INITIAL RESULTS OF AN MDO METHOD EVALUATION STUDY

NATALIA M. ALEXANDROV* AND SRINIVAS KODIYALAM[†]

Abstract

The NASA Langley MDO method evaluation study seeks to arrive at a set of guidelines for using promising MDO methods by accumulating and analyzing computational data for such methods. The data are collected by conducting a series of reproducible experiments. In the first phase of the study, three MDO methods were implemented in the iSIGHT[‡] framework and used to solve a set of ten relatively simple problems. In this paper, we comment on the general considerations for conducting method evaluation studies and report some initial results obtained to date. In particular, although the results are not conclusive because of the small initial test set, preliminary numbers suggest that the performance of the methods tends to be consistent with their predicted theoretical properties.

Key Words: Multidisciplinary Design Optimization, Method Evaluation

AMS Subject Classification: 65K05, 49M37

Introduction

Multidisciplinary Design Optimization (MDO) problems are optimization problems that describe complex coupled engineering systems. The systems are composed of physically interacting subsystems described by *disciplinary analyses*, each of which possesses a certain degree of autonomy but depends on other subsystems via a number of couplings, also known as interdisciplinary variables.

We distinguish *MDO formulations* from *optimization algorithms* in the following way. MDO formulations belong to an area that studies MDO problem definitions, including problem decomposition and integration. To analyze an MDO formulation, one considers such attributes as consistency, equivalence to other formulations, optimality conditions, and sensitivity of solutions to various perturbations. Optimization algorithms are used to solve a particular MDO formulation. It is then appropriate to speak of local convergence rates and of global convergence properties of an optimization algorithm applied to a specific formulation. An analogous distinction exists in the field of partial differential equations. On the one hand, equations are analyzed in terms of regularity, well-posedness, and the existence and uniqueness of solutions. On the other, one considers numerous algorithms for solving differential equations. The area of MDO methods studies MDO formulations combined with optimization algorithms, although at times the distinction is blurred. It is important to note that an optimization algorithm applied to two formulations that are mathematically equivalent in terms of solution sets may exhibit drastically different behaviors [1]. When we say “MDO methods” in this paper, we focus more on formulations, although optimization algorithms play a role as well.

A sizable and ever growing body of methods and their variants has been proposed for solving MDO problems. Yet, there is much speculation, but limited computational or analytical substantiation of practical applicability and algorithmic properties of MDO methods. An informative computational method study—of different scope and intent—was done by Shubin [2]. Haim et al. [3] compare performances of several nonlinear programming software packages on an MDO problem, given one specific formulation. However, in general, a practitioner of MDO has little basis for selecting a method among those appearing in print.

Several ongoing efforts at NASA Langley are aimed at addressing this deficiency. The present paper will acquaint the reader with the initial results of the first phase of a method evaluation study initiated last year. The objectives of the study are as follows:

*Member AIAA. Research Scientist, MDOB, NASA Langley Research Center, Hampton, VA, n.alexandrov@larc.nasa.gov.

[†]Member AIAA. Chief Engineer, Engineous Software, Inc., Morrisville, NC. Research done under NASA Contract L-6317.

[‡]The use of trademarks or names of manufacturers in this report is for accurate reporting and does not constitute an official endorsement, either expressed or implied, of such products or manufacturers by the National Aeronautics and Space Administration.

- Accumulate computational data on the performance of promising MDO methods.

- Compare the practical performance of the methods with their proven or conjectured analytical properties.

- Arrive at a classification of the methods and problems amenable to the methods, as well as a set of guidelines for using specific methods.

- Establish "standards" or guidelines for systematic, easily reproducible, method testing procedures.

We consider the last objective exceedingly important. Numerical results presented in publications about MDO methods are rarely easily reproducible by other researchers. This may be due to legitimate reasons, such as the complexity or unavailability of a particular commercial or proprietary code to a specific user. Nonetheless, the lack of at least a minimal set of easily reproducible results makes comparison, validation, and evaluation of methods extremely difficult, if not impossible, because one can always argue that a particular result is due strictly to an implementation and not to an intrinsic property of a method under consideration. A remedy is, of course, to ascertain that the test can be replicated at least for simple problems, thus providing a basis for legitimate discussions of implementations vs. method properties.

Our objectives present a formidable task, since comparing methods intended for solving even the conventional nonlinear programming problem is notoriously difficult, given the limitations of the problem selection, the implementation, and many other variables. However, our aim is not to declare one method superior to another. Instead, by accumulating computational data, we seek to understand under what circumstances the use of a specific method may be advisable.

In this paper, we comment on testing, in general, and describe our testing procedures and their limitations, in particular. We give detailed results for one test problem, followed by a summary of numerical results for all problems, as well as conclusions available to date.

A record of all tests and numerical results can be found in a forthcoming NASA contractor report [4]. A complete record of all tests, codes, and description files can be found at the NASA Langley Multidisciplinary Optimization Branch (MDOB) method evaluation site accessible via the MDOB homepage (`http://fmad-www.larc.nasa.gov/mdob/MDOB/`).[§] Interested readers are invited to make contributions to this site after familiarizing themselves with the submission requirements.

MDO Methods

During Phase I of the study, we collected numerical data on the Multidisciplinary Feasible method (MDF), the Collaborative Optimization approach (CO), and the Individual Discipline Feasible method (IDF). MDF is a mathematical idealization of the conventional approach to MDO. The nomenclature was introduced by Cramer et al. [5]. MDF was implemented to serve as a baseline result in this study. Antecedents of CO [6] and IDF [5] can be traced to work on optimization of large systems, such as that of Wismer [7] and Lasdon [8]. Both CO and IDF are aimed at solving large problems with a narrow bandwidth of coupling.

To give a brief description of the methods under study, we state the MDO problem as a nonlinear programming problem of the following form:

$$\begin{array}{ll} \text{minimize} & f(x, u(x)) \\ \text{subject to} & h(x, u(x)) = 0 \\ & g(x, u(x)) \geq 0, \end{array} \qquad (1)$$

where, given a vector of design variables x, the state variables $u(x)$ are defined via a block system of equations,

$$A(x, u(x)) = \begin{pmatrix} A_1(x, u_1(x), \ldots, u_N(x)) \\ \vdots \\ A_N(x, u_1(x), \ldots, u_N(x)) \end{pmatrix} = 0,$$

where N is the number of blocks. In the context of MDO, the blocks of the system usually represent the state equations for the disciplinary analyses and the necessary interdisciplinary couplings. The equations are known as the Multidisciplinary Analysis (MDA) system.

Multidisciplinary Feasible Method

The MDF formulation is a conventional method for solving problem (1). It is an example of the variable reduction approach to nonlinear programming, where only the design variables x are used as independent optimization variables. The statement of the problem is unchanged from formulation (1). Therefore, theoretically, the convergence properties of any optimization algorithm applied to MDF are just its

[§] As the paper is going to press, the NASA Langley web site is being reorganized. If the link is inoperative, please search for keywords.

convergence properties with respect to conventional nonlinear programming problems.

At each iteration of the optimization procedure, the design variable vector x is input into the MDA system. The system is then solved for the state vector u, thus reducing the dimension of the optimization problem. The solution of the MDA system, i.e., the multidisciplinary function evaluation, is frequently performed via the block Gauss-Seidel procedure (fixed-point iteration). The MDA system can be, in principle, solved by any method for solving nonlinear equations, such as the Newton's method. One should note, however, that unless a globalization strategy is used, there is no guarantee that a method for solving nonlinear equations will solve the MDA problem from arbitrary initial points.

MDF has been in use ever since nonlinear programming techniques were first applied to engineering optimization problems and, hence, is well understood. It makes full use of the existing analysis codes and gives rise to the smallest optimization problems in terms of the number of design variables. At each iteration of optimization, multidisciplinary equilibrium (or feasibility) is maintained via MDA. The term "Multidisciplinary Feasible" refers to this property. Given a careful statement of a particular problem and a good optimization algorithm applied to the formulation, MDF can be efficient.

The main drawback of MDF is its extreme expense. First, the method is costly to implement, because a practitioner of MDO has to face the difficult problem of analysis integration. Second, a complete MDA must be done not just at every iteration, but also for computing derivatives, if finite-difference derivatives are used. Moreover, optimization algorithms applied to MDF are sensitive to the convergence of MDA, in general, and the fixed-point iteration, in particular, which may be detrimental to the robustness of the method and its speed. Finally, the method is not immune to failing because of attempts to process points that cannot be analyzed.

Collaborative Optimization

In the CO approach, problem (1) is decomposed into a number of subsystems, usually along the disciplinary lines. The problem is then reformulated as a bilevel programming problem.

A system objective f is selected for the system level. The system-level constraints are the so-called "compatibility" or "coupling" constraints designed to bring the system into multidisciplinary equilibrium. The values and derivatives of the compatibility constraints are computed by solving the lower-level, subsystem optimization problems.

The subsystem problems, generally solved in parallel, are nonlinear programming problems whose objective is to minimize the discrepancy between the shared variables of the subsystems while satisfying the disciplinary constraints.

Braun et al. [9] comment on implementation and performance features of CO. A detailed discussion of CO's analytical and computational properties can be found in Alexandrov and Lewis [1]. The latter work contains complete, precise notation of CO and a simplified notation useful in initial implementation. We use the simplified notation here.

Assume that for each disciplinary subsystem i, given a vector of design variables x_i, the corresponding vector of responses $u_i(x_i)$ is computed via the analysis A_i, and the constraints

$$g_i(x_i, u(x_i)) \geq 0, \quad i = 1, \ldots, N,$$

must be satisfied, where each $g_i(x)$ is a vector. The vectors x_i pertaining to disciplines i are not necessarily disjoint subsets of the complete vector of design variables x. Then auxiliary vectors ξ_i and ν_i are introduced to represent the shared components of x_i and u_i at the system level.

The system-level optimization problem of the CO formulation has the form

$$\begin{array}{ll} \text{minimize} & f(\xi, \nu) \\ \text{subject to} & C(\xi, \nu) = 0, \end{array} \tag{2}$$

where $C(\xi, \nu)$ is the system of N compatibility constraints $c_i(\xi_i, \nu_i)$, each one having the form

$$c_i(\xi_i, \nu_i) = \frac{1}{2}[\| \xi_i - x_i^* \|^2 + \| \nu_i - u_i(x_i^*) \|^2],$$

where x_i^* is the solution of the i-th subsystem optimization problem:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}[\| \xi_i - x_i \|^2 + \| \nu_i - u_i(x_i) \|^2] \\ \text{subject to} & g_i(x_i, u(x_i)) \geq 0. \end{array}$$

The actual system compatibility constraint can be stated as a sum of the individual constraints. Other forms of the system-level compatibility constraints and subsystem problems exist ([6], [10], [11], [12]), but in Phase I, we focused on the CO formulation as shown above, as it is the most promising one [1].

CO has a number of attractive features. First, it dispenses with MDA. Instead of requiring multidisciplinary feasibility at each iteration of optimization, the feasibility is attained in the system optimization problem via the introduction of compatibility constraints. Thus, each iteration is feasible with respect to individual analyses, but not multidisciplinary feasible until a solution is reached.

CO has the very appealing attribute of disciplinary autonomy with respect to both the analyses and optimization, resulting in a relatively easy procedure for integration of the disciplinary analyses. The analyses can be processed in parallel. Another significant feature is the maintenance of intradisciplinary feasibility at each system iteration, which is important from the application perspective. As MDF, the method makes full use of the disciplinary analysis software.

The limitations of CO have to do with attaining multidisciplinary feasibility and with reformulating the original, conventional nonlinear programming problem into a nonlinear bilevel programming problem.

In particular, the method is intended for problems where the interdisciplinary coupling has small bandwidth. Problems with many couplings are not expected to benefit from this formulation. The formulation has more optimization variables than that arising from MDF, but given the first limitation, the increase in the number of variables should not be great, as the number of auxiliary variables depends on the bandwidth of the coupling.

Further, the CO formulation is aimed at solving a narrower class of problems than do MDF and IDF; namely, it does not consider general constraints at the system level. Technically, general constraints can be included at the system level. Their inclusion tends to impair the performance of the method, but is unavoidable at times. For instance, in one of our test problems, the hub frame design, the objective is to minimize the weight of the structure, subject to constraints on the translational and rotational displacements, stresses in the frame members, and local buckling of the frame members. The translational and rotational displacements of the frame structure are global responses and should not be treated as local responses at the subsystem level.

Another limitation is that, despite maintaining disciplinary autonomy, CO does not allow explicit optimization with respect to disciplinary objectives at the disciplinary level. Multiple disciplinary objectives have to be incorporated at the system level.

The decomposition procedure may be dictated by the application, but it presents a difficult problem for cases where structure is not well understood.

The "small bandwidth of coupling" feature simply limits the scope of problems amenable to CO. In addition, CO suffers from a difficulty that has to do with reformulating a nonlinear programming problem into a general, nonlinear, bilevel optimization problem. As such, CO is inherently difficult to solve by means of software intended for conventional, single-level, nonlinear programming problems. While the CO formulation is equivalent to the original nonlinear programming problem with respect to the solution sets, the formulation is not equivalent to the original problem with respect to optimality conditions [1]. This is an example of two problem formulations that are equivalent with respect to their solution sets, yet are drastically different numerically.

Based on theoretical analysis of CO [1], it is expected that much fine-tuning would be required to implement the method for a specific problem and that convergence behavior of conventional optimization methods applied to the CO formulation might be erratic.

Individual Discipline Feasible Method

The IDF formulation provides another approach to "breaking" the expensive MDA iteration. The words "Individual Discipline Feasible" refer to maintaining disciplinary feasibility at each optimization iteration, but not multidisciplinary feasibility until a solution is reached.

Various forms of IDF are described by Cramer et al. [5] and Lewis [13]. To state the formulation, let us assume that the system consists of two subsystems and let us write the disciplinary analyses A_1 and A_2 in more detail as follows:

$$A(x, u(x)) = \left(\begin{array}{c} A_1(x, u_1(x), u_{12}(u_2(x))) \\ A_2(x, u_{21}(u_1(x)), u_2(x)) \end{array} \right) = 0,$$

where u_{ij} represent the interdisciplinary flow of information from discipline j to discipline i. Then the IDF formulation introduces auxiliary variables x_{12} and x_{21}, and the optimization problem can be stated in the following form:

$$\begin{array}{ll} \text{minimize} & f(x, u_1(x_1, x_{12}, u_2(x, x_{21}))) \\ \text{subject to} & x_{12} - u_{12}(u_2(x, x_{21})) = 0 \\ & x_{21} - u_{21}(u_1(x, x_{12})) = 0 \\ & h(x, u_1(x_1, x_{12}, u_2(x, x_{21}))) = 0 \\ & g(x, u_1(x_1, x_{12}, u_2(x, x_{21}))) \geq 0, \end{array}$$

where, $u_1(x, x_{12})$ and $u_2(x, x_{21})$ are computed by solving the disciplinary equations

$$\begin{array}{rcl} A_1(x, u_1(x, x_{12}), x_{12}) & = & 0 \\ A_2(x, x_{21}, u_2(x, x_{21})) & = & 0. \end{array}$$

Thus, the IDF formulation is a single-level, nonlinear optimization problem.

On the positive side, IDF is trivially equivalent to the original nonlinear programming problem and is thus easy to analyze. That is, the optimality conditions of the original problem hold for the IDF formulation. The convergence properties of optimization algorithms applied to IDF are those of the algorithms applied to conventional nonlinear programming problems. Given a good solver for equality constrained optimization problems, the method is expected to be efficient.

IDF assures full use of disciplinary analysis software.

Similarly to CO, IDF is intended for problems with small bandwidth of interdisciplinary coupling, and the problem of decomposition is similar to that of CO. Unlike CO, IDF is not limited to problems without general constraints at the system level.

Also similarly to CO, formulations that arise from IDF have more optimization variables that those arising from MDF.

Importantly, although IDF maintains autonomy with respect to analyses, it lacks CO's autonomy with respect to disciplinary optimization. That is, while the analyses are performed autonomously during the analysis stage, the coupling is restored during the optimization step computation. This brings back the difficulties of integration.

Another weakness of IDF is its treatment of the disciplinary constraints. They are either ignored in descriptions of the formulation or simply relegated to the system, despite the need to handle disciplinary constraints at the disciplinary level—a wish usually expressed by practitioners of MDO.

Approach

We selected initial sets of ten problems, three methods, and a set of metrics to record during data collection. During Phase I, all methods were implemented in iSIGHT, which is a computational framework for multidisciplinary design optimization, produced by Engineous Software, Inc. The data have been compiled and preserved for evaluation. The iSIGHT problem description files are available to anyone who would like to duplicate the results or improve on them.

There has been much recent work in the area of approximations for engineering optimization, and Collaborative Optimization has been reported ([14], [15]) to show improvement due to the use of response surface methodology. In this study, we have not attempted to use approximations.

Implementation and Its Limitations

In this study, we are concerned with evaluating MDO methods, which is a notion composed of "formulations" and "algorithms". A formulation is a statement of the problem to be solved by an optimization algorithm. An algorithm is then a set of steps performed to solve a formulation. The two are very much interrelated, and are even difficult to distinguish for some methods. MDF, CO, and IDF are all formulations, but they each may be implemented in a variety of ways and their evaluation will be affected by the use of specific optimization algorithms. This makes the evaluation extremely difficult. However, trends in performance can still be discerned.

The implementation of the methods and the problems, as well as the software used to solve the problems, have a direct bearing on the performance of the methods. Moreover, we understand completely that to use a particular method to its full advantage, especially a method for solving problems as difficult as those that arise in MDO, one will, by necessity, perform a significant amount of fine-tuning, both in the problem statement and in all areas of implementation.

Taken to its extreme, however, this principle means that, given any problem and any initial method, with a sufficient amount of time the method can be "fine-tuned" so that the problem can be solved with reasonable efficiency. The danger of such an extreme is that what practically amounts to a new method has to be developed for each problem. The questions are then whether an original method can still be discerned after such an exercise and just how much of fine-tuning is required to produce reasonable performance.

An important consideration in making decisions about fine-tuning is the available resources for solving a specific problem. If an expensive problem will be solved many times, it makes sense to fine-tune the method and the problem to assure optimal performance. However, in some projects, it is anticipated that the problem under consideration will be solved only a few times due to its extreme expense. In such cases, one cannot afford fine-tuning and one has to be reasonably certain of a method's robustness before its application.

With this in mind, we made a decision to avoid fine-tuning as much as possible in the first phase of the study, in order to obtain an idea of what a reasonably intelligent user, who is not a method developer, will face when implementing a specific method. In further studies, a higher degree of fine-tuning the implementation is planned.

The choice of optimization software for solving the formulated problems and subproblems was limited by the set available in iSIGHT.

Problem	Title	Source
1	Conceptual Ship Design	[14]
2	Electronic Packaging	[16, 17]
3	Power Converter	[18, 16]
4	Speed Reducer	[19, 16]
5	Combustion of Propane	[20, 16]
6	Hear Dipole	[20, 21, 16]
7	Hub Frame Design	[16]
8	Isomerization of α-Pinene - Collocation Formulation	[20]
9	Propane, Isobutane, N-Butane Nonsharp Separation	[22]
10	Three-Component Separation	[22]

Table 1: Test problem set for Phase I.

Problem	1	2	3	4	5	6	7	8	9	10
Method										
MDF										
# Variables	6	8	6	7	4	4	120	80	48	52
# Constraints	7	3	4	11	4	4	764	115	38	40
CO										
System:										
# Variables	11	5	6	2	4	8	40	–	–	–
# Constraints	5	2	6	3	2	6	6	–	–	–
# Subsystems	5	2	2	3	2	2	2	–	–	–
Total # subsystem variables	18	12	12	11	7	12	120	–	–	–
IDF										
# Variables	14	12	12	–	6	8	–	–	–	–
# Constraints	11	5	6	–	6	6	–	–	–	–

Table 2: Problem dimensions.

Within that set, the choice was somewhat subjective and dependent on experience. At this stage, we have not conducted conclusive studies on the sensitivity of a particular formulation to the choice of the optimizer.

All three methods were implemented in the iSIGHT framework, using MDOL, the iSIGHT MDO Language. The implementation was both eased and made more difficult because the methods were tested within the iSIGHT framework. On the one hand, the framework provided a unified approach to testing. On the other hand, iSIGHT is designed for handling large, "production" problems, and there was significant overhead connected with coding smaller problems in the early stages of the study. In addition, we were restricted to using the optimization software available in iSIGHT for solving the optimization subproblems. Overall, we feel that at this initial stage, the benefits of using iSIGHT outweighed the disadvantages of the additional work necessary to incorporate the problems and the methods into the framework.

Implementation within iSIGHT also affected the performance of the methods on some of the problems, in that stand-alone implementation has resulted in a much faster convergence. Future method evaluation studies are planned both in and out of iSIGHT.

Given all the limitations, however, our results are available for examination, as is all the software that produced the results. Should the authors of a particular method or other researchers find that a certain trend we discerned has to do strictly with our implementation and not with the innate properties of that method, we welcome suggestions on how to make the tests more informative.

Finally, promising methods evolve continually, their implementation improves with increased understanding of their workings, and the difficulties apparent at one stage of their development may disappear at a more mature stage.

Problems

The ten problems selected for Phase I of the study are listed in Table 1. Earlier references exist for most of these problems. We indicate the sources we used. The reference for the MDOB Test Suite [16] is indicated for all problems directly downloaded from that set, but the primary reference for the problem is the other reference.

At this stage, the problems are not representative of realistic MDO applications. However, to actually solve a realistic MDO problem may require months or even years, which would somewhat incapacitate the effort. Therefore, we chose several tractable problems as an initial set we could easily handle. Some of the problems, such as the Conceptual Ship Design Problem, the Electronic Packaging Problem, and the Hub Frame Problem, exhibit some of the salient features of realistic MDO problems, albeit not their size, complexity, or expense.

While a method cannot be judged applicable to a realistic problem just because it performs well on a toy problem, if it does not perform well on relatively simple problems, that is an indication that the method must be further studied before attempting to use it for solving realistic problems.

During future studies, we plan to obtain numerical data on solving more realistic problems with the methods that are deemed promising after this initial testing. A summary of the problem dimensions is given in Table 2. In cases where the problems were found inappropriate for being solved by a specific method, the table contains "—".

Note that the constraints of the system-level problem for CO are just the compatibility constraints.

Metrics

We have considered a number of metrics in evaluating the performance of the methods. Some of the metrics are objective and quantitative, the others, more subjective and qualitative. Discussion of some of the metrics follows.

Generality. This is a measure of applicability of a method to various classes of problems. Our problem pool is small at this stage, but as it grows, we will classify the problems by their size in terms of variables and constraints, smoothness, nonlinearity, complexity, strength and bandwidth of coupling, presence of several objectives, and other features.

Robustness. Given that each method is designed for solving a specific class of problems, this metric evaluates the capacity of the method to provide a solution or a "satisfactory" point for that class of problems for a variety of perturbed data. For the initial set of problems that we considered, we measured robustness by attempting solution from several starting points.

Another, somewhat subjective, but important measure of robustness we considered was the amount of fine-tuning required to make a method produce an answer. For instance, we considered restructuring the problems in a number of ways, such as the incorporation of slack or squared slack variables, or changing the tolerances on constraints and optimality.

Performance. Performance is a very complex issue, strongly related to implementation, problem formulation, and the nature of the problem. In Phase I of the study, we quantified performance as *work* performed by each method during every optimization procedure.

The recorded work consists of the total number of analyses, with each "disciplinary" analysis counted as one function evaluation. Function evaluations computed during finite-difference derivative computations are included. In particular, for MDF, we report the total number of analyses, including the average number of analyses taken during fixed-point iterations in order to compute MDA, and those necessary to compute the finite-difference derivatives. For CO, we report the number of iterations taken by the system-level optimization problem, as well as the sum of the function evaluations in each subsystem, including those required for finite-difference computations. For IDF, we report the total number of function evaluations, including those taken for finite-difference computations, times the number of the subsystems.

Although the dimensions of the design space differ for MDF, CO, and IDF, by reporting the total number of analyses, we provide as complete a measure of work as possible, given the serious limitations imposed by a specific implementation of the methods and the problem formulations.

Example: Electronic Packaging

The Electronic Packaging problem ([16], [17]) considers a circuit that comprises two resistors mounted on a heat sink, resulting in two coupled subsystems: electrical and thermal. Operating temperatures affect component resistance, while resistance, in turn, influences the temperature. The objective function is watt density. The constraints require the operating temperatures of resistors to be below a threshold temperature and the currents through the two resistors to be equal. The problem has eight design variables, thirteen state variables, two parameters (*voltage* and

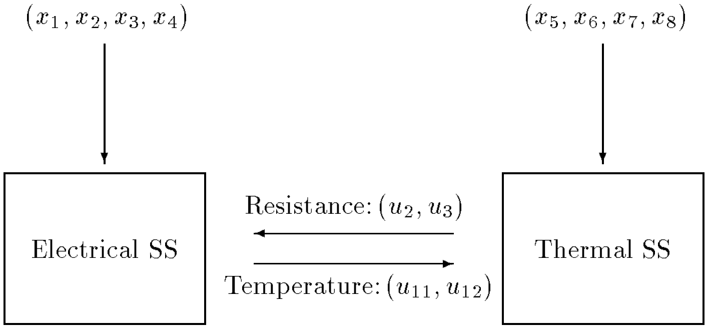

Figure 1: Input-output diagram for the electronics packaging problem.

$T°$), and may be stated as follows:

$$
\begin{array}{ll}
\text{maximize} & u_1 \\
\text{subject to} & h_1 = y_4 - y_5 \\
 & g_1 = u_{11} - 85.0 \leq 0 \\
 & g_2 = u_{12} - 85.0 \leq 0 \\
 & 0.050 \leq x_1 \leq 0.15 \\
 & 0.050 \leq x_2 \leq 0.15 \\
 & 0.010 \leq x_3 \leq 0.10 \\
 & 0.005 \leq x_4 \leq 0.05 \\
 & 10.00 \leq x_5 \leq 1000.0 \\
 & 0.004 \leq x_6 \leq 0.009 \\
 & 10.00 \leq x_7 \leq 1000.0 \\
 & 0.004 \leq x_8 \leq 0.009,
\end{array}
$$

where the first constraint represents the branch current equality, while the next two inequalities are the component reliability constraints.

The eight design variables are as follows: x_1 and x_2 are the heat sink width (m) and length (m); x_3 and x_4 are the fin length (m) and width (m): x_5 and x_7 are the nominal resistance (Ω) of components 1 and 2, respectively, at temperature $T°$; x_6 and x_8 are the temperature coefficients ($°K^{-1}$) of the electrical resistance components 1 and 2.

The state variables are as follows: u_1 is the watt density ($watt/m^3$); u_2 and u_3 are the resistances (Ω) at temperatures T_1 and T_2; u_4 and u_5 are currents ($amps$) in resistors 1 and 2; u_6 and u_7 represent power dissipation ($watts$) in resistors 1 and 2; u_8, u_9, and u_{10} are the total circuit current, resistance, and power, respectively; u_{11} and u_{12} are the component temperatures, T_1 and T_2 (°C), of resistors 1 and 2; and finally, u_{13} is the heat sink volume (m^3).

The states are described by the following equations:

$$
\begin{array}{rcl}
u_1 & = & -u_{10}/u_{13} \\
u_2 & = & x_5(1.0 + x_6(u_{11} - T°)) \\
u_3 & = & x_7(1.0 + x_8(u_{12} - T°)) \\
u_4 & = & u_3 u_8/(u_2 + u_3) \\
u_5 & = & u_2 u_8/(u_2 + u_3) \\
u_6 & = & u_4^2 u_2 \\
u_7 & = & u_5^2 u_3 \\
u_8 & = & voltage/u_9 \\
u_9 & = & (1.0/u_2 + 1.0/u_3)^{-1} \\
u_{10} & = & u_8^2 u_9 \\
u_{11} & = & \text{function}(x_1, x_2, x_3, x_4, u_6, u_7) \\
u_{12} & = & \text{function}(x_1, x_2, x_3, x_4, u_6, u_7) \\
u_{13} & = & x_1 x_2 x_3,
\end{array}
$$

where the states u_{11} and u_{12} are rather involved expressions. The code for this problem can be found at the MDO Test Suite web site accessible from the MDOB Homepage.

The reported results are for cases initiated from the same starting points, for all methods.

Fig. 1 depicts interdisciplinary interactions in the problem.

The following three subsections deal with solving the problem by using the three methods under study.

The following conventions are used in all data tables.

- "Convergence" means the attainment of a point that satisfies the Karush-Kuhn-Tucker (first-order necessary) conditions for a critical point of the problem within the maximum allowable number of iterations (set at $10,000$). However, when a particular problem was solved, known solutions were always found rather than just critical points of the problem.

- "Work" is defined as the total number of disciplinary analyses performed. For MDF, work is equal to the number of calls to the function evaluation procedure times the average number of fixed point iterations per MDA times the number of "disciplines". For CO, the total number of function evaluations at the subsystem level and the number of system iterations is reported. For IDF, we report the number of disciplinary function evaluations times the number of "disciplines".

  Function evaluations done during finite-difference computations are included in all work.

- The superscript "F" added to the value of the final objective indicates failure to converge to a critical point within the allowable number of iterations.

- In all tables, "–" means that the problem was found inappropriate for a particular method, and the tests were not run.

MDF Implementation

The MDF formulation has eight design variables, three explicit constraints, and its statement is that of the original problem. The problem was solved using the method of feasible directions in iSIGHT. The derivatives were computed using finite differences with the step size of 1%. The termination criteria included the satisfaction of the first-order necessary conditions for optimality, tolerances for the absolute and relative change in the objective function during several successive iterations, and the maximum number of allowable iterations. The MDF results are summarized in Table 3.

CO Implementation

For the CO approach, the system-level optimization problem is given by

$$\begin{array}{ll} \text{maximize} & \xi_1 \\ \text{subject to} & c_1 \leq 0.0001 \\ & c_2 \leq 0.0001. \end{array}$$

The system-level problem has five design variables: ξ_1, ξ_2, ξ_3, ξ_{11}, and ξ_{12}.

The Thermal Subsystem optimization problem is:

$$\begin{array}{ll} \text{minimize} & c_1 \\ \text{subject to} & h_1 = 0.0 \\ & g_1 = u_{11} - 85.0 \leq 0.0 \\ & g_2 = u_{12} - 85.0 \leq 0.0, \end{array}$$

where

$$\begin{array}{rcl} c_1 & = & (u_{11} - \xi_{11})^2 + (u_{12} - \xi_{12})^2 + (u_2 - \xi_2)^2 \\ & + & (u_3 - \xi_3)^2 + (u_1 - \xi_1)^2, \end{array}$$

where the variables of the Thermal Subsystem are $x_i, i = 1, \ldots, 4$, u_2, and u_3.

The Electrical Subsystem optimization problem is:

$$\begin{array}{ll} \text{minimize} & c_2 \\ \text{subject to} & g_1 = u_{11} - 85.0 \leq 0.0 \\ & g_2 = u_{12} - 85.0 \leq 0.0, \end{array}$$

where

$$c_2 = (u_2 - \xi_2)^2 + (u_3 - \xi_3)^2 + (u_{11} - \xi_{11})^2 + (u_{12} - \xi_{12})^2,$$

where the variables of the Electrical Subsystems are $x_i, i = 5, \ldots, 8$, u_{11}, and u_{12}.

The system-level optimization problem was solved by a combination of the exterior penalty function method and the method of feasible directions available in iSIGHT. The subproblems were solved using a sequential quadratic programming algorithm–DONLP–implemented in iSIGHT. The CO results are summarized in Table 4.

IDF Implementation

The compatibility constraints of the IDF formulation were implemented as inequalities, since that seemed to produce better results, given the optimization software available in iSIGHT. The optimization problem is:

$$\begin{array}{ll} \text{maximize} & u_1 \\ \text{subject to} & c_1 \leq 0.0001 \\ & c_2 \leq 0.0001 \\ & h_1 = u_4 - u_5 = 0.0 \\ & g_1 = u_{11} - 85.0 \leq 0.0 \\ & g_2 = u_{12} - 85.0 \leq 0.0, \end{array}$$

where the twelve design variables are $x_i, i = 1, \ldots, 8$, and four coupling variables ξ_2, ξ_3, ξ_{11}, and ξ_{12}.

The Thermal Subsystem evaluates u_1, h_1, and

$$c_1 = (u_{11} - \xi_{11})^2 + (u_{12} - \xi_{12})^2,$$

while the Electrical Subsystem evaluates

$$c_2 = (u_2 - \xi_2)^2 + (u_3 - \xi_3)^2.$$

The optimization problem was solved by a combination of the exterior penalty function method and the method of feasible directions available in iSIGHT. The IDF results are summarized in Table 5.

Initial	Initial	Initial	Final	Final	
Point	Objective	Max. Con. Viol.	Objective	Max. Con. Viol.	Work
1	7.7944D+01	2.1663D-08	6.3972D+05	1.2188D-03	83 × 3 × 2 = 498
2	6.8363D+03	-2.8956D-01	6.3972D+05	1.2188D-03	44 × 3 × 2 = 264
3	1.5111D+03	-4.2924D-02	6.3654D+05	1.4514D-03	44 × 3 × 2 = 264
4	1.4607D+03	-1.0249D-03	6.3694D+05	1.4211D-03	35 × 3 × 2 = 210

Table 3: Electronic packaging: MDF results.

Initial	Initial	Initial	Final	Final	Work
Point	Objective	Max. Con. Viol.	Objective	Max. Con. Viol.	
	(System)		(System)		
1	7.7944D+01	0.00D+00	3.51968D+05	1.00D-04 (c_1) 1.02D-08 (c_2)	110 system iter 4886 + 8899 = 13785
2	6.8300D+03	-2.89D-01	6.57163D+05	2.30D-04 (c_1) 1.30D-04 (c_2)	123 system iter 6315 + 13577 = 19872
3	1.5111D+03	-4.20D-02	6.5D+04^{F}	7.60D-03 (c_1) 6.57D-09 (c_2)	138 system iter 13414 + 12650 = 26064
4	1.4607D+03	-1.00D-03	6.5D+04^{F}	4.80D-03 (c_1) 1.11D-08 (c_2)	94 system iter 10205 + 9396 = 19701

Table 4: Electronic packaging: CO results.

Initial	Initial	Initial	Final	Final	Work
Point	Objective	Max. Con. Viol.	Objective	Max. Con. Viol.	
1	7.7944D+01	2.248D-03	6.8131D+05	6.00D-04 (c_1) 1.77D-06 (c_2)	135 × 2 = 270
2	6.8363D+03	-2.890D-01	6.5367D+05	1.10D-04 (c_1) 1.00D-04 (c_2)	4488 × 2 = 8976
3	1.5111D+03	-4.200D-02	6.7740D+05	6.00D-04 (c_1) 1.00D-04 (c_2)	2053 × 2 = 2106
4	1.4608D+03	-1.000D-03	6.7577D+05	1.70D-04 (c_1) 1.05D-05 (c_2)	3437 × 2 = 6874

Table 5: Electronic packaging: IDF results.

Summary of Results

The initial set of test problems was solved using the iSIGHT MDOL language based implementation of MDF, CO, and IDF from several starting points. The ratio of the successful runs to the number of attempted runs is summarized in Table 6. By a "successful run" we mean one that attains a critical point of the problem, to a specified degree of tolerance, within the allowable maximum number of iterations.

All runs were originally planned to be attempted from twelve starting points. Due to time limitations, all the data could not be generated to date. This explains the difference in the numbers of attempted runs among the problems and methods shown in Table 6.

During the problem formulation, we discovered that some of the problems did not have a suitable structure for the CO approach or for the IDF approach. We did not discard these problems, as they may prove useful for testing other methods.

Given the small problem sample, and the limitations mentioned earlier, we cannot make definitive statements about the methods under study. However, even this sample has demonstrated that the method properties predicted by analysis tend to hold.

Problem Method	1	2	3	4	5	6	7	8	9	10
MDF	12/12	12/12	12/12	12/12	3/3	5/12	10/10	1/6	5/10	5/6
CO	4/4	2/5	3/5	4/5	5/5	4/5	0/5	–	–	–
IDF	4/4	5/5	4/5	–	4/5	3/5	–	–	–	–

Table 6: Summary: number of successful runs / number of attempted runs.

Problem Method	1	2	3	4	5	6	7	8	9	10
MDF	610	220	610	81	3234	5024	8730	245	1574	1353
CO	15626	19872	1785	2102	837	40125	691058	–	–	–
IDF	9530	8976	382	–	544	932	–	–	–	–

Table 7: Summary: representative number of analyses for convergence.

Problem Method	1	2	3	4	5	6	7	8	9	10
MDF	667	275	1025	77	10626	3035	6887	245	1547	1353
CO	13065	18005	2983	2102*	837*	40125*	691058*	–	–	–
IDF	9640	6019	406	–	694	1071	–	–	–	–

Table 8: Summary: average number of analyses for convergence. Note: numbers marked by an asterisk are based on a single data point.

In particular, the following trends were observed with respect to convergence and the number of function evaluations.

MDF

The method converged to known solutions for the majority of cases. However, for two of the problems, the use of the fixed-point iteration to attain MDA caused very unstable behavior, and the optimizer was unable to converge from many starting points.

Given the small sample, the number of function evaluations is not conclusive. However, for the current dataset, MDF consistently requires fewer function evaluations than does CO (with the exception of one problem). MDF did better than IDF for two problems, and worse than IDF for three problems. Interestingly, IDF did worse on problems that 1 and 2, that exhibit more typical MDO structure than do problems 5 and 6, on which MDF performed poorly.

CO

The method was used to solve seven of the ten problems. The system-level compatibility constraints were actually implemented not as equalities, but as inequalities, as this alleviates some of the numerical difficulties associated with the equality constrained system-level problem ([1]). Some adjustment of the tolerances on the system-level constraints and on the lower-level convergence criteria had to be done. A general qualitative trend can be stated as follows: larger tolerances on the system-level constraints and smaller tolerances on the lower-level convergence criteria lead to better chances for attaining overall convergence.

Compared to the other two methods, CO typically required a significantly larger number of function evaluations. It was more efficient than MDF only for problem 5, and it was always less efficient than IDF.

CO also seemed to be less robust than the other methods, failing to find a critical point more frequently. On the one hand, this feature confirms the analytical properties of CO. On the other hand, this may be explained by our formulation of the problems, and by not spending a large amount of time on fine-tuning the method, which was intentional, as we mentioned.

Again, for a large test set, with a more evident MDO structure, the difference in performance may be less pronounced.

IDF

Similarly to CO, the compatibility constraints for IDF were relaxed to inequalities.

IDF performed consistently better than CO on our small sample of problems. However, one must remember that IDF is less attractive than CO with respect to the ease of integration and its handling of the disciplinary constraints.

A representative number of analyses required for each of the methods, for our test set, is summarized in Table 7, while the average number of analyses is given in Table 8.

General Comments

During the testing, we also considered the less tangible metrics, such as the ease of implementation. Here, the MDF was not typical because of the nature of our problems. It was the easiest to implement because our problems are small and relatively simple. However, in real applications, the integration of MDA can be a formidable task.

For our small test set, CO and IDF were about equally easy to implement. For large, more realistic applications, we expect that CO will be easier to implement, because of the way it handles disciplinary constraints.

The current and future method studies are proceeding on several fronts. Information is accumulated on other methods, such as the multilevel methods (MAESTRO [23], [24] [25]) and concurrent subspace optimization methods (CSSO [26], [27]); testing procedures are being fine-tuned for the methods evaluated in Phase I of the study; larger, more realistic problems are being added to the problem test set.

In conclusion, while the results are by no means complete, we have found this systematic study of methods very informative. The tests in general confirmed the expected numerical properties; however, a significant amount of further testing would be necessary to explain the numerical behavior of methods definitively. We invite other researchers to contribute to accumulating systematic data, that will eventually lead to a set of practical guidelines for the use of MDO methods (please see the method evaluation web site accessible from the MDOB Homepage).

Acknowledgments

The authors would like to thank R. Haftka for his suggestions.

References

[1] N.M. Alexandrov and R.M. Lewis. Analytical and computational aspects of collaborative optimization. *Engineering Optimization*, 1998. Submitted.

[2] G.R. Shubin. Application of alternative multidisciplinary optimization formulations to a model problem for static aeroelasticity. Technical Report BCSTECH-93-022, Boeing Computer Services, BCS Technology, December 1993.

[3] D. Haim, A.A. Giunta, M.M. Holzwarth, W.H. Mason, and L.T. Watson. Comparison of optimization software packages for an aircraft multidisciplinary design optimization problem. *Design Optimization*, 1998. Submitted.

[4] S. Kodiyalam. Evaluation of methods for multidisciplinary design optimization, Phase I. Technical report, NASA Contractor Report, 1998.

[5] E.J. Cramer, J. E. Dennis, Jr., P.D. Frank, R.M. Lewis, and G.R. Shubin. Problem formulation for multidisciplinary design optimization. *SIAM Journal on Optimization*, 4(4):754–776, November 1994.

[6] R.D. Braun. *Collaborative Optimization: An architecture for large-scale distributed design.* PhD thesis, Stanford University, May 1996. Department of Aeronautics and Astronautics.

[7] D. A. Wismer, editor. *Optimization Methods for Large-Scale Systems with Applications.* McGraw-Hill, New York, 1971.

[8] L.S. Lasdon. *Optimization Theory for Large Systems.* Macmillan, 1970.

[9] R.D. Braun, P. Gage, I. Kroo, and I. Sobieski. Implementation and performance issues in collaborative optimization. In *Proceedings of the Sixth AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Bellevue, WA*, Sept. 1996.

[10] H.M. Adelman, J.L. Walsh, and J.I. Pritchard. Recent advances in integrated multidisciplinary optimization of rotorcraft, AIAA paper AIAA-92-4777-CP. In *Proceedings of the Fourth*

AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization, pages 710–721, September 1992.

[11] J.L. Walsh, K.C. Young, J.I. Pritchard, H.M. Adelman, and W.R. Mantay. Integrated aerodynamic/dynamic/structural optimization of helicopter rotor blades using multilevel decomposition. Technical Report NASA TP 3465, NASA Langley Research Center, January 1995.

[12] J.L. Walsh, K.C. Young, J.I. Pritchard, H.M. Adelman, and W.R. Mantay. Multilevel decomposition approach to integrated aerodynamic/dynamic/structural optimization of helicopter rotor blades. Technical Report NASA TM 109084, NASA Langley Research Center, May 1994.

[13] R.M. Lewis. Practical aspects of variable reduction formulations and reduced basis algorithms in multidisciplinary design optimization. In N.M. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization: State-of-the-Art*, Philadelphia, 1997. SIAM.

[14] S. Kodiyalam, L. Swenson, and B. Stehlin. Multidisciplinary design optimization with object-oriented product modeling. In *Proceedings of the Conference on Optimization in Industry, organized by Engineering Foundation, NSF, and NASA Langley*, Florida, 1997.

[15] I.P. Sobieski and I.M. Kroo. Collaborative optimization using response surface estimation. In *Proceedings of the 36th Aerospace Sciences Meeting and Exhibit, Reno, NV*, January 1998. AIAA Paper AIAA-98-0915.

[16] S.L. Padula, N.M. Alexandrov, and L.L. Green. MDO Test Suite at NASA Langley Research Center. In *Proceedings of the Sixth AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Bellevue, WA*. AIAA, 1996.

[17] J.E. Renaud. An optimization strategy for multidisciplinary systems design. In *Proceedings of the International Conference on Engineering Design*, August 1993.

[18] G. Kott, G.A. Gabriele, and J. Korngold. Application of multidisciplinary design optimization to the power stage design of a power converter. *ASME Advances in Desing Automation*, 2, 1993.

[19] W.-C. Li. *Monotonicity and Sensitivity Analysis in Multilevel Decomposition-Based Design Optimization.* PhD thesis, University of Mariland, 1989.

[20] B. Averick, R.G. Carter, J.J. Moré, and G-L. Xue. The MINPACK-2 test problem collection. Technical Report ANL MCS-TM-150, Argonne National Laboratory, Argonne, Illinois, 1992.

[21] J.E. Dennis, D.M. Gay, and P.A. Vu. A new nonlinear equations test problem. Technical Report 83-16, Rice University, Houston, Texas, 1983.

[22] C.A. Floudas and P.M. Pardalos. *A Collection of Test Problems for Constrained Global Optimization.* Lecture Notes in Computer Science, Volume 455. Springer–Verlag, Berlin; New York, 1991.

[23] N.M. Alexandrov. *Multilevel algorithms for nonlinear equations and equality constrained optimization.* PhD thesis, Rice University, Department of Computational and Applied Mathematics, May 1993. Also available as Technical Report TR93–20.

[24] N.M. Alexandrov and J. E Dennis, Jr. Multilevel algorithms for nonlinear optimization. In J. Borggaard, J. Burkardt, M. Gunzburger, and J. Peterson, editors, *Optimal Design and Control*, pages 1–22. Birkhäuser, 1994.

[25] N.M. Alexandrov. Multilevel methods for MDO. In N.M. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization: State of the Art*, pages 79–89. Society for Industrial and Applied Mathematics, Philadelphia, 1997.

[26] J. Sobieszczanski-Sobieski. Optimization by decomposition: A step from hierarchic to non-hierarchic systems. Technical Report TM 101494, NASA, Hampton, Virginia, September 1988.

[27] J.E. Renaud. A concurrent engineering approach for multidisciplinary design in a distributed computing environment. In N.M. Alexandrov and M.Y. Hussaini, editors, *Multidisciplinary Design Optimization: State of the Art.* SIAM, 1997.