AIAA Journal
Vol. 45, No. 11, November 2007

Solving Very Large-Scale Structural Optimization Problems

F. Hüttner*
AIRBUS Deutschland GmbH, 28199 Bremen, Germany
and
M. Grosspietsch†
inuTech GmbH, 90429 Nuremberg, Germany

DOI: 10.2514/1.23932

Large-scale structural parametric optimization problems still impose a challenge to computer hardware. A three-step approach to reduce the run time and memory requirements is proposed. It is based on the observed sparsity of the matrix of partial derivatives in structural optimization. The approach includes a differentiation scheme to compute smaller gradient matrices in parallel and assemble them employing the chain rule, an adaptive filtering framework for an effective selection of active constraints based on numerical value and engineering knowledge and the pairing of a known sequential convex programming algorithm with a preconditioned conjugate gradient solver for the internal matrices. Software has been developed and successfully applied to the optimization of the outer wing panels of a large military transporter aircraft.

Nomenclature

A	=	square coefficient matrix, $A \in \Re^{m_{\text{active}} \times m_{\text{active}}}$
b	=	right-hand side vector, $b \in \Re^{m_{\text{active}}}$
b_{foot}	=	stringer foot width
c	=	slack variables for constraint functions $h^k_{l_j}$, $c \in \Re^{m_{\text{active}}}$
D_r	=	$\text{diag}(d_r) \in \Re^{m_{\text{active}} \times m_{\text{active}}}$
D_s	=	$\text{diag}(d_s) \in \Re^{n \times n}$
D_t	=	$\text{diag}(d_t) \in \Re^{n \times n}$
d_r	=	dual variables, $d_r \in \Re^{m_{\text{active}}}$
d_s	=	dual variables, $d_s \in \Re^n$
d_t	=	dual variables, $d_t \in \Re^n$
f	=	objective function
f^k	=	objective function of the MMA approximation
g_j	=	constraint function j
h_{web}	=	stringer web height
$h^k_{l_j}$	=	constraint functions of the MMA approximation
I	=	identity matrix
$I^{h,k}_{j,+}$	=	design variable index set for which the derivative of h^k_j is greater than or equal to 0
$I^{h,k}_{j,-}$	=	design variable index set for which the derivative of h^k_j is less than 0
$I^k_{f,+}$	=	design variable index set for which the derivative of f^k is greater than or equal to 0
$I^k_{f,-}$	=	design variable index set for which the derivative of f^k is less than 0
J	=	Jacobian of constraint function vector h^k
$\mathfrak{J}$	=	set of constraint indices
L_i	=	asymptotes (MMA approximation)
L_μ	=	Lagrangian of the MMA approximation
l_i	=	lower bound to design variable x_i
l_j	=	constraint index
M	=	preconditioning matrix, $M \in \Re^{m_{\text{active}} \times m_{\text{active}}}$
m	=	number of constraints
m_{active}	=	number of active constraints
n	=	number of design variables
np	=	number of panels
n_x, n_y, n_{xy}	=	internal normal and shear forces
R	=	$\text{diag}(r) \in \Re^{m_{\text{active}} \times m_{\text{active}}}$
RF_j	=	reserve factor function j
r	=	slack variables for constraint functions $-c_j \leq 0$, $r \in \Re^{m_{\text{active}}}$
S	=	$\text{diag}(s) \in \Re^{n \times n}$
s	=	slack variables for constraint $\underline{x}_i - x_i \leq 0$, $i = 1, \ldots, n$, $s \in \Re^n$
T	=	$\text{diag}(t) \in \Re^{n \times n}$
t	=	slack variables for constraint, $x_i - \bar{x}_i \leq 0$, $i = 1, \ldots, n$, $t \in \Re^n$
t_{foot}	=	stringer foot thickness
t_{skin}	=	skin thickness
t_{web}	=	stringer web thickness
U_i	=	asymptotes (MMA approximation)
U'_{local}	=	FEM responses (FEM results)
u_i	=	upper bound to design variable i
x	=	design variable vector, equivalent to x_{global}
x_{global}	=	design variable vector
$\underline{x}_i$	=	lower bound for design variable x_i (MMA approximation)
$\bar{x}_i$	=	upper bound for design variable x_i (MMA approximation)
x_{local}	=	vector of physical properties
x^k_i	=	design variable value for iteration k
y_t	=	dual variables, $y_t \in \Re^{m_{\text{active}}}$
γ	=	right-hand side of the system of linear equations
Δh	=	finite difference
κ	=	condition number of a matrix
μ	=	homotopy parameter
τ^k_i	=	factor
∇L_μ	=	gradient of the Lagrangian of the MMA approximation

I. Introduction

THE parametric design optimization of complex industrial structural components—driven by local structural strength and stability—typically leads to problems with several thousands of design variables and hundreds of thousands of constraints [1–3].

Mathematically, the pure size of a problem is not the only (or even best) indicator for the difficulty to solve it. Nonlinearity of constraint and objective functions, steadiness, numerical accuracy, sharp or flat optima, etc., may impose more severe solution difficulties. On the other hand, large problems can be solved easily when each design variable is dominated by a corresponding constraint. The solution is "fully stressed." A simple, but effective algorithm can be used to

Received 15 March 2006; revision received 16 June 2007; accepted for publication 2 July 2007.

*Stress Engineer, Composite Structural Engineering.

†Mathematician.

solve such problems, even when they are very large. This approach has been widely used in the engineering community and has even been recently introduced in MSC/Nastran [4].

The considered optimization problems here are "real-world" parametric structural optimization problems and are described by the following engineering rather than formal features:

1) Internal loads and other responses like strains, displacements, etc., are calculated by the finite element method (FEM).

2) The structure is subjected to different load conditions (load cases).

3) The proof of the structure is done based on different criteria, and a combination of criteria has to be considered for most locations.

4) The constraints are mostly nonlinear and often complex formulas or even numerical methods (e.g., energy method for buckling) are coded in specialized in-house programs.

5) Because of manufacturability and the nature of the constraint functions the spatial range of design variables and constraint functions is different, making direct links required for a fully stressed design impossible.

The run time to solve complex finite element models with an appropriate number of load cases and a large number of design variables and constraints precludes the use of genetic algorithms and makes the procedure of building a response surface impractical [5]. Standard gradient-based algorithms (sequential linear programming, sequential quadratic programming, sequential convex programming —SLP, SQP, SCP) require calculation and storage of a huge amount of (first- and second-) order gradient data, while the optimizer itself has to solve large linear systems of equations repeatedly. Despite increased computing resources, run time and memory requirements are comparatively high.

Different strategies have been employed to increase the performance of the algorithms: filtering to reduce the number of active constraints both for the optimizer itself and during gradient calculation, Taylor-series linearization (both included in MSC/Nastran [4]) instead of new design point evaluations for each iteration, and enhanced optimization algorithm (BIGDOT [6]) with less gradient information needed.

Although the increase of effectiveness is of no doubt, a further reduction of run time and memory requirements is still desirable. In the following a three-step approach is presented. At first, a differentiation scheme for the optimization problem is proposed, tailored for structural optimization problems. Without loss of generality it allows an effective assembly of the sensitivity matrix, enabling parallel computing, with additional advantages of higher precision of the derivatives and more transparency. Secondly, the reduction of the number of active constraints is achieved by a filtering framework. Finally, an effective preconditioned conjugate gradient (PCG) solver is proposed as an upgrade to a known but not widely used powerful SCP algorithm.

This approach to solve large-scale, parametric, and continuous optimization problems of structural strength is generally applicable to constrained nonlinear optimization problems based on finite element results.

II. Differentiation Scheme

Mathematically a continuous nonlinear optimization problem can be stated as

Minimize $f(x)$

subject to

$$\begin{aligned} g_j(x) \le 0, \quad j = 1, \ldots, m \\ l_i \le x_i \le u_i, \quad i = 1, \ldots, n \end{aligned} \tag{1}$$

where $f(x)$ is the objective function, $g(x)$ are m constraint functions, and x is the design vector of n independent design variables, limited to lower and upper bounds l and u, respectively.

Constraint functions of structural strength are usually called safety factors or reserve factors (RF). Other constraint functions such as allowable deformations, geometry ratios, or manufacturing constraints are not considered in the following but could easily be added, because they do not change the general nature of the optimization problem. The basic concept of reserve factors is that a structural response or a scalar mapping of a combination of responses does not exceed an allowable value, that is, reserve factors are required to be equal or greater than 1. The constraints are then

$$g_j = 1 - RF_j \tag{2}$$

or

$$g_j = \frac{1}{RF_j} - 1 \tag{3}$$

But, this concept, suggesting calculation of reserve factors is equal to imposing limits on FEM responses, is too narrow for practical use. In general, a reserve factor is an engineering assessment on the basis of material values and other local physical properties (or their representation in the FEM model—this representation is meant when "properties: are used in the following):

$$RF_j = f_j(x_{\text{local}}(x_{\text{global}}), U'_{\text{local}}(x_{\text{local}}(x_{\text{global}}))) \tag{4}$$

where x_{global} denotes the optimization vector (renamed from x for a better differentiation from x_{local}), x_{local} is the local structure dimensions, and U'_{local} is the local FE results (structural responses). It is assumed that the vector x_{local} describes the physical properties of the structure, whereas x_{global} describes the optimization problem. x_{local} is a function p of x_{global} (the fact that some components of x_{local} as the Young's modulus will be constant during the optimization can be ignored for the following):

$$p\colon x_{\text{global}} \to x_{\text{local}} \tag{5}$$

The derivatives of Eq. (4) can be calculated employing the chain rule as follows:

$$\begin{aligned} \frac{\partial RF_j}{\partial x_{\text{global},i}} &= \sum_k \frac{\partial RF_j}{\partial x_{\text{local},k}} \cdot \frac{\partial x_{\text{local},k}}{\partial x_{\text{global},i}} \\ &+ \sum_l \sum_o \frac{\partial RF_j}{\partial U'_{\text{local},l}} \cdot \frac{\partial U'_{\text{local},l}}{\partial x_{\text{local},o}} \cdot \frac{\partial x_{\text{local},o}}{\partial x_{\text{global},i}} \end{aligned} \tag{6}$$

The jth reserve factor depends on k local properties and l local structural responses, while in turn the responses depend on o other properties.

The application of the chain rule to calculate derivatives of the constraint function depending on structural responses is not new (see MSC/Nastran [4]) and mathematically almost trivial. But Eq. (6) offers great potential, when it is not only seen as a *multiplication* rule, but as an *assembling* rule. Then, it reduces the number of required derivatives, decouples the derivatives to make parallel computation possible, and shows that the gradient or sensitivity matrix—the matrix containing all partial derivatives of constraint j with respect to design variable i—must be sparse.

The main idea for the following is that in the case of large optimization problems, that is, a large n and m, the dependencies of reserve factor functions k and l are considerably bigger than 1, but still small, and do not grow with an increasing problem size. The reason for this is that a complex FEM model cannot produce direct responses for all critical failure modes; think, for example, of a lightweight structure composed of thousands of locally supported buckling fields or a complete aircraft model with millions of rivets to be assessed. Thus, reserve factors are calculated based on the structural responses from the FEM model and additional material and geometry information. On the other hand, theoretical and/or experimental models with dozens of free parameters are difficult or impossible to develop, and so limiting k and l.

An example (Fig. 1) is the calculation of the stability reserve factor of a flat plate between two rib and stringer planes. The stringer foot is integrated on the skin. Symmetric geometry is assumed per plate. Material values are constant.

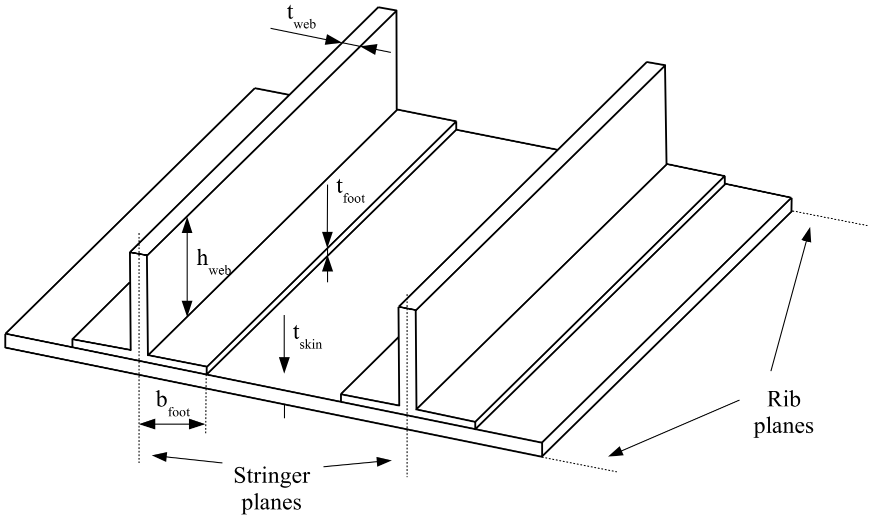


Fig. 1 Example geometry description.

Without going into detail of the calculation method itself, the critical buckling load depends on the skin thickness t_{skin}, stringer foot width b_{foot} and thickness t_{foot}, and the reserve factor additionally on the normal and shear loads in the skin n_x, n_y, and n_{xy}:

$$RF = f(t_{\text{skin}}, t_{\text{foot}}, b_{\text{foot}}, n_x, n_y, n_{xy}) \tag{7}$$

The differentiation of reserve factor j over design variable i for the plate (*) is then

$$\begin{aligned}\frac{\partial RF_j}{\partial x_{\text{global},i}} &= \frac{\partial RF^{(*)}}{\partial t_{\text{skin}}} \cdot \frac{\partial t_{\text{skin}}}{\partial x_{\text{global},i}} + \frac{\partial RF^{(*)}}{\partial t_{\text{foot}}} \cdot \frac{\partial t_{\text{foot}}}{\partial x_{\text{global},i}} + \frac{\partial RF^{(*)}}{\partial b_{\text{foot}}} \\ &\cdot \frac{\partial b_{\text{foot}}}{\partial x_{\text{global},i}} + \frac{\partial RF^{(*)}}{\partial n_x} \cdot \sum_o \frac{\partial n_x}{x_{\text{local},o}} \cdot \frac{\partial x_{\text{local},o}}{\partial x_{\text{global},i}} + \frac{\partial RF^{(*)}}{\partial n_y} \\ &\cdot \sum_o \frac{\partial n_y}{x_{\text{local},o}} \cdot \frac{\partial x_{\text{local},o}}{\partial x_{\text{global},i}} + \frac{\partial RF^{(*)}}{\partial n_{xy}} \cdot \sum_o \frac{\partial n_{xy}}{x_{\text{local},o}} \cdot \frac{\partial x_{\text{local},o}}{\partial x_{\text{global},i}}\end{aligned} \tag{8}$$

Table 1 compares the size of the gradient matrix and the number of partial derivatives of the reserve factor function for different sizes of the optimization problem, measured in number of panels np. Ten load cases and one reserve factor per case are assumed as well as five design variables per panel (the three from above plus stringer height h_{web} and thickness t_{web}).

The number of partial derivatives in the gradient matrix is $m \times n$, the number of required multiplications in Eq. (6) is $m \times n \times (k + l) \times o$. But the number of the partial derivatives of the reserve factor functions itself is only $m \times (k + l)$.

If, as assumed above, the number of constraints and design variables increase proportional with the size of a problem, while the complexity of the assessment function, measured as the number of local variables, remains constant, then the number of (time consuming) differentiation of these functions increases only linearly, while the gradient matrix increases quadratic in size.

The chain rule equitation (6) also decouples the differentiations. From a practical point of view this has two advantages. The terms $\partial RF/\partial x_{\text{local}}$ and $\partial RF/\partial U'_{\text{local}}$ can be evaluated analytically if the functions are explicitly known and easily structured, but in the general case of more sophisticated functions, involving complex input and/or numerical methods, they have to be approximated, for example, by forward differences

$$\begin{aligned}\frac{\partial RF}{\partial x_{\text{local}}} &\approx \frac{f(x_{\text{local}} + \Delta h, U'_{\text{local}}) - f(x_{\text{local}}, U'_{\text{local}})}{\Delta h} \\ \frac{\partial RF}{\partial U'_{\text{local}}} &\approx \frac{f(x_{\text{local}}, U'_{\text{local}} + \Delta h) - f(x_{\text{local}}, U'_{\text{local}})}{\Delta h} \\ &\Delta h, x_{\text{local}} \in \Re\end{aligned} \tag{9}$$

Because the finite difference Δh is applied directly to the variables of f, it can be controlled and adjusted, and thus, the accuracy of the derivatives is increased. And, as the differentiations do not depend on other derivatives, all calculations can be done in parallel.

The sparsity of the gradient matrix can also be derived from equitation (6). As demonstrated, the number of reserve factor functions is small compared to the size of the gradient matrix. The derivatives $\partial x_{\text{local}}/\partial x_{\text{global}}$ build the unity matrix in case that there is a one-to-one relationship between a local structure dimension and a global optimization variable. In general, the function and its derivatives are well known and simple. The term $\partial U'_{\text{local}}/\partial x_{\text{local}}$ is calculated employing either the direct or the adjoint method which is a standard feature of today's FE packages. In the case of stresses and strains $\partial U'_{\text{local}}/\partial x_{\text{local}}$ has a strong sparse character. So, the gradient matrix can be expected to be sparse because all the basic matrices of derivatives are sparse and contain neither full columns nor full rows.

It should be mentioned that—as practical advantages—the definition of the FE-sensitivity analysis can be kept untouched, if the number of global optimization variables or the function x_{local} (x_{global}) changes. The derivatives $\partial x_{\text{local}}/\partial x_{\text{global}}$ can be kept constant through the whole optimization process in case that there is a direct assignment or a linear interpolation between global design variables and local structural dimensions.

Table 1 Comparison of the number of partial derivatives in the example problem

No. of panels np	No. of partial derivatives $(5 \times k) \times (10 \times k)$	No. of partial reserve factor derivatives $10 \times k \times 6$
1	50	60
100	500,000	6,000
10,000	5,000,000,000	600,000

III. Filtering Framework

To solve the optimization problem (1), a constraint filtering framework is established, splitting the problem into a series of smaller sized problems. The basic idea for the method stems from cutting plane algorithms employed in the bounding step of branch and cut algorithms applied to combinatorial optimization problems. Such methods are widely used to reduce the number of constraints, based on the observation that only a relative small subset of constraints is critical for the optimal design [4,6,7]. To describe the idea the following optimization problem is introduced:

Minimize $f(x)$
subject to

$$\begin{aligned} &g_j(x) \leq 0, \quad j \in \mathfrak{J} \subseteq \{1, \ldots, m\} \\ &l_i \leq x_i \leq u_i, \quad i = 1, \ldots, n \end{aligned} \tag{10}$$

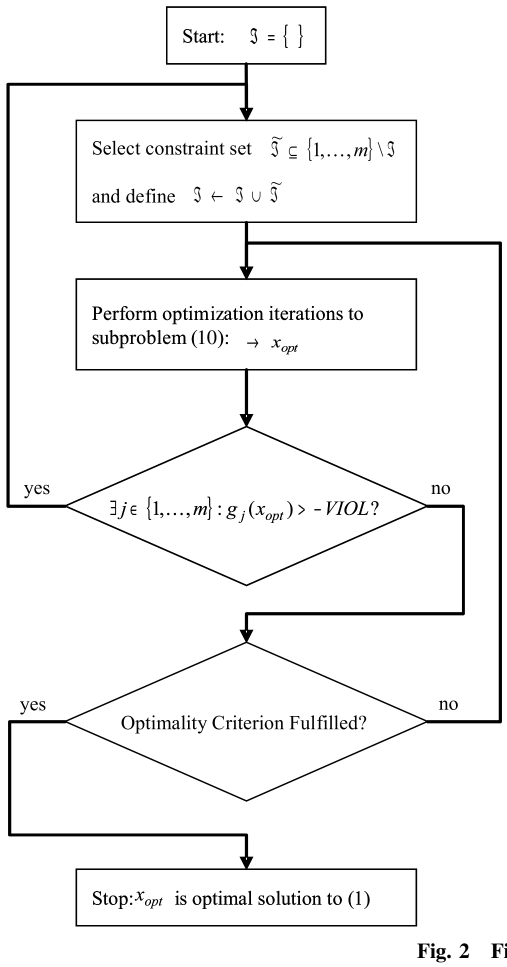
where the so-called subproblem (10) contains only a subset of all constraints $g_j(x) \leq 0$ considered in the original problem (1). Because Eq. (10) operates on a larger set of feasible values for design variables the objective function value for an optimal solution to Eq. (10) is lower than or equal to the objective function value for an optimal solution to Eq. (1). This means, if an optimal solution to the subproblem (10) satisfies all constraints $g_j(x) \leq 0$, from the underlying problem (1), the solution is also optimal for the problem (1). The basic algorithm consists of generating and solving subproblems until all constraint functions of the optimization problem (1) are satisfied. Subproblems are generated by adding constraints from Eq. (1) to Eq. (10), which were not considered before. The overall filtering framework to solve Eq. (1) is shown in the flowchart (Fig. 2).

A common selection strategy in step 2 of Fig. 2 is to add the *r*-most critical constraint functions to the subproblem, which is purely based on the constraint's value and constitutes a reasonable, general-purpose "black-box" approach. There are two disadvantages observed using that selection strategy. Both are related to the fact that many redundant constraints are added, even if it would be sufficient to add only a small subset. Besides the large number of active constraints, meaning increased run time, the even more critical issue is the violation of some regularity constraint qualification (linear independence of gradients), usually causing an abnormal termination of the underlying optimization algorithm in step 3 of Fig. 2. One method to overcome this violation would be to extend the selection strategy by some kind of gradient information. Unfortunately that would mean that in every step the full set of constraint derivatives must be generated, which is not reasonable due to the huge effort in terms of run time and hardware resources. The solution proposed here is to employ the engineers knowledge to group the constraints based on an engineering judgement about similar behavior and passing only one constraint per group to the new subproblem based on a chosen threshold. In case of engineering optimization problems the physical representations of constraints are well known. They are assumed to be reserve factors and thus are related to typical failure modes (static strength, buckling, crack propagation, etc.). Reserve factors of different load cases may change proportionally with a design change. This is the case when a failure mode is dominated by one load component only as it is the case for the crack opening by a normal load or the Euler instability by pressure loads. Also, when many reserve factors are calculated within an area that is described by only one (local or global) design variable, reserve factors may change proportionally with a change. An opposite example is the skin buckling from the example above, where the normal load in the stringer direction is redistributed between the skin and the stringer depending on their respective areas and stiffnesses and, therefore, the influence of the two others on the reserve factor may vary heavily, when the area ratio is changed to a new design point.

An example for defining one group would be to summarize all reserve factors of a single type for one specific position over all load cases. For example, for one stringer between two ribs, the blade stability constraints for all load cases define one group.

IV. Optimizer

Subproblems (10), generated by the filtering framework, are processed by the sequential convex programming algorithm SCPIP [8,9], enhanced by an effective PCG. Like most optimization


Start: $\mathfrak{J} = \{\}$
Select constraint set $\tilde{\mathfrak{J}} \subseteq \{1,\ldots,m\} \setminus \mathfrak{J}$ and define $\mathfrak{J} \leftarrow \mathfrak{J} \cup \tilde{\mathfrak{J}}$
Perform optimization iterations to subproblem (10): $\rightarrow x_{opt}$
$\exists j \in \{1,\ldots,m\} : g_j(x_{opt}) > -VIOL$?
yes
no
Optimality Criterion Fulfilled?
yes
no
Stop: x_{opt} is optimal solution to (1)

1.Initialization

2.Select a set of constraints not considered in the subproblem and add those to the set of constraints $\mathfrak{J}$ considered in the subproblem (10).

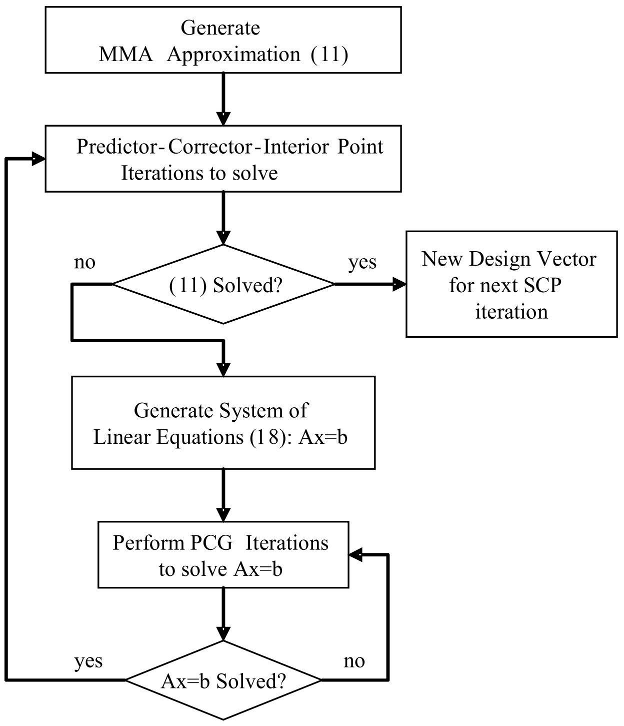
3.Perform a couple of optimization iterations to the newly generated subproblem.

4.Check if the retrieved design vector fulfills all constraint functions.

5.If no constraint function is violated for the design vector check optimality condition, else continue with step 1.

6.If the optimality criterion was fulfilled the design vector of the current iteration is optimal for the original program (1) otherwise go to step 2 and perform further optimization iterations to the problem (10)

Fig. 2 Filtering framework flowchart.



1. Replace the optimization problem (10) by its MMA approximation (11).
2. Perform one or more predictor-corrector-interior point method iterations until the optimization problem (11) is solved. The design vector for the next iteration is then derived from the solution to (11).
   a) One iteration of the predictor-corrector-interior point method consists of solving a system of linear equations, Ax=b for two different right-hand sides b.
   b) These systems of linear equations are solved by the preconditioned conjugated gradient solver.

Fig. 3 Flowchart of an SCP iteration.

algorithms for nonlinear optimization problems, the used SCP method iteratively replaces the original problem by a sequence of easier to solve approximations. SCPIP implements among others the method of moving asymptotes (MMA) which is known to work well for displacement dependent functions. To show where the advantages of a preconditioned conjugate gradient solver in combination with the SCP algorithm can be seen, the following paragraphs give an overview of important components of the SCP algorithm. A more detailed description can be found in [10]. The flowchart in Fig. 3 shows the steps which are performed for one SCP iteration, that is, how the MMA approximation, the predictor corrector interior point method, and the PCG solver interact to generate a new design vector.

The SCP optimization routine replaces for each iteration k the original problem (10) by the well-understood convex MMA-approximation Fig. 3, which is solved by a predictor corrector interior point method.

A. MMA Approximation

For iteration k the MMA approximation is described by the following convex optimization program (11):

Minimize $f^k(x)$

subject to

$$\begin{array}{lll} h^k_{l_j}(x) \le 0, & & j = 1, \dots, m_{\text{active}} \\ \underline{x}_i \le x_i \le \bar{x}_i, & & i = 1, \dots, n \\ l_j \in \mathfrak{J}, & l_j \ne l_k, & \forall\ j \ne k \in \{1, \dots, m_{\text{active}}\} \end{array} \tag{11}$$

where the approximating functions f^k and $h^k_{l_j}$ are defined as

$$f^k(x) = f(x^k) + \sum_{i \in I^k_{f,+}} \left[\frac{\partial f(x^k)}{\partial x_i} \cdot \left(\frac{(U^k_i - x^k_i)^2}{(U^k_i - x_i)} - (U^k_i - x^k_i) \right) + \tau^k_i \cdot \frac{(x_i - x^k_i)^2}{(U^k_i - x_i)} \right] - \sum_{i \in I^k_{f,-}} \left[\frac{\partial f(x^k)}{\partial x_i} \cdot \left(\frac{(x^k_i - L^k_i)^2}{(x_i - L^k_i)} - (x^k_i - L^k_i) \right) - \tau^k_i \cdot \frac{(x_i - x^k_i)^2}{(x_i - L^k_i)} \right] \tag{12}$$

$$h^k_{l_j}(x) = h_{l_j}(x^k) - \sum_{i \in I^{h,k}_{f,+}} \left[\frac{\partial h^k_{l_j}(x^k)}{\partial x_i} \cdot \left(\frac{(U^k_i - x^k_i)^2}{(U^k_i - x_i)} - (U^k_i - x^k_i) \right) \right] - \sum_{i \in I^{h,k}_{f,-}} \left[\frac{\partial h^k_{l_j}(x^k)}{\partial x_i} \cdot \left(\frac{(x^k_i - L^k_i)^2}{(x_i - L^k_i)} - (x^k_i - L^k_i) \right) \right] \tag{13}$$

with design variable index sets,

$$\begin{array}{ll} I^k_{f,-} = \left\{ i\colon \dfrac{\partial f(x^k)}{\partial x_i} < 0 \right\}, & I^k_{f,+} = \left\{ i\colon \dfrac{\partial f(x^k)}{\partial x_i} \ge 0 \right\} \\ I^{h,k}_{j,-} = \left\{ i\colon \dfrac{\partial h_{l_j}(x^k)}{\partial x_i} < 0 \right\}, & I^{h,k}_{j,+} = \left\{ i\colon \dfrac{\partial h_{l_j}(x^k)}{\partial x_i} \ge 0 \right\} \end{array} \tag{14}$$

The MMA approximation is a linearization in the reciprocal variables $1/(x_i - L_i)$ or $1/(x_i - U_i)$ depending on the sign of the derivative, where an additional term

$$\tau^k_i \cdot \frac{\left(x_i - x^k_l\right)^2}{(U^k_i - x_i)}$$

is added for the objective function. The asymptotes L_i and U_i are modified for each iteration to retrieve a better convergence, further controlling the upper and lower bounds $\bar{x}_i$ and $\underline{x}_i$ for the variables. To keep the MMA approximation as small as possible, but still of sufficient size, only constraints from the constraint set $\mathfrak{J}$ are taken into account, tending to become infeasible. These constraints are called active and are determined by a user provided threshold. If a constraint value becomes greater than this threshold (e.g., -0.8), the constraint is set active, otherwise inactive, that is, $h^k_{l_j}(x) \ge -0.8 \Rightarrow$ constraint $l_j \in \mathfrak{J}$. The MMA approximation consists only of m_{active} constraints, where $m_{\text{active}} \le |\mathfrak{J}| \ll m$. It should also be mentioned that although MMA was motivated to approximate displacement dependent functions, it is applicable to general functions.

The convex optimization problems (11) are solved by a predictor–corrector–interior point method to retrieve a new design vector for the subproblem (10). For predictor–corrector–interior point methods the main effort can be seen in repetitively solving linear systems of equations. The linear systems of equations for the considered optimization problem have a very sparse character. Methods using this characteristic must be employed to solve the MMA approximation subproblem effectively. The following paragraph shows the basic idea of interior point methods.

B. Predictor–Corrector–Interior Point Method

Originally, interior point methods were developed for solving linear problems [11], but the same principle can be applied to convex optimization problems (like the MMA approximation) and seems to be superior to the dual method for large-scale problems (15) [10]. Interior point methods can be motivated as follows. To formulate an unconstrained optimization problem, all inequality constraints (box constraints included) are converted to equality constraints by introducing positive slack variables (c, r, s, t), where the positivity is enforced by logarithmic barrier functions.

$$\text{Minimize} \quad f^k(x) - \mu\left(\sum_{j=1}^{m_{\text{active}}} \ln r_j + \sum_{i=1}^{n} \ln s_i + \sum_{i=1}^{n} \ln t_i\right)$$

$$\begin{aligned} \text{subject to} \quad h^k_{l_j}(x) + c_j = 0, &\quad j = 1, \ldots, m_{\text{active}} \\ -c_j + r_j = 0, &\quad j = 1, \ldots, m_{\text{active}} \\ \underline{x}_i + x_i + s_i = 0, &\quad i = 1, \ldots, n \\ x_i - \bar{x}_i + t_i = 0, &\quad i = 1, \ldots, n \\ l_j \in J & \end{aligned} \tag{15}$$

Slack variables r are not absolutely necessary but otherwise the constraints were not allowed to become identically zero. Introducing dual variable vectors y, d_r, d_s, and d_t and the homotopy parameter μ the Lagrangian of Eq. (15) reads as follows:

$$\begin{aligned} L_\mu(x, y, c, r, s, t, d_r, d_s, d_t) &= f^k(x) + y^T \cdot (h^k(x) + c) + d_r^T \\ &\cdot (-c + r) + d_s^T \cdot (\underline{x} - x + s) + d_t^T \cdot (x - \bar{x} + t) \\ &- \mu\left(\sum_{j=1}^{m} \ln r_j + \sum_{i=1}^{n} \ln s_i + \sum_{i=1}^{n} \ln t_i\right) \end{aligned} \tag{16}$$

Applying a Newton step $[\nabla^2 F(x^k)\nabla x = -\nabla F(x^k)]$ to the first-order optimality condition $\nabla L_\mu = 0$ leads to the following system of linear equations:

$$\begin{pmatrix} \nabla_{xx}L & J & & & & & & -I & I \\ J^T & & I & & & & & & \\ & I & & & & & -I & & \\ & & & D_r & & & R & & \\ & & & & D_s & & & S & \\ & & & & & D_t & & & T \\ & & -I & I & & & & & \\ -I & & & & I & & & & \\ I & & & & & I & & & \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta c \\ \Delta r \\ \Delta s \\ \Delta t \\ \Delta d_r \\ \Delta d_s \\ \Delta d_t \end{pmatrix} = -\nabla L_\mu \tag{17}$$

with I denoting the identity matrix, J the Jacobian of the constraint functions h^k,

$$\nabla_{xx}L = \frac{\partial^2}{\partial x^2} f^k + \frac{\partial}{\partial x} J \cdot y$$

and S, R, T, D_t, D_r, and D_s denote diagonal matrices containing the slack variables r, s, t or the dual variables d_t, d_r, and d_s on the diagonal. Noting that the problem is separable the system can be further reduced to

$$(J^T \cdot \theta^{-1} \cdot J - D_r \cdot R) \cdot \Delta y = \gamma \tag{18}$$

with $\theta = \nabla_{xx}L + S^{-1} \cdot D_s + T^{-1} \cdot D_T$.

Algorithm 1 The method of preconditioned conjugate gradients

```
$i \leftarrow 1$                                                  1
residual $\leftarrow b - Ax$                                      2
$d \leftarrow M^{-1}$residual                                     3
$\delta_{\text{new}} \leftarrow$ residual$^T d$                   4
$\delta_0 \leftarrow \delta_{\text{new}}$                         5
While $i < i_{\max}$ and $\delta_{\text{new}} > \varepsilon^2 \delta_0$ do   6
    $q \leftarrow Ad$                                             7
    $\alpha \leftarrow \frac{\delta_{\text{new}}}{d^T q}$         8
    $x \leftarrow x + \alpha d$                                   9
    residual $\leftarrow$ residual $- \alpha q$                   10
    $s \leftarrow M^{-1}$residual                                 11
    $\delta_{\text{old}} \leftarrow \delta_{\text{new}}$          12
    $\delta_{\text{new}} \leftarrow$ residual$^T s$               13
    $\beta \leftarrow \frac{\delta_{\text{new}}}{\delta_{\text{old}}}$   14
    $d \leftarrow s + \beta d$                                    15
    $i \leftarrow i + 1$                                          16
```

The coefficient matrix of Eq. (18) has dimension $m_{\text{active}} \times m_{\text{active}}$ and must be solved for each iteration of the predictor–corrector method for two different right-hand sides: the first is to estimate the homotopy parameter μ and in the second step the results are used to determine a new design vector for the next iteration. This is repeated until two subsequent iterates differ by at most a given threshold. To solve the linear system of Eq. (18) a preconditioned conjugated gradient solver can be used.

C. Preconditioned Conjugated Gradient Solver

Let $Ax = b$ denote the linear system of Eq. (18). Because the coefficient matrix A of Eq. (18) is symmetrical and positive semidefinite, either the Cholesky decomposition in combination with forward–backward substitution or a preconditioned conjugate gradient solver should be the solver of choice. For the considered problem the conjugated gradient solver combined with a preconditioning strategy showed the best performance, where preconditioning turned out to be essential for convergence. The convergence behavior of the conjugated gradient solvers strongly depends on the spectral condition number

$$K = \frac{\max(|\lambda| : \lambda \text{ is eigenvalue of } A)}{\min(|\lambda| : \lambda \text{ is eigenvalue of } A)}$$

of the coefficient matrix A. The idea behind preconditioning is to consider the problem $M^{-1}Ax = M^{-1}b$ instead of solving $Ax = b$ directly, where the condition number of $M^{-1}A$ should be smaller than the condition number for A. Further the matrix M should be easy to invert, which holds for Jacobi preconditioning, where the matrix M is chosen as $M = \text{diag}(A)$. The method of preconditioned conjugate gradients to solve a system of linear equations $Ax = b$ is described in Algorithm 1 [12].

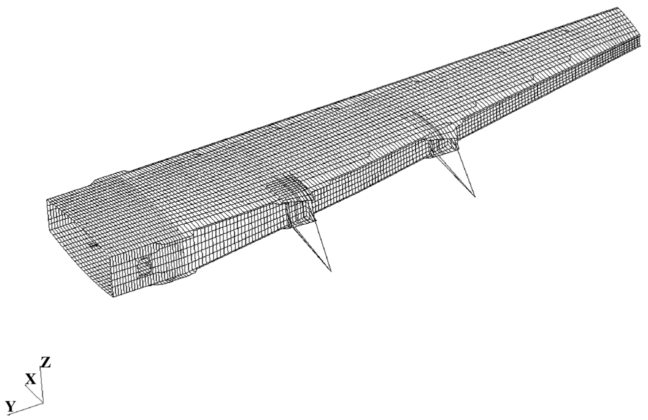
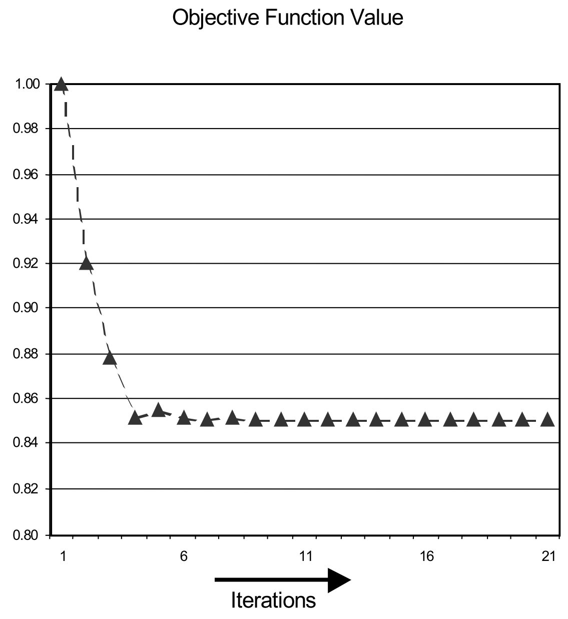


Fig. 4 Wing FEM model.



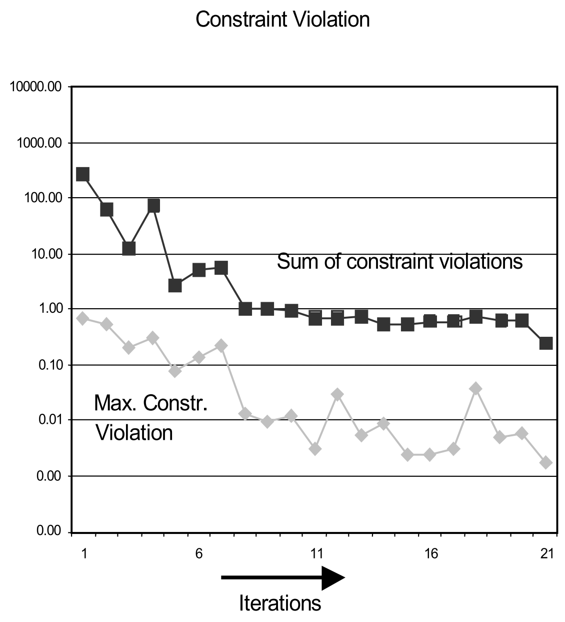


Fig. 5 History of weight and constraint violation.

The only step where the coefficient matrix is used is step 7, the calculation of the product of the coefficient matrix A with a vector d. This multiplication can be performed using Eq. (18), such that the matrix A is not known explicitly, whose assembly can be very time consuming. One further advantage is that the sparsity of the Jacobian J can be employed fully during the solution of the system of linear equations, even if the sparsity does not hold for the entire coefficient matrix $A = (J^T \cdot \theta^{-1} \cdot J - D_r \cdot R)$.

V. Example

The industrial applicability of the concepts is demonstrated by the optimization of the outer wing panels of a large military transporter aircraft (Fig. 4) during the detailed design phase. In this phase, the global arrangement is fixed, here a traditional design of stringer stiffened skin on a rib-spar frame, along with main geometry data, for example, stringer positions.

In the example here, with several hundred skin and stringer sections, a set of 1500–2800 design variables is required for a detailed structure description, while the number of reserve factors/constraints sums up to 1.4×10^6 for the two-dozen most critical load cases.

Design variables are mostly directly linked to skin thickness and stringer cross-section dimensions at each required location [to use the terminology from above, all dimensions and thickness form the (variable part of) vector x_{local}]. Stringer dimensions especially, but skin thickness as well, have also been interpolated by linear or quadratic functions for some optimization runs.

Constraints are mainly strength and stability reserve factors. The lightweight structure is critical to local buckling of skin and stringers and (local) Euler instability. The strength of the material as well as of the bolted joints has to be assessed. Secondary bending effects due to internal pressure or predeformed geometry increase local loads and their effect are to be considered, while a mismatch in thermal expansion coefficients of rib and skin material also creates additional loads.

The FEM model itself comprises shell and bar elements with over 60,000 degrees of freedom.

MSC/Nastran is used as the solver for the linear FEM model as well as for the computation of gradients of the FE results, called "sensitivity analysis" in MSC/Nastran.

The FEM gradient computations are run in parallel per load case and reserve factor type on a workstation cluster using a queuing system for the parallel computing.

Depending on the initial design point the optimizer requires 15–20 iterations, which takes in total about 20 h until an (engineering) feasible, near-optimum design is achieved. Each iteration required a new linear FEM solution and new sensitivity and gradient computations. A typical (normalized) weight and constraint violation sum history is shown in Fig. 5. Note that for large-scale optimization problems with counteracting constraints infeasible starting points are nearly inevitable. At the final iteration the index set $\mathfrak{J}$ contains about 12,000 constraints (out of 1.4×10^6) from which 8500 were active.

VI. Conclusions

This paper demonstrates a new approach to parametric structural optimization software. It was driven by the need in an industrial large-scale project to solve optimization problems with a large number of design variables and critical constraints in an acceptable run time within the available computer architectures. The optimization software had to integrate with MSC/Nastran and existing software tools to compute complex and highly nonlinear reserve factors. A secondary goal was to provide transparency and traceability within the huge amount of data typically generated by optimization software.

The mathematical concepts employed are an adjunct to existing methods used in commercial software; the novel features are a differentiation scheme with an adaptive filtering framework and the extension of a known stable and highly convergent optimization algorithm by a preconditioned conjugate gradient solver for sparse matrices.

A practical example is shown which weight optimizes a large military transporter's wing panel with 1.4×10^6 side constraints—around 12,000 of them critical at the final design point—and 2800 design variables; in this example the software ran in an acceptable time on a typical industrial computer architecture.

References

[1] Vanderplaats, G. N., "Very Large Scale Continuous and Discrete Variable Optimization," *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA, Reston, VA, 2004.

[2] Balabanov, V. O., and Venter, G., "Multi-Fidelity Optimization with High-Fidelity Analysis and Low-Fidelity Gradients," *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA, Reston, VA, 2004.

[3] Schuhmacher, G., Murra, I., and Laxander, A. a. o., "Multidisciplinary Design Optimization of a Regional Aircraft Wing Box," *9th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA, Reston, VA, 2002.

[4] MSC/Nastran, Design Sensitivity and Optimization User's Guide, 2004.

[5] Venter, G., Haftka, R. T., and Starnes, J. H., "Construction of Response Surfaces for Design Optimization Applications," *6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, AIAA, Reston, VA, 1996.

[6] Vanderplaats Research & Development, Inc., Colorado Springs, CO, Genesis 7.5 Reference Manual, 2002.

[7] Engineous Software, Inc., Cary, NC, iSight User's Guide, Ver. 7.1, 2002.

[8] Zillober, Ch., and Vogel, F., "Adaptive Strategies for Large Scale Optimization Problems in Mechanical Engineering," *Recent Advances in Applied and Theoretical Mathematics*, edited by N. Mastorakis, World Scientific and Engineering Society Press, Athens, Greece, 2000, pp. 156–161.

[9] Zillober, Ch., and Vogel, F., "Solving Large Scale Structural Optimization Problems," *Proceedings of the 2nd ASMO UK/ISSMO Conference on Engineering Design Optimization*, ASMO UK, Swansea, 2000.

[10] Zillober, Ch., "SCPIP. An Efficient Software Tool for the Solution of Structural Optimization Problems," *Structural and Multidisciplinary Optimization*, Vol. 24, No. 5, 2002, pp. 362–371.

[11] Wright, S. J., *Primal-Dual Interior-Point Methods*, SIAM Publications, Berlin, 1997.

[12] Shewchuk, J. R., "*An Introduction to the Conjugate Gradient Method Without the Agonizing Pain Edition 1 1/4*, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2004.

A. Messac
Associate Editor