Deterministic Design Optimization of Structures in OpenMDAO Framework

Rula M. Coroneos[1] and Shantaram S. Pai[2]
National Aeronautics and Space Administration, Glenn Research Center, Cleveland, Ohio 44135

Nonlinear programming algorithms play an important role in structural design optimization. Several such algorithms have been implemented in OpenMDAO framework developed at NASA Glenn Research Center (GRC). OpenMDAO is an open source engineering analysis framework, written in Python, for analyzing and solving Multi-Disciplinary Analysis and Optimization (MDAO) problems. It provides a number of solvers and optimizers, referred to as components and drivers, which users can leverage to build new tools and processes quickly and efficiently. Users may download, use, modify, and distribute the OpenMDAO software at no cost. This paper summarizes the process involved in analyzing and optimizing structural components by utilizing the framework's structural solvers and several gradient based optimizers along with a multi-objective genetic algorithm. For comparison purposes, the same structural components were analyzed and optimized using CometBoards, a NASA GRC developed code. The reliability and efficiency of the OpenMDAO framework was compared and reported in this report.

Nomenclature

A	=	cross sectional area
α	=	step length
d	=	direction vector
$f(x)$	=	objective function
g	=	constraint
ℓ	=	length
LB	=	lower bound
n	=	number of design variables
ρ	=	material density
UB	=	upper bound
W	=	weight
x	=	design variable

I. Introduction

A structural design problem can be represented as a mathematical model whose constituent elements are design parameters, constraints and an objective(s) or merit function(s). The design parameters specify the geometry and topology of the structure and physical properties of its members. Some of these can be independent design parameters and others could be dependent on the independent design variables. Design parameters are chosen by judgment and experience of the engineer so as to reduce the size of the problem. This results in large savings in computational time, which in turn reduces the cost of the design phase. From the design parameters, a set of derived parameters are obtained which are defined as behavior constraints e.g., stresses, displacements, natural frequencies etc. These behavior constraints are functionally related through laws of structural mechanics to the design variables. The objective or the merit function is formulated based on-real-world performance goals for the structure and is ultimately a function of the design parameters. This function is minimized if it is weight or cost but can be maximized if it is performance or a combination of these functions.

The process of design uses the results of analysis to make decisions about the problem description. The analysis may return stress, strain, reactions, shear and moment values, and comparison with allowable performance relative to specified constraints (such as strength, deflection, etc.). Based on the results of the analysis, the design process changes the problem description (design variables) and then additional analysis is performed in an iterative process and this design process repeats until the best solution is found.

[1] Computer Scientist, Structures and Dynamics Branch, 21000 Brookpark Rd./MS 49-8, AIAA Senior Member
[2] Aerospace Engineer, Structures and Dynamics Branch, 21000 Brookpark Rd./MS 49-8, AIAA Senior Member

The basic requirement for an efficient structural design is that the response of the structure should be a feasible design. There can be a large number of feasible designs, but it is desirable to choose the best or optimum from these several designs. The best design could be in terms of minimum weight, minimum cost, or maximum performance, or a combination of these. The optimization problem is classified as linear or nonlinear based on the nature of equations with respect to the design variables. If the objective function and the behavior constraints involving the design variables are linear then the optimization is termed as linear optimization problem. If even one of them is nonlinear, it is classified as the nonlinear optimization problem. The constraints for structural design applications are typically nonlinear, thus it becomes a nonlinear programming problem. In general, the design variables are real but sometimes they could be integers for example, the number of layers in plies, orientation angle, etc., when composite materials are used. The behavior constraints could be equality constraints or inequality constraints depending on the nature of the problem.

The subject matter of this report is presented in the subsequent five sections. In Section 2, the formulation of the structural optimization problem is introduced. In Section 3, an overview of the OpenMDAO framework is presented. In Section 4, several numerical examples are provided and conclusions are given in Section 5.

II. Structural Optimization Problem Formulation

The structural optimization problem can be cast as a nonlinear mathematical programming problem as: Find the n design variables x within prescribed lower and upper bounds $(x_i^{LB} \le x_i \le x_i^{UB}, i = 1,2,...,n)$ such that the scalar objective function $f(x)$ is minimized.

where x represents the independent active design variables for all groups of shell and beam elements.

The weight of the structural components has a nonlinear form because of the nature of the design variable formulation. The overall weight W, for a structure composed of truss and beam elements in symbolic form can be expressed as:

$$W = \sum_{j=1}^{m} \rho_j \ell_j A_j \tag{1}$$

where ρ_j is the material density for each member, ℓ_j is the member length, A_j is the cross-sectional area and m is the number of members. Alternatively, the overall weight of the structure composed of shell elements is:

$$W = \sum_{j=1}^{m} \rho_j A_j t_j \tag{2}$$

The constraints g, are imposed on stress, displacements and frequency responses. In general, the stress and displacement constraints are formulated as follows:

$$g = |\text{value}| \le \text{allowable} \tag{3}$$

or equivalently:

$$g = \left| \frac{\text{value}}{\text{allowable}} \right| - 1 \le 0 \tag{4}$$

The natural frequency formulation is given as:

$$g_{\text{freq}} = \left(\frac{\text{freq\_allowable}}{\text{freq\_value}} \right)^2 - 1 \le 0 \tag{5}$$

The general formula to update the design variables, (A_j) in a nonlinear programming algorithm at the k^{th} intermediate iteration is given as:

$$\{A\}_k = \{A\}_{k-1} + \alpha_{k-1}\{d\}_{k-1} \tag{6}$$

where the step length α_{k-1} is calculated to find the local minimum of the objective or weight in this study along the direction $\{d\}_{k-1}$ in the feasible domain.

Over the years, a large number of techniques have been suggested to solve these equations resulting in an optimal design. However, these techniques do not always lead to a global optimum. These at best lead to local optimum. If the constraint equations and the objective function are convex functions, then it is possible to conclude that the local optimum will be a global optimum. However, in most of the structural design problems it is practically impossible to check the convexity of the function. One of the simplest ways is to start with different feasible solutions and check the solutions for global optimality. In other words, if the solutions with same objective value are always found starting from different initial solutions, the solutions can be conceived as globally optimal.

III. OpenMDAO Framework Overview

OpenMDAO is an open source framework for analyzing and solving Multidisciplinary Analysis and Optimization (MDAO) problems (Ref. 1). The framework is hosted at the site: (http://openmdao.org/). In OpenMDAO, a problem is represented by a system of objects called components. A conceptual view of a simple component is shown in Figure 1.

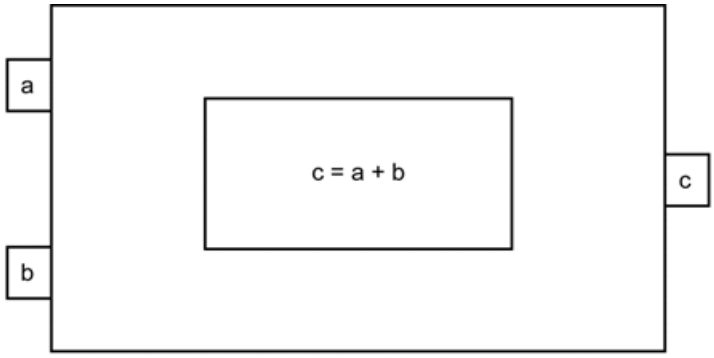

Figure 1.—Conceptual view of a component (a, b are inputs, c is output).

Components within OpenMDAO can be as simple or complex as necessary. The inputs (a and b) and outputs (c) to a component are Python objects, so they are not limited to being simple types like floating point or integer. A component in OpenMDAO can be thought of as a black box that transforms inputs to obtain outputs, but it also is the main building block for putting together analyses, that can represent entire disciplines such as structural solutions with MSC/Nastran (Ref. 2).

A *Workflow* in OpenMDAO framework is an object that executes a group of components in a particular order, as shown in Figure 2.

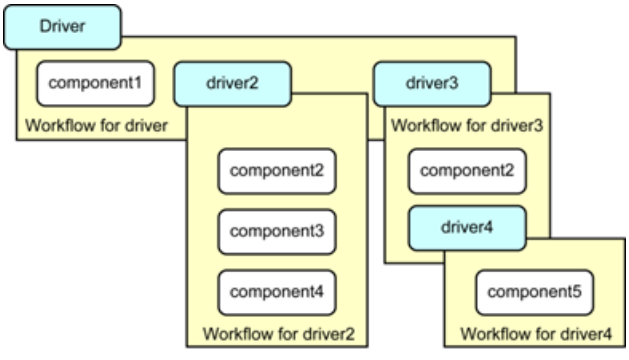

Figure 2.—View of an iteration hierarchy.

A *Driver* within OpenMDAO is a special kind of Component that executes a Workflow repeatedly until some condition is met, shown in Figure 2. Some examples of Drivers are optimizers, solvers, and design space explorers. Gradient based drivers such as NEWSUMT or SUMT (Ref. 3), ConMin (Ref. 4), NLPQ (Ref. 5), IpOPT (Ref. 6) were used for this study. The famous multi-objective evolutionary optimizer NSGA-II (Non-Dominated Sorting Genetic Algorithm) (Ref. 7) has also been wrapped in OpenMDAO and tested in this study.

An *Assembly* in the framework is a special kind of Component that contains other components. One of those components must be a Driver named *driver*. When an Assembly executes, it executes *driver*, which then executes its Workflow. A Driver's Workflow may contain other Drivers, and each of those Drivers has a Workflow of its own.

The hierarchical structure defined by the contents of an Assembly's drivers and the contents of their workflows is called an *iteration hierarchy*.

Figure 2 shows an example of an iteration hierarchy involving four different Drivers. Note that in this example the same component, *component2*, appears in two different workflows, and is executed in both of them.

The data flow within an Assembly having one Driver and four Components, is shown in Figure 3.

A solid line between two Components indicates that one of them is supplying inputs to the other. Each dashed line between a Driver and a Component indicates a parameter, objective, or constraint in the Driver that references an input or output variable in the Component. The arrow at the end of a dashed or solid line indicates the direction of the data flow between two connected objects. OpenMDAO handles all data passing between components, and also has the ability to check and convert units on data connections. The functionality of OpenMDAO can be extended through the use of plugins.

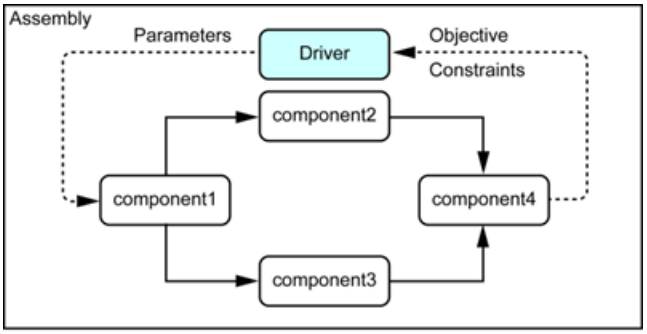

Figure 3.—View of an assembly showing data flow.

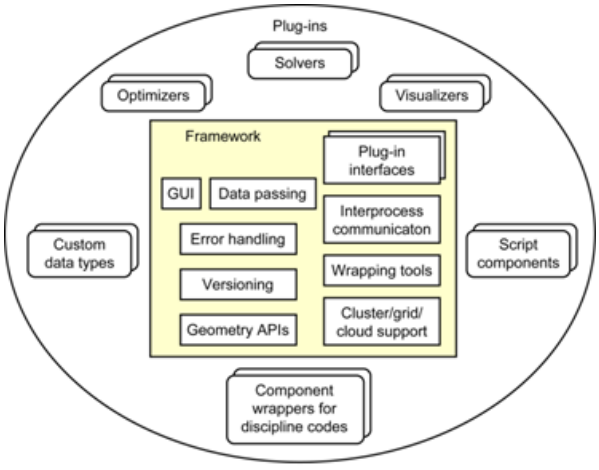

Figure 4.—Top level context diagram.

In Figure 4, objects of the sort found outside of the Framework box can be integrated into the framework as plugins. This means that a user can create any of these and the framework will understand how to interact with them. This is possible because plugins have a specific interface and are packaged in a way that the framework expects. In particular, the plugin system allows the distribution of wrappers to nondistributable or closed-source tools that the user has. To learn how to create your own plugins, see the Plugin Developer Guide (Ref. 1).

IV. Structural Test Cases in OpenMDAO

Many test cases have been implemented in OpenMDAO framework from different engineering disciplines, which include a few academic problems. For structural analysis, several test cases have been implemented in OpenMDAO, including some well-studied problems from the literature, (Refs. 8 and 9). Solutions to these problems are available for several optimization approaches. Results of each of these problems have been compared with CometBoards and found that OpenMDAO outputs almost identical values. These problems were formulated in OpenMDAO to test the Python wrapper codes, validate, and verify the OpenMDAO overall framework implementation including the performance of the components and drivers. For testing and validation of the framework the structural test cases in this study were also formulated and executed in CometBoards (Ref. 8).

For structural analysis, the MSC/Nastran (Ref. 2) commercial code has been wrapped in the OpenMDAO framework and used for the analysis of the structural test cases. A closed form solution of the three bar truss problem was also implemented in FORTRAN and wrapped using the F2PY (Fortran to Python) interface generator to provide early validation. For the single objective optimization, NEWSUMT or SUMT for short (Ref. 3), ConMin (Ref. 4), NLPQ (Ref. 5) and IpOPT (Ref. 6) optimizers were used and imported from the standard library that comes with OpenMDAO. However, NLPQ and IpOPT are plugins that are installed separately. NLPQ is a commercial product that has been approved for use at NASA Glenn Research Center.

To further test and validate the multi-objective optimization capabilities of the OpenMDAO framework, the NSGA-II (Ref. 7) algorithm was used. This driver can be imported from the openmdao drivers api library as "from openmdao.lib.drivers.api import NSGAdriver".

Seven structural test cases are presented in this report:

1. A three-bar truss design for single and multi-objective optimization
2. A ten-bar truss design for single and multi-objective optimization
3. A twenty-five bar truss antenna tower
4. A sixty-bar trussed ring
5. A geodesic dome design
6. A composite plate and
7. A NASA ceramic matrix composite blade

For the single objective optimization, minimum weight was the objective function and thickness or areas of the members were the design variables. A grouping strategy was also followed to reduce the number of design variables, depending on the problem size. Stresses and displacements were considered as the behavior constraints. For the multi-objective 3-bar truss test case the weight along with the volume of the structure were minimized. Multiple static load conditions and behavior limitations were specified to ensure that several types of behavior constraints were active at the optimum. Each problem had a specified initial design and a set of upper and lower bounds. Typically, default optimization parameters and convergence criteria specified in the individual codes were used. These parameters, however, were changed when convergence difficulty was encountered. These problems were executed on a Linux x86_64 machine at 2.67 GHz. OpenMDAO results including the optimum weight, number of design variables, number of active constraints, iteration number and optimum design along CPU time history from each of the four optimizers are given in the following tables and figures. For comparison and validation purposes, results from CometBoards using the NEWSUMT optimizer denoted in this report as (CB_SUMT) is also included.

A. Three-Bar Truss Design

The popular 3-bar truss (Refs. 8 and 9), as shown in Figure 5(a) was subjected to two loading cases. Node 1 is free to move in the x and y directions and nodes 2, 3, and 4 are fixed. The truss is made of steel with Young's modulus of 30×10^6 psi and density of 0.289 lb/in.^{3}. The design variables were the cross-sectional areas of the bars, with an initial design of 1.0 in.^{2} and a lower bound of 0.01 in.^{2}. Behavior constraints were imposed on stress and displacements. Initially, the truss was optimized as a single objective optimization problem using gradient optimizers where the objective function was to minimize the weight. Optimization results along with comparison from CometBoards are depicted in Tables 5 and 6. All four optimizers converged to the same optimum design. A graphical representation of number of iterations versus weight is shown in Figure 5(b). CPU time in minutes is plotted in Figure 15. It is interesting to note that the CPU time required by SUMT in OpenMDAO was about 81 percent less than that of CometBoards (CB_SUMT).

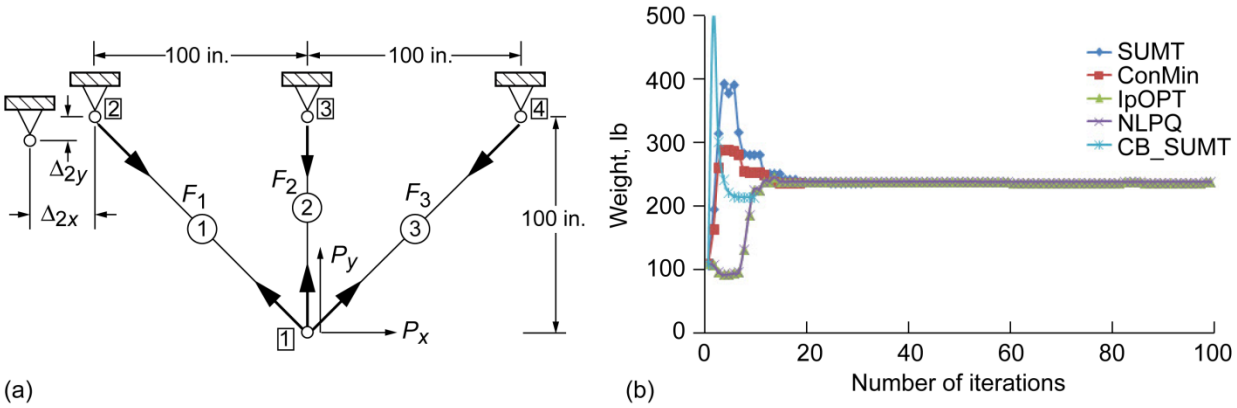

Figure 5.—(a) Three-bar truss and (b) convergence history.

The 3-bar truss was also analyzed as a multi-objective problem where the weight and enclosed volume of the truss were the two objective functions. The NSGA-II algorithm was used as the optimizer. For this type of shape optimization, the design variables were the three cross-sectional areas of the bars, and position of node 1 in the y direction, along with position of nodes 2 and 4 in the x direction. Node 3 was fixed since it carries the load of the structure. Behavior constraints were imposed on stress and displacements. The allowable strength for all members is 20 kips per square inch (ksi) for both tension and compression. The cross-sectional areas were initialized to 1.0 in.^{2}.

The structural analysis is performed using the MSC/Nastran Solution 101. The optimization parameters in the NSGA-II algorithm were passed as: population size was set to 80; generations to 50; and max solutions to 4000. The crossover and mutation probabilities were set to 1.0 and 0.5, respectively. The distribution index for crossover was 20 and for mutation 50. The optimum weight computed from NSGA-II is 179.38 lb with a volume of 24,165 in.^{3}, shown in Figure 6 and the new design compared with the initial design is depicted in Figure 7. NSGA-II converged to a Pareto Front that is entirely lighter than the weights given with single objective optimization due to the different settings of the nodal restraints.

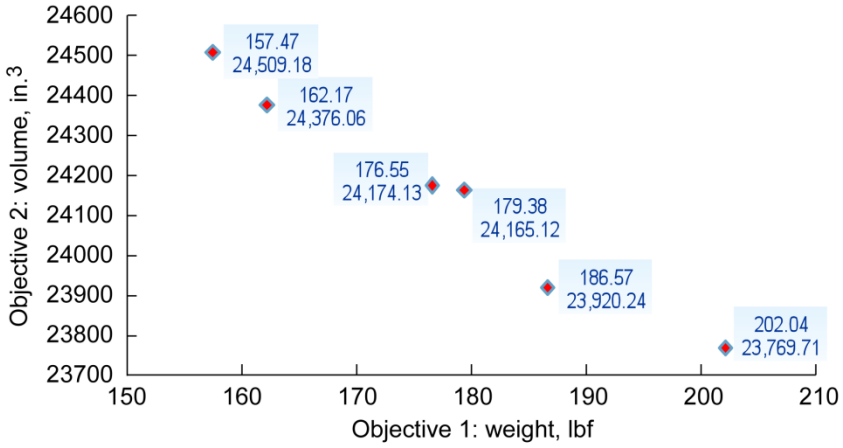

Figure 6.—Pareto optimal frontier for the three-bar truss using NSGA-II.

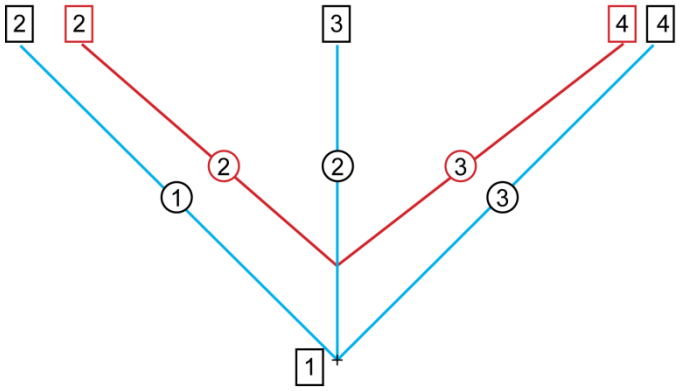

Figure 7.—Initial (blue) and final (red) designs for the three-bar truss using NSGA-II (nodes in squares, elements in circles).

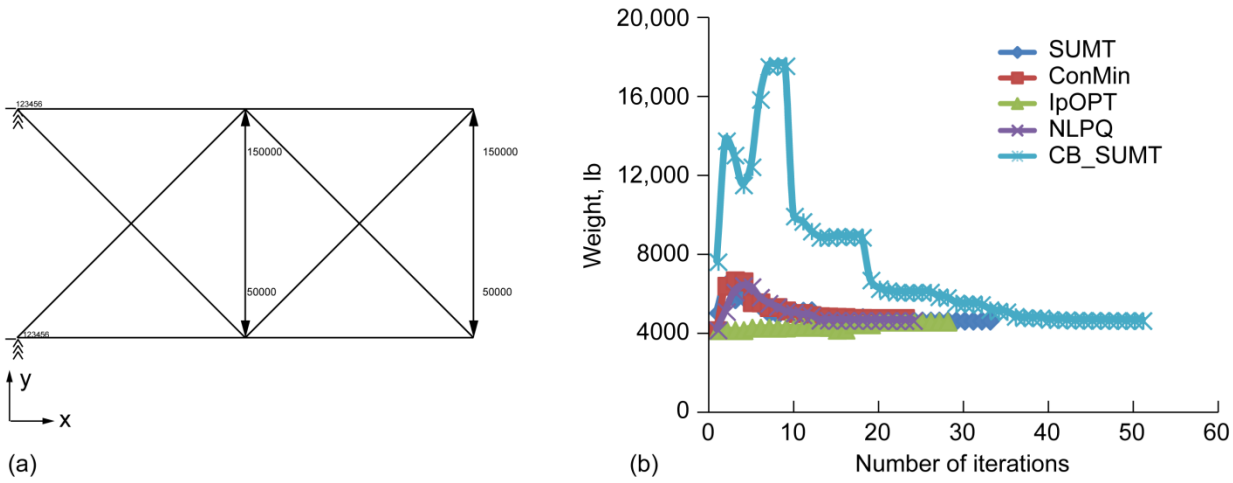

Figure 8.—(a) Ten-bar truss and (b) convergence history.

B. Ten-Bar Truss Design

The 10-bar truss is another well-known cantilever truss illustrated in Figure 8(a). Each member's area is treated as an independent design variable giving a total of 10 for this problem. The truss is made of aluminum with Young's modulus of 10×10^6 psi. The structure is restrained at nodes 1 and 10. All other nodes are allowed to move horizontally and vertically. The structural responses consist of stresses and displacements. The constraint is that each member's stress may not exceed 25×10^3 psi for both tension and compression and the displacement at nodes 3 and 4 may not exceed 2 in. in the y-direction. The cross sectional areas are the design variables within the range of ($0.1 < 10.0 < 100.0$) in.^{2}. Optimization results for the 10-bar truss are summarized in Tables 5 and 6. A graphical representation of the weight convergence history is shown in Figure 8(b). All methods converged to about the same optimum weight with minor deviations. CPU time for all methods is depicted in Figure 15 and ranges between 19 min for ConMin to 92 min for SUMT in OpenMDAO. The reduction in CPU time for SUMT in OpenMDAO compared with CometBoards was about 85 percent.

C. Twenty-Five Bar Truss (Power Transmission Tower)

The tower, shown in Figure 9(a), represents a structure that carries transmission lines, consististing of 25 axial-force members, and is made of aluminum. The 25 bar members are grouped into 8 design variables, see Table 1. The structure is required to be double symmetric about the x and y axes, in spite of the directional nature of these loads. The minimum displacements of ±0.35 in. are at the upper nodes 1 and 2 and the members are designed for a 25,000 lb/in.^{2} tensile and compressive stress. The range of the members size is ($0.01 < 1.0 < 10.0$) in.^{2}. Comparison of total weight, constraint activity and optimum designs by the five methods are presented in Tables 5 and 6 along with the convergence history in Figure 9(b).

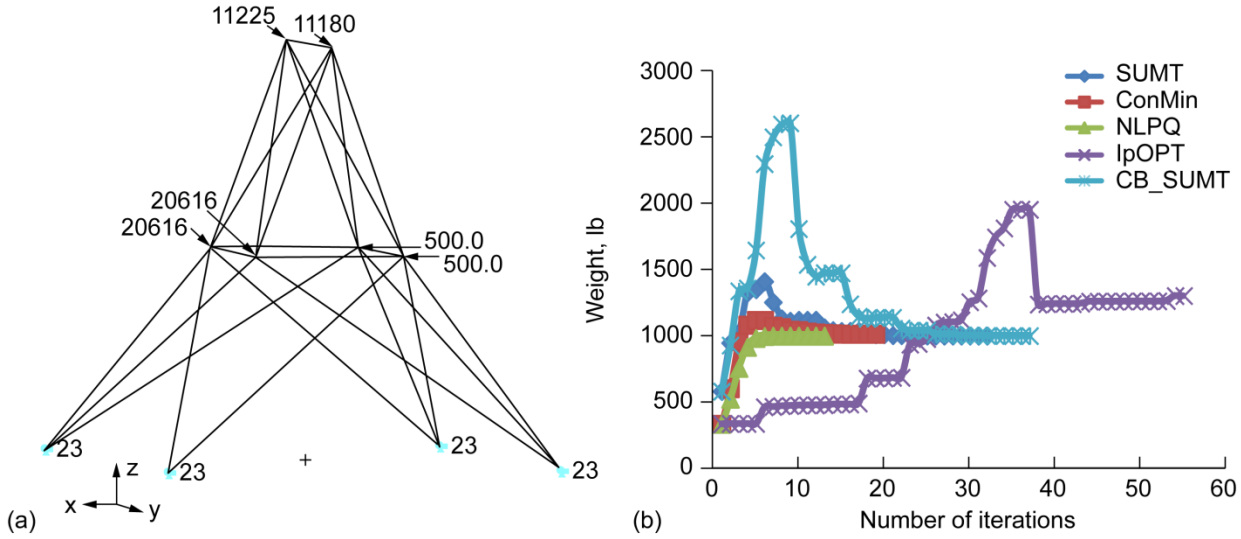

Figure 9.—(a) Twenty-five-bar truss and (b) convergence history.

The CPU time in OpenMDAO ranges from 12 min for ConMin and 134 for IpOPT, seeFigure 15. Optimum weight for IpOPT differed by 30.35 percent compared with SUMT and NLPQ with no active constraints. All other optimizers in OpenMDAO produced 5 active stress constraints, see Table 5.

TABLE 1.—TWENTY-BAR TRUSS DESIGN VARIABLE GROUPING

Problem	Design variable	Members grouped
25-bar antenna tower (8LDV)	1	1
	2	2,3,4,5
	3	6,7,8,9
	4	10,11
	5	12,13
	6	14,15,16,17
	7	18,19,20,21
	8	22,23,24,25

D. Sixty-Bar Trussed Ring

The 60-bar trussed ring is shown in Figure 10. Its inner and outer radii are 90 and 100 in. and is made of aluminum withYoung's modulus E = 10×10^6 psi and density = 0.1 lb/in.^{3}. It is subjected to two load conditions, denoted in the figure as LC1 and LC2. The 60 bar areas of the structure were grouped into 25 design variables to obtain a reduced set of design variables as shown in Table 2. The problem is solved for both stress and displacement constraints. The strength allowable for both tension and compression is 10×10^3 psi. Displacement limitations were imposed along both x and y directions at nodes 10, 4, 19, and 13 with limiting values of (1.25, 1.75, 2.75, and 2.25 in.), respectively. The initial area for all bars was set to 1.0 in.^{2}. Convergence iterations for the five methods is plotted in Figure 11. Results from all optimization methods are given in Tables 5 and 6, along with CPU times for each optimizer in Figure 15.

The CPU time varied from 12 min for ConMin to 133 min for IpOPT. However, ConMin produced only one active stress constraint compared to 12 active stresses for SUMT and NLPQ.

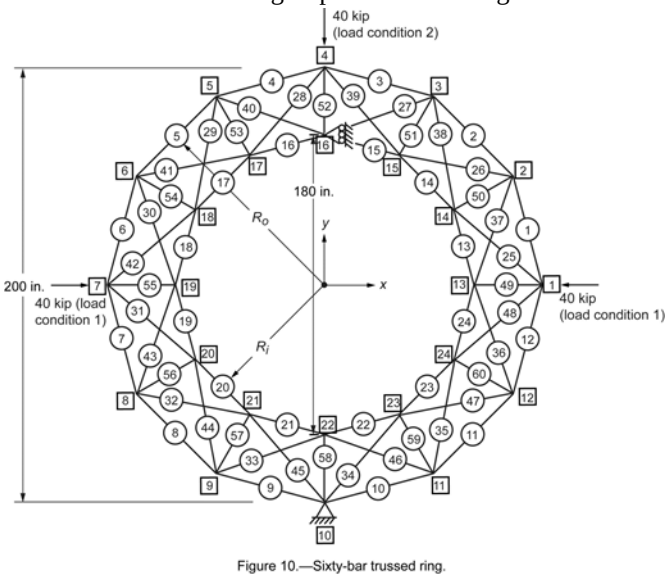

Figure 10.—Sixty-bar trussed ring.

TABLE 2.—DESIGN VARIABLE GROUPING FOR THE 60-BAR TRUSSED RING

Problem	Design variable	Members grouped
60-bar trussed ring (25LDV)	1	49,50,51,52,53,54,55,56,57,58,59,60
	2	1,13
	3	2,14
	4	3,15
	5	4,16
	6	5,17
	7	6,18
	8	7,19
	9	8,20
	10	9,21
	11	10,22
	12	11,23
	13	12,24
	14	25,37
	15	26,38
	16	27,39
	17	28,40
	18	29,41
	19	30,42
	20	31,43
	21	32,44
	22	33,45
	23	34,46
	24	35,47
	25	36,48

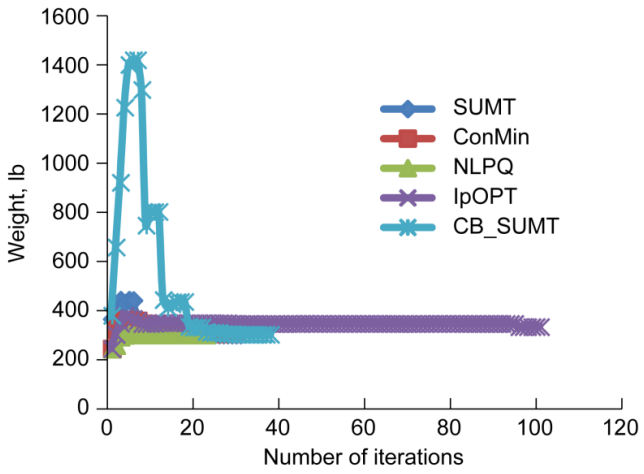

Figure 11.—Sixty-bar trussed ring iteration history.

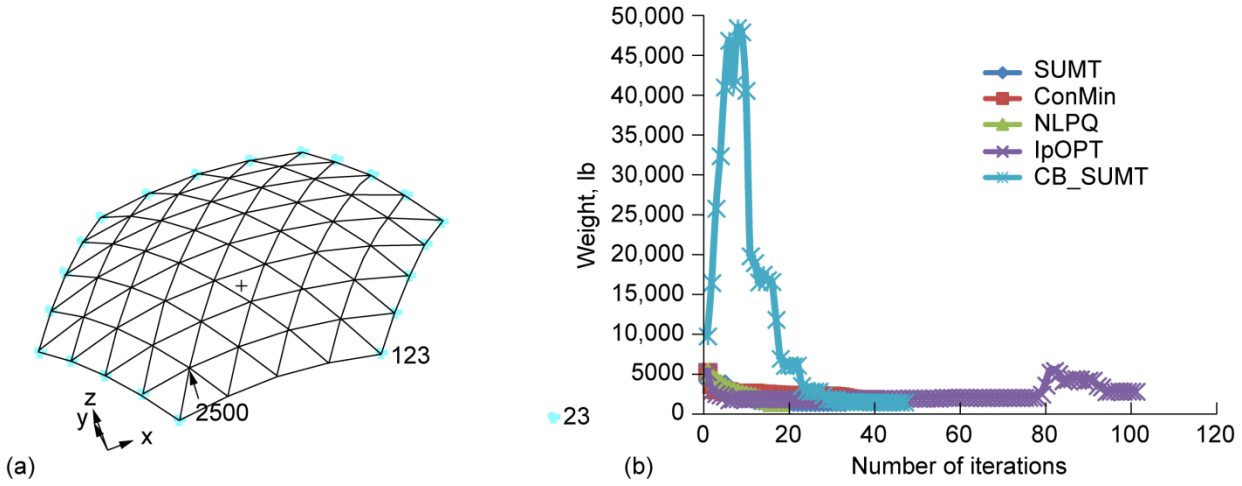

Figure 12.—(a) Geodesic dome and (b) convergence history.

E. Geodesic Dome

A geodesic dome, shown in Figure 12(a), with a diameter of 240 in. and a height of 30 in. was subjected to a distributed load of 925 kip. It was modeled using 156 bars and 96 triangular membrane elements. Material data for the bars is: $E = 30\times10^6$ psi with density $\rho = 0.289$ lb/in.^{3} and Poisson's ratio $\nu = 0.3$. Triangular membranes were made of aluminum with modulus $E = 10\times10^3$ psi, density $\rho = 0.1$ lb/in.^{3} and Poisson's ratio $\nu = 0.3$. The stress allowable for the bars is $\sigma_o = \pm25\times10^3$ psi, for the membranes is $\sigma_o = \pm10\times10^3$ psi and the displacement limitation was specified at 0.5 in. The areas of the bars were grouped to obtain eight linked design variables and the triangular membranes were grouped to obtain four linked design variables, each of which are shown in Table 3. The dome had a total of 253 constraints (156 axial stresses for bars, 96 Von Mises stress for membranes and one displacement constraint). The optimum weight obtained was 1539.597 lb with 60 active stress constraints, see Tables 5 and 6. Weight convergence history is depicted in Figure 12(b). All optimizers converged with small deviations. CPU time in OpenMDAO varied between 26 min for NLPQ and 390 min for SUMT, see also Figure 15.

TABLE 3.—DESIGN VARIABLE GROUPING FOR GEODESIC DOME

Problem	Design variable	Members grouped
Geodesic dome (12LDV)	1	1-6
	2	7-12
	3	13-30
	4	31-42
	5	43-72
	6	73-90
	7	91-132
	8	133-156
	9	157-162
	10	163-180
	11	181-210
	12	211-252

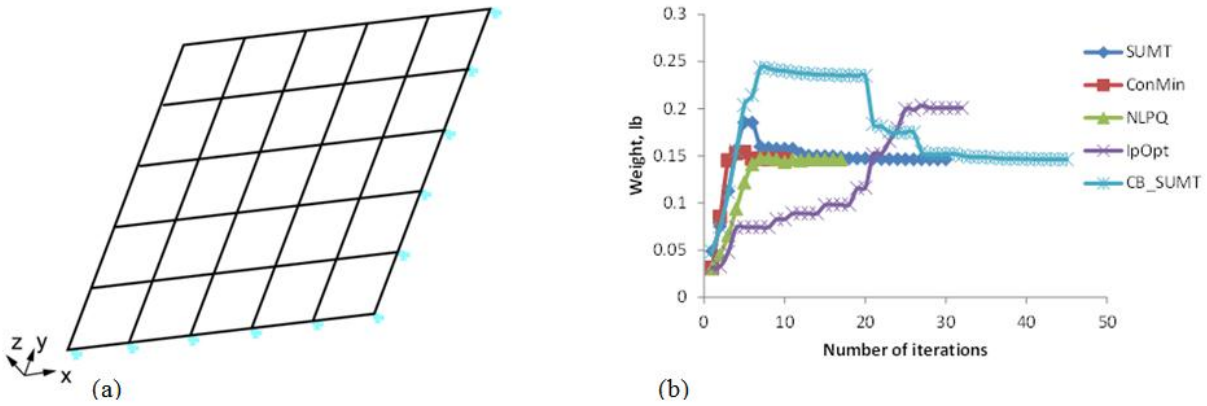

Figure 13. – (a) Composite plate and (b) convergence history

TABLE 4.—COMPOSITE PLATE DESIGN VARIABLE GROUPING

Problem	Design variable	Members grouped
Composite plate (3LDV)	1	2,3,4,6,10,11,15,16,20,22,23,24
	2	7,8,9,12,13,14,17,18,19
	3	1,5,21,25

F. Composite Plate

A 1×1 in. composite plate, shown in Figure 13(a) is loaded with a total of 60000 lb/in. in the z-direction on the right side edge. The left side reacts the loads with X,Y,Z and all rotations, and in addition rotations in R_z constraints at all nodes. The three-composite 4 ply lay-up is made of graphite/epoxy tape with total initial thickness of (0.5, 0.6, 0.7 in.), respectively. The same elastic and strength properties were applied for each composite ply lay-up.

The composite laminate contained four plies as: [0/–45/45/0] or1 ply in the 0° direction, the second ply lay-up contained 1 ply in the –45° direction, the third ply contained one ply lay-up in the 45° direction and the last ply contained 1 ply in the 0° direction. Maximum Strain failure theory as implemented in the MSC/Nastran composite property card, PCOMP, was considered for the analysis. The 25 shell element thicknesses were grouped to obtain a reduced set of three design variables, one group for each of the composite properties, as shown in Table 4. Constraints were specified on maximum strains not to exceed 4×10^{-3} micro strain, and displacement limitation was in the z-direction at node 18 of 0.0415 in. The optimum weight obtained from four optimizers was 0.146 lb with 3 active strain constraints and one displacement. The weight from IpOPT was 0.201 lb with no active constraints, see Tables 5 and 6. The iteration history of all optimizers is plotted in Figure 13(b). In OpenMDAO, the CPU time varied between 3 min for ConMin and 52 min for SUMT, see also Figure 15.

G. Ceramic Matrix Composite (CMC) Blade Design

The turbine engine blade design, shown in Figure 14 was analyzed and optimized for weight using higher fidelity analyses and optimization. The finite element model of the hollow blade with a solid cap was developed using MSC/Patran (Ref. 11), made with a proprietary Ceramic Matrix Composite (CMC) material. The material axis is oriented as follows: high elastic modulus is along the length of the blade or radial direction, intermediate modulus is along the width or blade chord, while the smallest modulus is oriented through the thickness of the blade. The 25945 CQUAD4 shell elements were grouped into two sets of design variables. The first set of design variables consist of all the cap elements which are the red elements in Figure 14 with their element numbers ranging from (25601 to 25945). The second group consists of elements starting from (1 to 25600) for the wall of the blade and are shown in blue in Figure 14. The initial total ply thickness for the first group (cap) is 0.5 in. and for the second group (wall) is 0.03 in. The volume of the blade is 4.459 $in.^3$ with an optimum weight of 0.038 lb, with four of the optimizers, however no active constraints were produced. The optimum weight from NLPQ was 0.049 lb with the stress constraint being active, see Tables 5 and 6. The CPU time varied between 655 min for SUMT and 41 for NLPQ.

Figure 14.—Composite turbine hollow blade with solid cap.

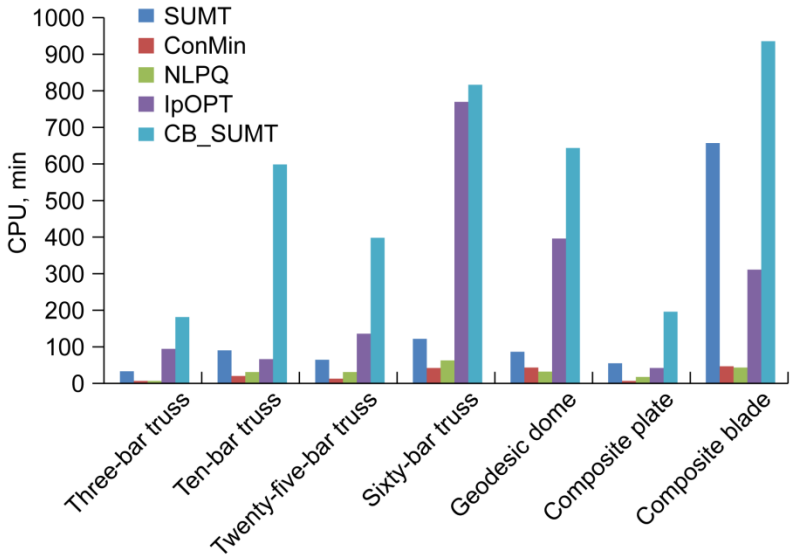

Figure 15.—CPU time for different optimization methods for all problems.

TABLE 5.—SUMMARY OF RESULTS OF SINGLE OBJECTIVE OPTIMIZATION FOR SEVEN TEST CASES

Problem, Number of design variables (DV[a])	Constraints specified	Weight in lb, Active Constraints, Iterations, CPU time (mins)				
		SUMT	ConMin	NLPQ	IpOPT	CometBoards (SUMT)
3-bar truss (3DV)	3S[a], 2D[a]	237.115 1S, 1D 33 33.191	237.151 1S, 1D 17 6.183	237.101 1S, 1D 9 4.231	237.357 1S,1D 101 93.320	237.194 1S,1D 31 180.0
10-bar truss (10DV)	10S, 2D	4677.478 2S, 1D 33 92.357	4806.917 1S,1D 21 19.055	4673.891 2S,1D 24 27.0	4620.878 Inf. 32 67.442	4678.363 2S, 1D 52 600.0
25-bar antenna tower (8LDV)	8S, 2D	998.194 5S 32 62.718	1011.804 5S 17 12.337	998.084 5S 13 26.9	1301.144 Inf. 58 133.731	998.482 6S 37 397.0
60-bar trussed ring (25LDV[a])	25S, 24D	308.621 12S,1D 32 123.282	312.748 1S,1D 28 43.930	308.553 12S,1D 18 59.0	340.0244 1S,1D 101 764.673	308.673 12S,1D 38 810.0
Geodesic dome (12LDV)	252S, 1D	1539.597 120S 33 79.753	1929.653 Inf. 38 39.315	1539.517 119S 17 26.0	2229.409 Inf. 111 390.488	1540.02 120S 48 643.0
Composite plate (3LDV)	3Strain, 1D	0.146 3Strain,1D 30 51.49	0.146 3Strain,1D 8 2.74	0.146 3Strain,1D 16 15.0	0.201 Inf. 36 35.72	0.146 3Strain,1D 45 193.0
Composite blade (2LDV)	2S, 1D	0.038 Inf. 31 654.949	0.038 Inf. 3 43.466	0.049 1S 3 40.938	0.038 Inf. 18 304.624	0.038 Inf. 31 935.0

[a](DV: Design variables; LDV: Linked design variables; S: stress; D: displacement, Inf: infeasible design)

TABLE 6.—CALCULATED OPTIMUM CROSS-SECTIONAL AREAS (in.^{2})

Problem	Member	SUMT	ConMin	NLPQ	IpOPT	CometBoards (SUMT)
3-bar truss	1	3.5356	3.5343	3.5330	3.5346	3.53339
	2	3.3382	3.3380	3.3380	3.3425	3.3394
	3	0.0101	0.01	0.0100	0.0116	0.0105
10-bar truss	1	23.5538	25.7191	24.2520	13.4781	23.5257
	2	0.1004	0.1000	0.1000	10.6337	0.1026
	3	1.9707	2.0182	1.9701	10.1799	1.9709
	4	14.3413	13.5977	14.1167	10.1787	14.2870
	5	25.2238	27.9918	26.0566	13.4609	25.2846
	6	0.1002	0.1000	0.1000	9.9380	0.1013
	7	12.8738	12.4031	12.2700	10.8161	12.4223
	8	12.8738	12.5652	12.5188	10.5432	12.9392
	9	20.3151	20.1855	19.8251	11.0619	20.2743
	10	0.1003	0.1000	0.1000	10.3493	0.1016
25-bar antenna tower	1	0.3015	0.6688	0.3070	1.3877	0.2992
	2	2.8265	3.2492	2.8287	6.4425	2.8280
	3	5.4753	5.2978	5.4726	4.9730	5.4766
	4	1.8049	1.9988	1.8091	5.2028	1.8136
	5	0.1119	0.7026	0.1199	1.5624	0.1175
	6	2.9120	2.8756	2.9104	3.3168	2.9109
	7	2.9482	2.7997	2.9450	3.2615	2.9477
	8	3.0179	2.9805	3.0182	3.0693	3.0194

Problem	Member	SUMT	ConMin	NLPQ	IpOPT	CometBoards (SUMT)
60-bar trussed ring	1	1.1472	1.1648	1.1467	1.1486	1.1478
	2	2.0272	2.0152	2.0258	2.0050	2.0286
	3	0.5002	0.5001	0.5000	0.9505	0.5006
	4	1.7672	1.8828	1.7655	2.1853	1.7632
	5	1.7663	1.8328	1.7618	2.2338	1.7624
	6	0.5763	0.5001	0.5711	1.2119	0.5762
	7	1.8550	1.8774	1.8403	1.7041	1.8595
	8	1.8251	1.8731	1.8527	1.7808	1.8287
	9	0.9884	0.5001	0.9932	1.3513	0.9892
	10	1.8863	1.8450	1.8873	1.6304	1.8844
	11	1.9382	1.8660	1.9297	1.4650	1.9363
	12	0.5002	0.5001	0.5000	0.9750	0.5005
	13	2.0139	2.0216	2.0131	2.0712	2.0151
	14	1.2442	1.2537	1.2442	1.6003	1.2446
	15	1.0154	1.0383	1.0148	1.3828	1.0166
	16	0.6864	0.7119	0.6897	0.5544	0.6861
	17	0.7239	0.7501	0.7231	0.8434	0.7240
	18	1.0579	1.0183	1.0673	1.1683	1.0575
	19	1.1229	1.1338	1.1233	1.7024	1.1232
	20	1.1510	1.1406	1.1502	1.1400	1.1515
	21	1.0658	1.0213	1.0687	0.9284	1.0659
	22	1.0480	0.7521	1.0486	1.0821	1.0486
	23	0.7008	0.7098	0.6928	0.5632	0.7006
	24	1.0293	1.0351	1.0262	1.1443	1.0305
	25	1.2585	1.2467	1.2578	1.2420	1.2590
Geodesic dome	1	0.0100	0.7057	0.0100	0.1764	0.0104
	2	0.0100	0.6031	0.0100	0.4244	0.0105
	3	0.0100	0.01	0.0100	2.0594	0.0102
	4	0.0100	0.01	0.0100	0.4416	0.0103
	5	0.0100	0.01	0.0100	0.2221	0.0102
	6	0.0100	0.01	0.0100	0.1025	0.0102
	7	0.0100	0.01	0.0100	0.2409	0.0102
	8	0.0100	0.01	0.0100	0.0340	0.0102
	9	0.4084	0.4009	0.4083	0.7031	0.4085
	10	0.3983	0.4719	0.3983	0.2241	0.3983
	11	0.3895	0.4828	0.3895	0.4711	0.3896
	12	0.3808	0.4707	0.3808	0.4274	0.3809
Composite plate (thickness, $in.^3$)	1	2.6829	2.7137	2.6782	4.6486	2.6819
	2	2.4288	2.4066	2.4332	3.1416	2.4308
	3	3.0934	3.0920	3.0921	1.7898	3.0935
Composite blade (thickness, $in.^3$)	1	0.0101	0.0100	0.4998	0.0100	0.0102
	2	0.0100	0.0100	0.0099	0.0100	0.0100

V. Conclusions

This paper presents the optimization results of several benchmark structural analysis and optimization problems obtained using the NASA GRC developed open source OpenMDAO framework. The results generated in OpenMDAO for performing multidisciplinary analysis and optimization were compared and verified with the results obtained from the same optimizers available in CometBoards. All optimization algorithms in OpenMDAO were executed with the same constant set or default set of parameters and control options. The results may be improved by tuning the parameters depending on the model and data. This comparison showed that most of the optimizers produced identical results with minor deviations; however, the computing time in OpenMDAO is much faster than that of CometBoards. The performance of NEWSUMT optimizer was improved dramatically in OpenMDAO, a 73 percent reduction in CPU time for the composite plate to 87 percent reduction in CPU time for the geodesic dome. Although ConMin converged faster than most of the other optimizers in the study, it produced infeasible designs for two of the seven problems. Overall, NLPQ was found to be the most efficient and robust optimization algorithm for the structural problems presented in this report.

Acknowledgments

The work is supported by the Subsonic Fixed Wing project under NASA's Fundamental Aeronautics Program. The authors thank the OpenMDAO team software developers, Justin S. Gray (team lead), Keith G. Marsteller, Kenneth T. Moore, Bret A. Naylor, Herbert W. Schilling, and Scott E. Townsend.

References

[1]OpenMDAO Documentation, http://openmdao.org

[2]MSC/NASTRAN 2005 Quick Reference Guide, Vol. 1&2, MacNeal-Schwendler Software Corporation, (2004).

[3]Miura, H.; and Schmit, L.A., Jr.: NEWSUMT: A FORTRAN Program for Inequality Constrained Function Minimization, Users' Guide. NASA CR–159070, 1979.

[4]Vanderplaats,G.N., CONMIN User's Manual,
www.eng.buffalo.edu/Research/MODEL/mdo.test.orig/CONMIN/manual.html

[5]Schittkowski, K.: NLPQL: A Fortran Subroutine for Solving Constrained Nonlinear Programming Problems. Annals of Operations Research, Vol. 5, 1986, pp. 485–500.

[6]IPOPT: https://projects.coin-or.org/Ipopt

[7]Deb K., "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation, Vol. 6, No. 2, 2002.

[8]Guptill, J.D., Coroneos, R.M., Patnaik, S.N., Hopkins, D.A., Berke, L., "CometBoards Users Manual Release 1.0," NASA TM 4537, 1996.

[9]Patnaik, S.N., Coroneos, R.M., Guptill, J.D., Hopkins, D.A. "Comparative Evaluation of Different Optimization Algorithms for Structural Design Applications," Int. J. Numerical Methods in Engr., Vol. 39, pgs. 1761–1774, 1996.

[10]MSC/Patran 2010, MacNeal-Schwendler Software Corporation.