpha_{\max}^- = 0$ then set $\alpha = \min\{\tilde{\alpha}_k^i, \max\{\alpha_{\max}, \alpha_{\max}^-\}\}$, Else $\alpha = \min\{\tilde{\alpha}_k^i, \alpha_{\max}, \alpha_{\max}^-\}$.

Step 2. If $\alpha_{\max} > 0$ and $f(x_k + \alpha d_i) \leq f(x^k) - \gamma\alpha^2$ then go to **Step 3.** If $\alpha_{\max}^- > 0$ and $f(x_k - \alpha d_i) \leq f(x^k) - \gamma\alpha^2$ then set $d_i = -d_i$, $\alpha_{\max} = \alpha_{\max}^-$ and go to **Step 3.** Set $\alpha_k = 0$, $\tilde{\alpha}_{k+1}^i = \theta\alpha$, and go to **Step 4.**

Step 3. Compute α_k by the linesearch procedure Expansion Step$(d_i, \alpha, \alpha_{\max})$ and set $\tilde{\alpha}_{k+1}^i = \alpha_k$.

Step 4. Set $\tilde{x}_{k+1} = x_k + \alpha_k d_i$, $\tilde{\alpha}_{k+1}^j = \tilde{\alpha}_k^j$ for any $j \in \{1, \ldots, n\}$ and $j \neq i$.

Step 5. If $h_k \geq n$ then find x_{k+1} with PSO_procedure$(a, b, \tilde{x}_{k+1})$ such that $x_{k+1} \in \mathcal{F}$, Else set $x_{k+1} = \tilde{x}_{k+1}$. If $x_{k+1} \neq \tilde{x}_{k+1}$ then $h_{k+1} = 1$ Else $h_{k+1} = h_k + 1$. Set $i = mod(i, n) + 1$, $k = k + 1$ go to **Step 1.**

TABLE I

THE GLOBALLY CONVERGENT ALGORITHM PLA-PSO.

Expansion Step$(d_i, \alpha, \alpha_{\max})$:

Step 0. Set $\gamma > 0$, $\delta \in (0,1)$.

Step 1. Set $\bar{\alpha} = \min\{\alpha_{\max}, \alpha/\delta\}$. If $\alpha = \bar{\alpha}$ or $f(x_k + \bar{\alpha}d_i) > f(x^k) - \gamma\bar{\alpha}^2$ then set $\alpha_k = \alpha$ and **Stop**.

Step 2. Set $\alpha = \bar{\alpha}$ and go to **Step 1.**

TABLE II

THE LINESEARCH SCHEME Expansion Step FOR PLA-PSO.

VII. NUMERICAL EXPERIMENTS

In order to give numerical examples of the application of the PSO algorithms, a test problem has been set up and solved. The application is related to a real situation affecting the performances of some sailing yachts, with particular reference to the America's Cup Class. For these sailing yachts, stability is substantially enforced by concentrating the large part of the hull weight in a single streamlined body, usually called "keel bulb". In the America's Cup regattas, the yacht proceeds along two directions only: downwind and upwind, and marks are placed in order to obtain this path, according to weather conditions. When the yacht proceeds downwind, the boat is substantially horizontal on the water. On the contrary, when the yacht is going upwind, it is forced to proceed tracing a zig-zag path, and the wind is alternatively coming form port and starboard, with an angle not far from 45 degrees. In this situation, a large side force is acting on the sails, and it tends to capsize the boat. Since most of the hull weight is represented by the keel bulb (typically more than 80% of the total weight), a righting moment is produced whenever the yacht starts heeling. With reference to figure 1 if the keel bulb submergence at rest is L and θ is the heeling angle, the keel bulb weight is responsible for the righting moment $\mathbf{P} \cdot L \cdot sin(\theta)$. The lower the heel angle, the higher the applied force on the sails and the propulsive force. For this reason,

PSO_procedure$(a, b, \tilde{x}_{k+1})$:

Set $q \geq 1$ integer and $P \geq 1$ integer. Set $i = 1$.
Set $a \leq p_h^1 \leq b$, $h = 1, \ldots, P$,
Set $v_j^1 \in \mathbb{R}^n$, $j = 1, \ldots, P$, and
Set $z_1^1 = \tilde{x}_{k+1}$, $a \leq z_j^1 \leq b$, $j = 2, \ldots, P$.
While $i \leq q$
Set χ_j^i, w_j^i, $c_{h,j}$, $r_{h,j}$, $j, h = 1, \ldots, P$.
Compute for $j = 1, \ldots, P$

$$v_j^{i+1} = \chi_j^i \left[w_j^i v_j^i + \sum_{h=1}^{P} c_{h,j} r_{h,j} (p_h^i - z_j^i) \right],$$

$$z_j^{i+1} = Proj_{\mathcal{F}} \left[z_j^i + v_j^{i+1} \right],$$

where $Proj_{\mathcal{F}}[y] = z$ is given by $(\ell = 1, \ldots, P)$

$$z_\ell = \begin{cases} y_\ell & \text{if } a^\ell \leq y_\ell \leq b^\ell \\ a_\ell & \text{if } y_\ell < a^\ell \\ b_\ell & \text{if } y_\ell > b^\ell \end{cases}$$

Set $i = i + 1$.
End While
Set $x_{k+1} = argmin_{\{1 \leq h \leq P,\ i \leq q\}} \{f(p_h^i)\}$.

TABLE III
THE PSO_procedure FOR PLA-PSO.

the submergence of the keel bulb is as high as possible, and the America's Cup Class Rules enforce a maximum depth.

In figure 1, a picture of a classical resulting configuration for this class of yacht is depicted. The bulb is attached to the hull by a long and thin keel fin. The keel fin is also responsible for the side reaction of the hull: elsewhere, the yacht is only able to go along the wind direction. Keel fin is asked to give an high side force without paying it with a large resistance: for this reason, the keel fin is extremely thin. It is now evident how this appendage plays a double (crucial) role: structural and hydrodynamic. The design of such appendages normally do not involve an integration of structural and hydrodynamic computations. In reality, due to the extreme slenderness of the appendage, deformation of the keel fin under loads is not negligible: the sailing team is often able to observe the deformation of the keel fin from aboard when the yacht is heeled. As a consequence, hydrodynamic performances are strongly influenced by the loads, since the shape of the keel fin in motion is different than the shape of the keel fin at rest. A traditional CFD optimization will assume the body as rigid, therefore failing to capture the fin deformation and its unavoidable influence on the hydrodynamic performance of the sailing yacht.
This is a good example of the usefulness of MDO, since a difference of 1% in performances is able to determine the winner and the looser for this competition.

A. Definition of the optimization problem and the Simulation methodology

The shape of the keel fin has been parameterized by using 4 design variables. The modified geometry is obtained by superimposition of a Béziér patch to the original keel fin. Further details of the parametrization technique are given in

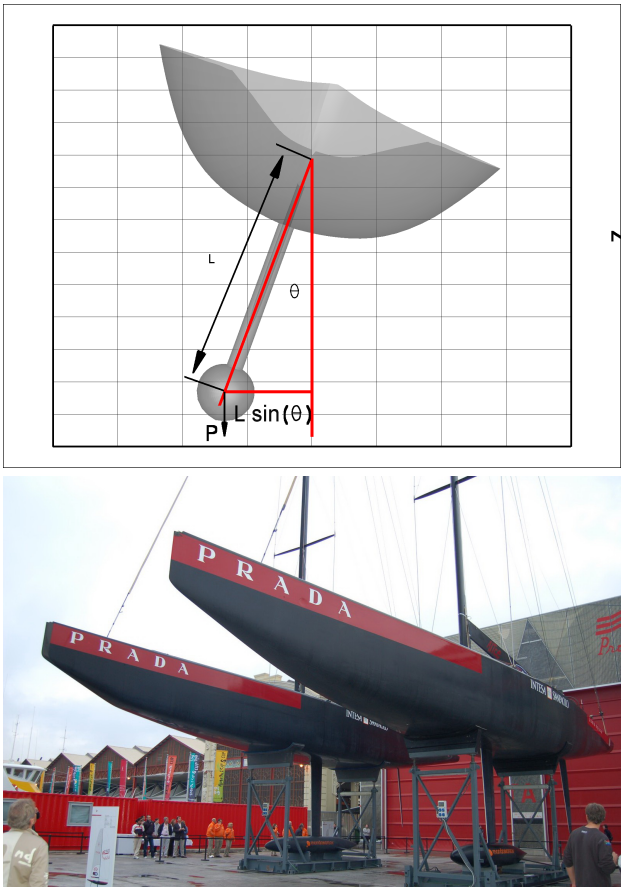

Fig. 1. **On top**: scheme of the righting moment given by the ballast bulb. **On bottom**: Classical configuration of an America's Cup Class Yacht.

[32].

Only box constraints on the 4 design variables have been imposed to the optimization problem, allowing variations $\in [-0.1 : 0.1]$ that correspond to about $[-20cm : +20cm]$ in physical units in the y (transversal) direction.

The original keel fin design has been taken from the available data of the "Il Moro di Venezia" America's Cup sailing yacht [5]. The original keel fin is cylindrical, that is, the section of the fin does not change throughout the span, this feature has been preserved for the optimal keel fin. Objective function is the ratio between the side force and the resistance: it is the inverse of the price to pay for the unit side force. This function has to be maximized.

Two different codes are used for the solution of the structural and hydrodynamic problems. For the hydrodynamic part, a non-linear BEM for potential flows with lifting surfaces and free surface is applied, while a FEM is used for the structural problem. A similar problem has been previously faced in [12]. Since then, some changes in the numerical solvers have been applied. The most important is the application of an open-source code for the solution of the structural problem: references and validation are given in http://www.calculix.de. Moreover, not only the isolated keel fin is now modelled: the whole yacht but the keel bulb are considered in the

hydrodynamic problem. The keel bulb is modelled uniquely during the structural analysis: its weight is applied to the lower part of the keel fin. The objective function is computed for a single hull speed (Fr = 0.30) and for the hull advancing under an heel angle of 20 degrees and a drift angle of 4 degrees. A picture of the pressure on the hull for the original geometry is reported in figure 2. Only the keel fin is considered in the structural problem solution.
In what follows we describe a full MDA analysis (see Section III), without introducing the optimization procedure. As Table 2 will suggest, the unavoidable elasticity of the keel fin introduces a decrease in its efficiency: this simply yields a *multidisciplinary equilibrium* but possibly *not an optimal solution*.

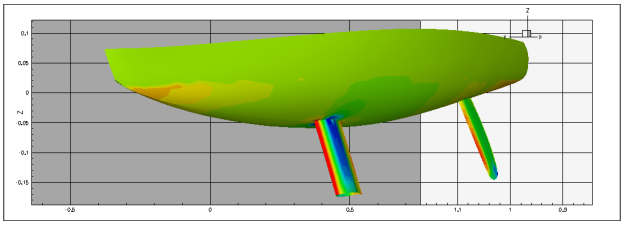

Fig. 2. Pressure on the hull at the speed of Fr=0.3 for the original configuration. Hull, rudder and keel fin are analyzed, while the keel bulb is not included. Red regions indicate high pressure, while blue regions stand for low pressure. Hydrostatic pressure is not here accounted for.

In order to compute the objective function, a first solution of the hydrodynamic problem is produced with the undeformed geometry. The resulting pressure loads are then applied to the keel fin, together with the weight of the keel bulb, and the structural problem is solved. The geometry is now deformed according to the results provided by the structural solver, and a new hydrodynamic computation is performed on the updated geometry. A new set of loads is derived, and the algorithm proceeds. After a number of iterations, a convergent value for loads and deformation will be reached, and the objective function is taken from this convergent solution (i.e. the MDA block of Section III is satisfied). A view of the obtained deformed keel fin is reported in figure 4.

The effect of the consideration of deformability is also evident from Table IV. Here the hydrodynamic characteristics of the keel fin assumed as an infinitely rigid body or a deformable body are reported. The resistance is higher and the side force is smaller if deformations are accounted for, with a resulting decrease on the global efficiency of about 10%.

	Resistance	Side Force	Efficiency
Rigid body	13.321	15.903	1.194
Elastic body	13.373	14.639	1.095

TABLE IV

VARIATION OF EFFICIENCY OF THE KEEL FIN AS A CONSEQUENCE OF THE DEFORMATION UNDER LOADS. FORCES ARE IN KGF, MODEL SCALE.

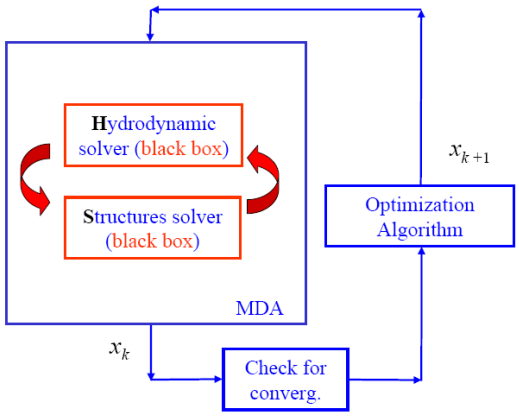

Fig. 3. Our MDO scheme.

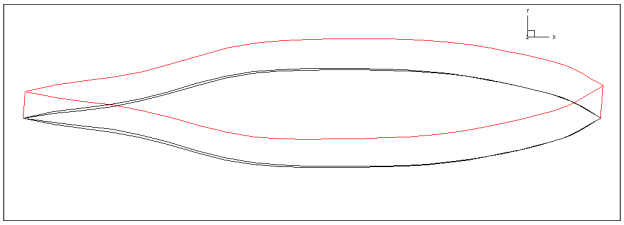

Fig. 4. Maximum deformation of elastic keel (red line), with respect to the rigid one (black line), at the junction with the keel bulb.

B. Algorithmic scheme for the MDA convergence

In order to stop the MDA loop, a quantity able to give an evidence of the degree of coupling between the disciplines has to be defined. According to [12], this quantity has been obtained by measuring the discrepancies in the hydrodynamic prediction between two successive iterations of the MDA. We monitor the ratio

$$\rho = \frac{f(\mathbf{x_k}) - f(\mathbf{x_{k-1}})}{f(\mathbf{x_k})}$$

where $f(\mathbf{x_k})$ is the objective function value at the kth step of the disciplinary convergence iteration and $f(\mathbf{x_{k-1}})$ is the objective function at the previous step, and $\mathbf{x_k}$ are the design unknowns. The coupling between the disciplines is considered satisfactory if ρ is lower than a selected value ρ_0. A different degree of coupling between the disciplines can be obtained by varying the value of ρ_0. If ρ_0 is small, a high degree of coupling is asked. The latter MDO approach is a SAD scheme (see Section III).

C. A SAD/MDF scheme for the acceleration of the MDA

Observe that the algorithm does not change significatively the investigated geometries when it is close to convergence. This feature can be used in order to enforce implicitly the convergence of the disciplinary variables too: instead of starting from the undeformed geometry for each single MDA, and looking directly for an accurate degree of coupling, we can

assume as the initial value of the deformation the same scheme as from the end of the previously performed MDA, and wait the convergence of the optimization algorithm in order to gain an high degree of coupling between the disciplines. In fact, if a nearly-correct initial condition for the disciplinary variables is applied, we can obtain a very small value of ρ just from the start of the MDA. This approach can be defined as SAD/MDF, since it speeds up the *multidisciplinary feasibility* with respect to the SAD scheme described above.

When the change in the two successively tested geometries is not large, this assumption is helpful: if the hypotheses of continuity of the deformation with the design variable variation holds, once the parameters start stabilizing also the convergent local deformation does not change significatively. On the contrary, it could result in an increase of the iterative process for the single MDA. What has been verified in [12] is that the total number of calls to the solvers is decreased by this approach when a local optimization algorithm is adopted, since it naturally produces a sequence of geometries not far from each other. Otherwise, when largest steps in the design variable space are produced during the optimization algorithm iterate, the effectiveness of this approach is not fully demonstrated. Moreover, if the degree of coupling is not high, there might be a substantial difference in the results obtained by different optimization algorithms, if the coupling among disciplines is relaxed.

In our global optimization framework we introduce an iterative procedure which is used to select, in the feasible domain, the most promising designs.
Let us consider our PSO algorithm described in Section IV: here the different individuals in the swarm are moving along individual trajectories, whose speed is self-regulated by the particle position x_j^k, according to (15). Hence, we perform an MDA analysis for each individual of the swarm, relying on the concrete possibility that the state unknowns s (see (3) did not change significantly with respect to the previous iteration.

D. Comparison between local and global strategies

As a first step, it is essential to verify the usefulness of the proposed strategy in this contest. In figure 5 we report a numerical experience using a SAD scheme, which proves that both the optimization strategies - the simplex and the proposed PSO method - yield an improvement of the objective function (we recall that a maximization is performed). The simplex method was successfully applied in [12], while our PSO algorithm was studied in [10]. Convergence of the simplex method is faster, but the final optimal value found by this method is significantly lower than the real one. Moreover, PSO is much competitive with the simplex method.

Figure 6 report the history of the design variables value during the optimization process. A maximum number of iterations was imposed for both the algorithms. The simplex method reached the convergence and stopped before this limit was outreached finding an optimum $f(\mathbf{x^o}) = 1.449$, but missing a better optimum found by the PSO, $f(\mathbf{x^o}) = 1.542$. We recall that the initial value of the objective function (for the original - but deformable - keel fin) was 1.095. Hence, the improvements have been +32% with the simplex algorithm and +41% with the PSO.

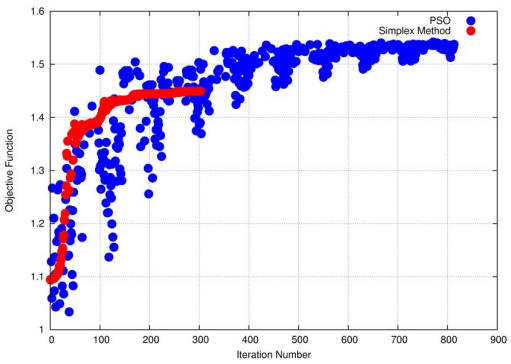

Fig. 5. Comparison of the course of the simplex method (red dots) and PSO algorithm (blu dots) optimization algorithm for the proposed problem using the SAD formulation. On the abscissa, the iteration number is reported. On the vertical axis is reported the value of the objective function for all the particles of the swarm (PSO) and for the vertices of the simplex.

It is evident how the searching strategy is different for the two algorithms. After a transient phase (say $\sim$ 100 iterations), the simplex method generates a monotonic sequence converging to a sub-optimal solution. On the contrary, PSO spreads out a number of elements (the swarm encompasses 16 particles) and generates a sequence which appear as a dumped oscillation.

In the next section we are going to test the performance of PSO for the solution of the SAD/MDF framework.

E. Comparison between SAD and SAD/MDF formulation using PSO

Our emphasis now is on giving evidence of the usefulness and applicability of the SAD/MDF formulation, in order to speed up the process with respect to SAD.

Here, we do not require a strong precision when solving the MDA; nevertheless, when we are approaching the optimal solution, very similar designs are generated by the algorithm. This in turn implies that if, from the iteration k to $k+1$ the quantity $\|x_k - x_{k+1}\|$ is relatively small, then also $\|s_k - s_{k+1}\|$ is accordingly small. Thus, the number of iterations in order to get the convergence of MDA, is possibly reduced, hence we are not far from the solution. As a first check, a comparison between SAD and SAD/MDF formulation with the same level of requested convergence in the coupling parameter has been tested by posing $\rho_0 = 10^{-2}$. Result are shown in figure 7, where the value obtained for the convergence ρ is reported versus the total number of calls of the (hydrodynamic and structural) disciplinary solvers, hence representing the convergence history of each of the MDA performed.

For the SAD algorithm (blue dots), results concentrates around two different levels: around $\rho = 0.05$ and ρ=0.01 (that is, the around prescribed value for ρ_0). It is worthwhile

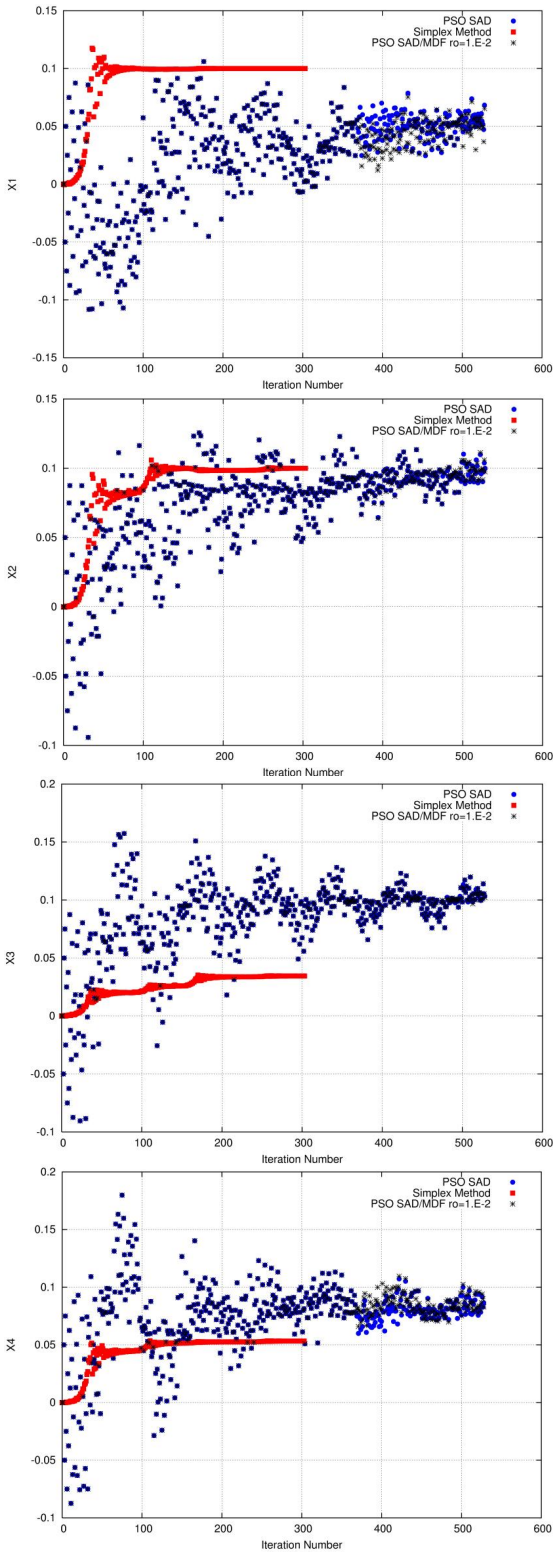

Fig. 6. Comparison of the course of the *simplex method* and the *PSO algorithm*, applied for the solution of the proposed SAD formulation. On the abscissa, the iteration number is reported. The vertical axis shows the value of each of the four design variables respectively. Also the results obtained by using PSO for the SAD/MDF formulation are reported.

to recall the procedure: when the SAD algorithm performs a MDA, the initial values of the state variables s are set to 0, i.e. the SAD starts each time from the undeformed geometry: a first convergence is obtained with a high value of ρ (corresponding to the higher band in figure 7). When a second iteration is performed, the convergence is obtained (the second band is indeed lower than the tolerance value ρ_0 , set in this case equal to 10^{-2}. With the SAD/MFD method instead, once the population has been evaluated for the first time, each particle has its own initial deformed shape: as a consequence, the first value of ρ obtained at each MDA is lower than the corresponding SAD value. This means that the convergence of the MDA will be faster and that the degree of the disciplinary coupling in SAD/MDF is higher. Furthermore, the number of total calls decreases, passing from 1089 (SAD) to 610 (SAD/MDF) (-44%), reducing the wall-clock time of the entire optimization process.

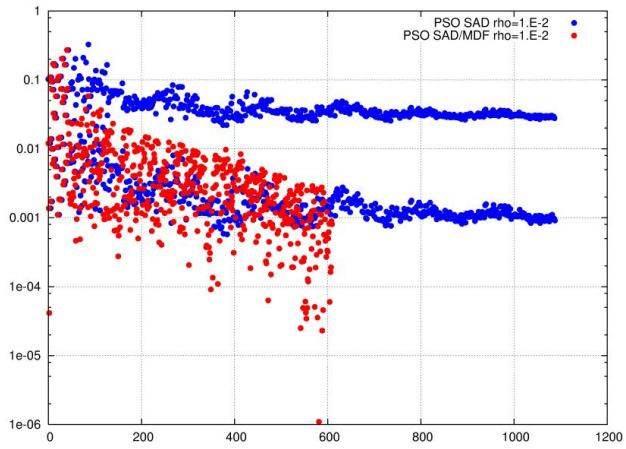

Fig. 7. Coupling parameter during the MDA iteration for the SAD and SAD/MDF algorithm. On the horizontal axis, the counter of the number of calls to the solvers is reported, while the vertical axis shows the actual degree of coupling between the disciplines. Even tought the required accuracy is poor (10^{-2}), the SAD/MDF algorithm gives an higher degree of coupling as a consequence of the convergence of the optimization algorithm.

The figure 8 is reports this tendency. Here, the total number of calls to the solvers is reported as a function of the required coupling accuracy ρ_0. The saving on the number of solver calls seems to be pretty constant even when the required accuracy is varied.

From figure 6 is also quite evident how little discrepancies in the convergent values of the design variables are found when comparing SAD and SAD/MDF algorithm results. On the other hand, no real differences in the objective function value are detected, as reported in figure 9. This means that, from the standpoint of the obtained improvements and final shape, differences between the two algorithms are negligible. The computational advantage in adopting the SAD/MDF algorithm is hence extremely promising.

F. Optimization results

In figure 10 we report the shape of the sections for the original and for several different optimized keel fins, using

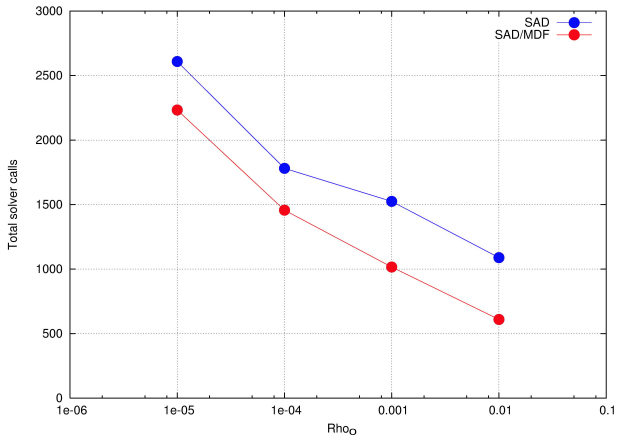

Fig. 8. Coupling parameter during the MDA iteration for the SAD and SAD/MDF algorithm. On the horizontal axis, the required level of accuracy for the discipline couplig, on the vertical axis, the total number of calls to the solvers is reported.

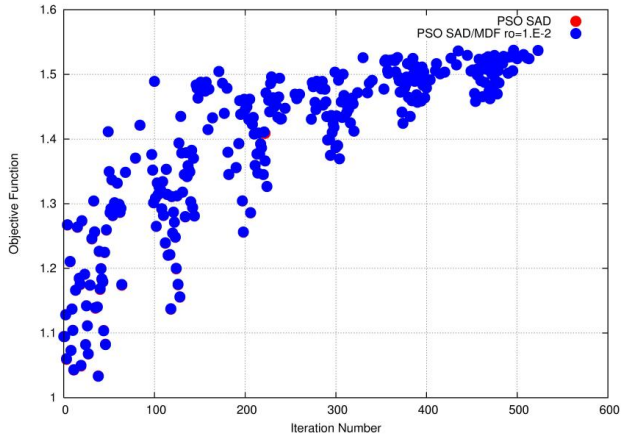

Fig. 9. Comparison of the course of the SAD and SAD/MDF algorithms. On the abscissa, the iteration number is reported. On the vertical axis, the value of the objective function. The agreement between the two algorithms is almost complete, hence the two set of solutions are almost overlapped.

both the SAD and the SAD/MDF algorithms and solving the optimization problem with the simplex or with the PSO.

The optimal geometries found by the SAD or the SAD/MDF algorithm are almost coincident, and the difference seems to be made entirely by the two optimization algorithms, the advantage of the SAD/MDF with respect to the SAD being the reduced computational time.

It is also interesting to notice that, as figure 10 shows, both the PSO and the simplex find an optimal shape thicker than the original one (see in figure 4), that in turn leads to a reduced elastic deformation under the effect of the hydrodynamic loads, as can be observed by comparing the lateral bending in figure 10 and 4.

VIII. Conclusions

Optimization problems in which the solution depends on more *disciplines* may be tackled with MDO. In this paper we

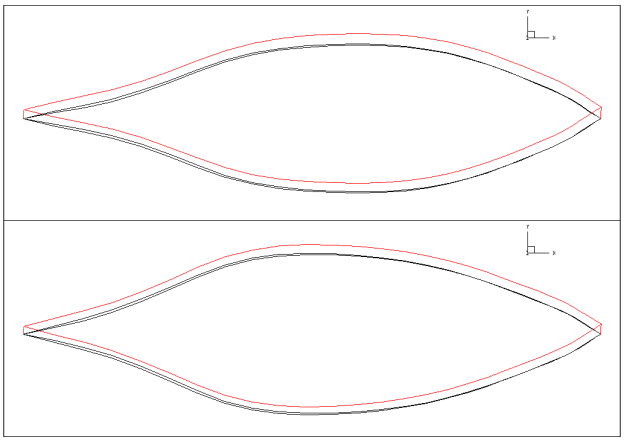

Fig. 10. Comparison of the optimized geometries. SAD and SAD/MDF results obtained using the simplex (top) and the PSO (bottom). The root geometry is black, while the deformation at the tip is reported in red.

presents results for the MDO optimal shape of the keel fin of a sailing yacht, simultaneously accounting for hydrodynamics and elasticity. In the (MDO) framework the keel fin is assumed to be elastic and hence it can be modified by the hydrodynamic loads.

For this problem we have studied and compared different MDO formulations. The optimal design problem is finally tackled considering a Global Optimization (GO) problem within a MDO framework. Some numerical experiments have been performed on a test problem, showing different algorithmic structures for the MDO formulation and how these can be helpful in reducing the total computational cost of the optimization process Moreover, has been also shown how global optimization algorithms can be applied in this context.

Future work will include a more articulated reshaping of the profile (as to the optimizer), the modeling of the keel fin using non-isotropic materials (like carbon fiber) for the structural analysis, and the use of a RANS solver for the hydrodynamic analysis.

Acknowledgements

EFC, DP and AP have been partially supported by the ONR research program NICOP on Global Optimization N000140510617, through Dr. Pat Purtell and by the Programma Ricerca INSEAN 2005-2007. Finally, GF thanks the research program Sicurezza.

REFERENCES

[1] NM Alexandrov, MY Hussaini (Eds.), 1997, *Multidisciplinary Design Optimization - state of the art*, Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization, SIAM Proceedings Series.

[2] NM Alexandrov, RM Lewis, 1999, *Comparative properties of Collaborative Optimization and other approaches to MDO*, Proceedings of the First ASMO UK/ISSMO CONFERENCE on Engineering Design Optimization, July 8-9, 1999, MCB Press.

[3] NM Alexandrov, RM Lewis, 2000, *Algorithmic Perspectives on Problem Formulations in MDO*, AIAA 2000-4719.

[4] NM Alexandrov, RM Lewis, 2000, *Analytical and Computational Properties of Distributed Approaches to MDO*, NASA Langley Research Center, Hampton, Virginia, AIAA 2000-4718.

[5] D Battistin, D Peri, EF Campana, 2005, *Geometry and Resistance of the IACC Systematic Series "Il Moro di Venezia"*, Proc. of 17^{th} Chesapeake Sailing Yacht Symposium.

[6] DP Bertsekas, 1995, *Nonlinear Programming*, Athena Scientific, Belmont, Massachusetts.

[7] RD Braun, 1996, *An architecture for large-scale distributed design*, PhD thesis, Stanford University, Department of Aeronautics and Astronautics.

[8] EJ Cramer, JE Dennis Jr., PD Frank, RMLewis, GR Shubin, 1994, *Problem Formulation for Multidisciplinary Optimization*, SIAM Journal on Optimization, Vol. 4., Issue 4, pp. 754-776.

[9] EF Campana, G Fasano, D Peri, A Pinto, 2006, *Particle Swarm Optimization: efficient globally convergent modifications*, III European Conference On Computational Mechanics - Solids, Structures and Coupled Problems in Engineering, Lisbon.

[10] EF Campana, G Fasano, A Pinto, 2005, *Particle Swarm Optimization: dynamic system analysis for parameter selection in global optimization frameworks*, Technical Report INSEAN 2005-023, Italy.

[11] EF Campana, G Fasano, A Pinto, 2006, *Dynamic system analysis and initial particles position in Particle Swarm Optimization*, IEEE Swarm Intelligence Symposium, Indianapolis

[12] EF Campana, G Fasano D Peri, 2005, *Issues of Non-Linear Programing for Multidisciplinary Design Optimization (MDO) framework*, NuTTS '05 Symposyum, Varna (Bulgaria).

[13] M Clerc, J Kennedy, 2002, *The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space,* IEEE Transactions on Evolutionary Computation, Vol. 6, No. 1.

[14] S Dempe, 2002, *Foundations of Bilevel Programming*, Kluwer Academic Publishers, Dordrecht, Boston, London, The Netherlands.

[15] R De Leone, M Gaudioso, L Grippo, 1984, *Stopping criteria for linesearch methods without derivatives*, Mathematical Programming, No. 30, pp. 285-300.

[16] PC Fourie, AA Groenwold, 2000, *Particle Swarms in Size and Shape Optimization,* Proceedings of the International Workshop on Multidisciplinary Design Optimization, Pretoria, South Africa, pp. 97-106.

[17] A Griewank, 2000 *Evaluating Derivatives,* SIAM Frontieres in applied mathematics, Philadelphia.

[18] UM Garcia-Palomares, JF Rodriguez, *New sequential and parallel derivative-free algorithms for unconstrained minimization*, SIAM Journal of Optimization, No. 13, pp. 79-96.

[19] WE Hart, 1995, *Evolutionary Pattern Search Algorithms,* Technical Report SAND95-2293, Sandia National Laboratories, Albuquerque.

[20] WE Hart, 1996 *A stationary point convergence theory for evolutionary algorithms,* Foundations of Genetic Algorithms 4, Morgan Kaufmann, San Francisco, CA, pp. 325-342.

[21] WE Hart, 1997 *A generalized stationary point convergence theory for evolutionary algorithms,* Proceedings of the International Conference on Genetic Algorithms, Morgan Kaufmann, San Francisco, CA, pp. 127-134.

[22] RT Haftka, Z Gurdal, MP Kamat, 1990, *Elements of Structural Optimization*, Kluwer Academic Publishers, Dordrecht, The Netherland.

[23] J Haslinger, RAE Mäkinen, 2003 *Introduction to Shape Optimization,* SIAM Advances in Design and Control, Philadelphia.

[24] J Kennedy, RC Eberhart, 1995, *Particle swarm optimization*, Proceedings of the 1995 IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, IV: 1942-1948, 1995.

[25] TG Kolda, RM Lewis, V Torczon, 2003, *Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods* SIAM Review Vol. 45, No. 3, pp. 385-482.

[26] RM Lewis, 1997, *Practical aspects of variable reduction formulations and reduced basis algorithms in Multidisciplinary Design Optimization*, in Natalia M. Alaxandrov, M. Y. Hussaini (Eds.), 1997, *Multidisciplinary Design Optimization - state of the art*, Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization, SIAM Proceedings Series.

[27] S Lucidi, M Sciandrone, 2002, *A derivative-free algorithm for bound constrained optimization,* Computational Optimization and Applications, Vol. 21, pp. 119-142

[28] S Lucidi, M Sciandrone, 2002, *On the global convergence of derivative-free methods for unconstrained optimization*, SIAM Journal of Optimization, No. 13, pp. 97-116.

[29] RM Lewis, V Torczon, 1999, *Pattern search methods for bound constrained minimization* SIAM Journal on Optimization, Vol. 9, pp. 1082-1099.

[30] B Mohammadi, O Pironneau, 2001, *Applied Shape Optimization for Fluids*, Clarendon Press, Oxford.

[31] R Mendes, 2004, *Population Topologies and Their Influence in Particle Swarm Performance,* PhD Dissertation, University of Minho, Departamento de Informtica Escola de Engenharia Universidade do Minho.

[32] D Peri, M Rossetti, EF Campana, 2001, *Design optimization of ship hulls via CFD techniques,* Journal of Ship Research, **45**, 2, 140-149.

[33] JD Pinter, 1996, *Global Optimization in Action. Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications,* Kluwer Academic Publishers, The Netherlands.

[34] PE Sarachik, 1997, *Principles of linear systems,* Cambridge University Press.

[35] Y Shi, R Eberhart, 1998, *Parameter Selection in Particle Swarm Optimization,* The seventh Annual Conference on Evolutionary Computation, 1945-1950.

[36] JF Schutte, AA Groenwold, 2005, *A Study of Global Optimization Using particle Swarms,* Journal of Global Optimization, No. 31, pp. 93-108.

[37] V Torczon, 1997, *On the convergence of pattern search methods,* SIAM Journal on Optimization, No. 7, pp. 1-25.

[38] MW Trosset, 1997, *I know it when I see it: Toward a definition of direct search methods,* SIAG/OPT Views-and-News: A Forum for the SIAM Activity Group on Optimization, No. 9, pp. 7-10.

[39] R Thareja, RT Haftka, 1986, *Numerical Difficulties associated with using equality constraints to achieve multi-level decomposition in structural optimization*, AIAA Paper 86-0854.

[40] G Venter, J Sobieszczanski-Sobieski, 2004, *Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization* Structural and Multidisciplinary Optimization, vol. 26, No. 1-2, pp. 121-131.

[41] AIF Vaz, LN Vicente, *A particle swarm pattern search method for bound constrained global optimization*, to appear in Journal of Global Optimization.

[42] YL Zheng, LH Ma, LY Zhang, JX Qian, 2003, *On the convergence analysis and parameter selection in particle swarm optimization,* Proceedings of the Second International Conference on Machine Learning and Cybernetics, Xi'an.

APPENDIX ON OPTIMIZATION

We use $\mathbb{R}^n$ to indicate the real n-dimensional space. The Euclidean norm of vector x is $\|x\| \doteq < x, x >^{1/2}$, while $\|x\|_\alpha$ represents the general α_norm, $\alpha > 0$. The subscripts identify the different particles in a PSO scheme (see Section IV), while the superscripts indicate the iteration. We denote by I the identity matrix of suitable dimension. The symbol $\{x_k\}_\mathcal{K}$ represents the subsequence $\mathcal{K}$ of the sequence $\{x_k\}$. Finally, we say that the function $f(x)$, $x \in \mathbb{R}^n$, is *continuously differentiable* in $\mathbb{R}^n$, if the n partial derivatives $\partial f/\partial x_i$, $i = 1, \ldots, n$, exist and are continuous in $\mathbb{R}^n$.

Let us consider the following general mathematical programming problem,

$$\min_{x \in \mathcal{A}} \; f(x) \tag{29}$$

where $\mathcal{A} \subseteq \mathbb{R}^n$, $n \geq 1$, $\mathcal{A}$ is the so called *feasible set* and $f : \mathbb{R}^n \to \mathbb{R}^q$ is the *objective function*. Observe that whenever $q = 1$, (29) represents the *minimization of a real functional*, while $f(x) = const.$ for any $x \in \mathcal{A}$ indicates that (29) is a *feasibility problem*. Finally, if $q > 1$, (29) is a *multiobjective* problem, where the simultaneous minimization of q real functionals is claimed.

A *local minimum* of $f(x)$ on the set $\mathcal{A}$ is a point $x^* \in \mathcal{A}$ such that:

$$f(x^*) \leq f(x), \qquad \forall x \in \mathcal{A} \cap B(x^*, \rho),$$

where $B(x^*, \rho) \subset \mathbb{R}^n$ is a ball centered in x^* and radius $\rho > 0$.

Similarly, a *global minimum* of $f(x)$ on the set $\mathcal{A}$ is a point $x^* \in \mathcal{A}$ such that:

$$f(x^*) \leq f(x), \qquad \forall x \in \mathcal{A}.$$

We usually introduce a set of *optimality conditions* for (29), as a set of analytical relations satisfied by the so called *stationary points* of $f(x)$. Several optimality conditions may be defined, according with the information available on $f(x)$ and $\mathcal{A}$. In particular, we distinguish between *first order conditions*, where just the gradient $\nabla f(x)$ and the Jacobian of the constraints are involved, and *second order conditions*, where also the second order derivatives are required. *Karush-Kuhn-Tacker* (KKT) conditions are among the most adopted optimality conditions for nonlinear optimization.

Furthermore, a larger information on the objective function and the constraints (e.g. convexity, Lipschitz continuity, etc.) may yield stronger optimality conditions. Unless we differently specify, when we say that the point $x^* \in \mathcal{A}$ simply satisfies some optimality conditions, we mean that the latter conditions are only *necessary* for x^* to be a local minimum.

Then, we say that an iterative algorithm for solving (29) is *globally convergent* when it generates the (possibly infinite) sequence $\{x_k\} \subset \mathcal{A}$, converging to a *stationary point* of (29), regardless of the choice for the initial point x_0. For instance, when $\mathcal{A} \equiv \mathbb{R}^n$ and $f(x)$ is continuously differentiable, the latter definition corresponds to an algorithm which generates the sequence $\{x_k\}$, such that

$$\liminf_{k \to \infty} \|\nabla f(x_k)\| \to 0.$$

Note that the latter condition may be strengthened under additional assumptions, and possibly we can generate a sequence of points $\{x_k\}$ such that

$$\lim_{k \to \infty} \|\nabla f(x_k)\| \to 0,$$

or even

$$\lim_{k \to \infty} x_k = x^*, \qquad \nabla f(x^*) = 0.$$