AIAA JOURNAL
Vol. 38, No. 1, January 2000

Bilevel Integrated System Synthesis

Jaroslaw Sobieszczanski-Sobieski*
NASA Langley Research Center, Hampton, Virginia, 23681
and
Jeremy S. Agte† and Robert R. Sandusky Jr.‡
George Washington University, Washington, D.C. 20052

Bilevel integrated system synthesis is a method for the optimization of engineering systems by decomposition. It separates the system level optimization, having a relatively small number of design variables, from the potentially numerous subsystem optimizations that may each have a large number of local design variables. The subsystem optimizations are autonomous and may be conducted concurrently. Subsystem and system optimizations alternate, linked by sensitivity data, producing a design improvement in each iteration. Starting from a best guess initial design, the method improves that design in iterative cycles; each cycle comprises two steps. In step one, the system level variables are frozen and the improvement is achieved by separate, concurrent, and autonomous optimizations in the local variable subdomains. In step two, further improvement is sought in the space of the system level variables. Optimum sensitivity data link the second step to the first. The method prototype was implemented using MATLAB® and iSIGHT programming software and tested on a simplified, conceptual level supersonic business jet design, and a detailed design of an electronic device. Satisfactory convergence and favorable agreement with the benchmark results were observed. Modularity of the method is intended to fit the human organization and map well on the computing technology of concurrent processing.

Nomenclature

$\boldsymbol{G}_r$	= vector of the constraint functions, $g_{r,t}$ local to BB_r, $g_{r,t} \leq 0$ is a satisfied constraint
G_{yz}	= constraints in a black box (BB) that have a stronger dependence on $\boldsymbol{Y}$ and $\boldsymbol{Z}$ than on $\boldsymbol{X}$
$\boldsymbol{G}_0$	= vector of constraints active at the constrained minimum, length NG_0
I	= identity matrix
$\boldsymbol{L}$	= vector of the Lagrange multipliers corresponding to $\boldsymbol{G}_0$, length NG_0
NB	= number of BBs in the system
opt	= optimized quantity subscript
$\boldsymbol{P}$	= vector of parameters p_i kept constant in the process of finding the constrained minimum, length NP
T	= transposition superscript
$\boldsymbol{X}$	= vector of all concatenated $\boldsymbol{X}_r$, length NX
$\boldsymbol{X}_r$	= vector of the design variables $x_{r,j}$ local to BB_r, length NX_r
$\boldsymbol{XL}, \boldsymbol{XU}$	= lower and upper bounds on $\boldsymbol{X}$, side constraints
$\boldsymbol{Y}$	= vector of all concatenated $\boldsymbol{Y}_r$, length NY
$\boldsymbol{Y}_r$	= vector of behavior (state) variables output from BB_r, coupling variables, length NY_r
$\boldsymbol{y}_{r,i}$	= element of $\boldsymbol{Y}_r$; some are routed as inputs to other BBs, and may also be routed as output to the outside
$\boldsymbol{Y}_{r,s}$	= vector of variables input to BB_r from BB_s, coupling variables; by this definition $\boldsymbol{Y}_{r,s}$ is a subset of $\boldsymbol{Y}_r$, vector length $NY_{r,s}$
$\boldsymbol{y}_{r,s,i}$	= element of $\boldsymbol{Y}_{r,s}$
$\boldsymbol{Z}$	= vector of the design variables z_k that are shared by two or more BBs, system-level variables; length NZ
$\boldsymbol{ZL}, \boldsymbol{ZU}$	= lower and upper bounds on $\boldsymbol{Z}$, side constraints
0	= subscript denoting the present state from which to extrapolate, or the optimal state
Δ	= increment
$\Delta\boldsymbol{ZL}, \Delta\boldsymbol{ZU}$	= move limits
Φ	= the system objective function equated to one, particular $y_{1,i}$
ϕ_r	= local objective function in BB_r

I. Introduction

OPTIMIZATION of complicated engineering systems by decomposition is motivated by the obvious need to distribute the work over many people and computers to enable simultaneous, multidisciplinary optimization. It is important to partition the large undertaking into subtasks, each small enough to be easily understood and controlled by people responsible for it. This implies granting people in charge of a subtask a measure of authority and autonomy in the subtask execution, and allowing human intervention in the entire optimization process.

Reconciliation of the need for subtask autonomy with the system level challenge of everything influences everything else is difficult. Each of the leading multidisciplinary design optimization (MDO) methods that have evolved to date[1,2] tries to address that difficulty in a different way. In the system optimization based on the global sensitivity equations (GSE),[3–5] the partitioning applies only in the sensitivity analysis whereas optimization involves all of the design variables simultaneously. The concurrent subspace optimization method provides for separate optimizations within the modules,[6–10] but handles all the design variables simultaneously in the coordination problem. The collaborative optimization method[11,12] also enables separate optimizations within the modules, each performed to minimize a difference between the state and design variables and their target values set in a coordination problem. This problem combines the system optimization with the system analysis, therefore its dimensionality may be quite large.

Received 26 August 1998; presented as Paper 98-4916 at the AIAA/USAF/NASA/ISSMO 7th Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, 2–4 September 1998; revision received 30 June 1999; accepted for publication 15 July 1999.

*Manager of Computational Aerosciences and Multidisciplinary Research Coordinator. Fellow AIAA.

†Graduate Research Scholar Assistant, Joint Institute for Advancement of Flight Sciences; also Lieutenant, U.S. Air Force.

‡Professor, Joint Institute for Advancement of Flight Sciences. Fellow AIAA.

Most of the aforementioned method implementations had to overcome difficulties with integration of dissimilar codes. This has stimulated use of neural nets and response surfaces as means by which subdomains in the design space may be explored off-line and still be represented to the entire system. Unfortunately, effectiveness of this approach is limited to approximately 12–20 independent variables; hence, it is best suited for the early design phase. Consequently, a clear need remains for a method applicable in later design phases when the number of design variables is much larger. Methods that build a path in design space fit that requirement. Ultimately, one needs both domain-exploring methods and path-building methods, enhanced with seamless gear shifting between the two.

Motivated by this state of affairs, bilevel integrated system synthesis (BLISS) attacks the problem by performing an explicit system behavior and sensitivity analysis using the GSE, autonomous optimizations within the subsystems performed to minimize each module contribution to the system objective under the local constraints, and a coordination problem that engages only a relatively small number of the design variables that are shared by the modules. Solution of the coordination problem is guided by the derivatives of the behavior and local design variables with respect to the shared design variables. These derivatives may be computed in two different ways, giving rise to two versions of BLISS.

In either version, BLISS builds a gradient-guided path, alternating between the set of disjointed, modular design subspaces and the common system-level design space. Each segment of that path results in an improved design so that if one starts from a feasible state, the feasibility in each modular design subspace is preserved while the system objective is reduced. In case of an infeasible start, the constraint violations are reduced while the increase of the objective is minimized. Because the system analysis is performed at the outset of each segment of the path, the process can be terminated at any time, if the budget and time limitations so require, with the useful information validated by the last system analysis. In addition to enabling complete human control in the subspace optimization, BLISS allows the engineering team to exercise judgment, at any point in the procedure, to intervene before committing to the next successive pass.

BLISS has been developed in a prototype form and has been successfully demonstrated on the small-scale test cases reported herein.

II. Glossary of Terms

A compilation of special notations follows. A module in the mathematical model of a system is a black box (BB). BBA($\boldsymbol{Y}_r$, ($\boldsymbol{Z}$, $\boldsymbol{X}_r$)) is the analysis of BB$_r$ used to compute $\boldsymbol{Y}_r$ for given $\boldsymbol{Z}$ and $\boldsymbol{X}_r$. BBOF$_r$ is the BB objective function computed in BB$_r$ BBOPT$_r$ ($\boldsymbol{X}_r$, ϕ_r, $\boldsymbol{G}_r$), that is, optimization in BB$_r$ defined by Eq. (10). BBOSA$_r$($\boldsymbol{X}_{r,\text{opt}}$, $\boldsymbol{Z}$, $\boldsymbol{Y}_{r,s}$) is the analysis of BB$_r$ optimum for sensitivity to parameters. BBSA($\partial \boldsymbol{Y}_r/\partial(\boldsymbol{Z}, \boldsymbol{X}_r, \boldsymbol{Y}_{r,s})$) is the sensitivity analysis of BB$_r$ to compute its output derivatives with respect to $\boldsymbol{Z}$, $\boldsymbol{X}_r$, and $\boldsymbol{Y}_{r,s}$.

SA(($\boldsymbol{P}$, $\boldsymbol{Z}$, $\boldsymbol{X}$), $\boldsymbol{Y}$) is system analysis (SA) that is a computation that outputs $\boldsymbol{Y}$ for a system defined by $\boldsymbol{P}$, $\boldsymbol{Z}$, and $\boldsymbol{X}$. The system objective function (SOF) is computed in one of the BBs. SOPT($\boldsymbol{Z}$, Φ) is the system objective optimization (SOPT) defined by Eq. (13). SSA($\partial \boldsymbol{Y}/\partial \boldsymbol{Z}$, and $\partial \boldsymbol{Y}/\partial \boldsymbol{X}$) is the system sensitivity analysis (SSA) to compute sensitivity of the system response $\boldsymbol{Y}$ with respect to $\boldsymbol{Z}$ and $\boldsymbol{X}$.

The algorithm of Sec. III will introduce the following terms for the specific example of an aircraft. BB1 is performance analysis, BB2 is aerodynamics, and BB3 structures. In the algorithm, Φ is maximum range for given mission characteristics. $\boldsymbol{Y}_{1,2}$ includes the aerodynamic drag; $\boldsymbol{Y}_{1,3}$ includes the structural weight; $\boldsymbol{Y}_{2,1}$ includes Mach number; $\boldsymbol{Y}_{3,1}$ includes takeoff gross weight (TOGW); $\boldsymbol{Y}_{2,3}$ includes the structural deformations that alter the aerodynamic shape, and $\boldsymbol{Y}_{3,2}$ includes the aerodynamic loads.

In the algorithm $g_{1,t}$ is a noise abatement constraint on the mission profile, $g_{2,t}$ the limit of the chordwise pressure gradient, and $g_{3,t}$ the allowable stress; $x_{1,j}$ is cruise altitude, $x_{2,j}$ the leading-edge radius, $x_{3,j}$ the sheet metal thickness in the wing skin panel 138; and z_1 is wing sweep angle, z_2 wing aspect ratio, z_3 wing airfoil maximum depth-to-chord ratio, and z_4 location of the engine on the wing.

III. Algorithm

The algorithm is introduced using an example of a generic system of three BBs, as shown in Fig. 1. Three is a number small enough for easy conceptual grasp and compact mathematics, yet large enough to unfold patterns that readily generalize to larger *NB*. Even though the system in Fig. 1 is generic, it may be useful to bear a specific example in mind.

The system in Fig. 1 is characterized by BB-level design variables $\boldsymbol{X}$, and by system-level design variables $\boldsymbol{Z}$. As a reference, if an all in one optimization were performed, observing the system at a single level and making no distinction between the treatment of $\boldsymbol{X}$ variables and the treatment of $\boldsymbol{Z}$ variables, the problem could be stated as follows:

$$\begin{aligned} &\text{given } \boldsymbol{X} \text{ and } \boldsymbol{Z}, && \text{find } \Delta\boldsymbol{X} \text{ and } \Delta\boldsymbol{Z} \\ &\text{minimize } \Phi(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}(\boldsymbol{X}, \boldsymbol{Z})), && \text{satisfy } \boldsymbol{G}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}(\boldsymbol{X}, \boldsymbol{Z})) \end{aligned} \quad (1)$$

Because BLISS approaches this optimization by means of a system decomposition, the algorithm depends on the availability of the derivatives of output with respect to input for each BB. That assumes the differentiability of the BB internal relationships to at least the first order. It is immaterial how the derivatives are computed, finite differencing may always be used, but it is expected that in most cases one will utilize one of the more efficient analytical techniques.[13]

The algorithm comprises the SA and sensitivity analysis, local optimizations inside of the BBs (that include the BB internal analyses), and the system optimization. We will not elaborate on SA beyond pointing out that it is highly problem dependent, and likely to be iterative if there are any nonlinearities in the BB analyses. Each pass through the BLISS procedure improves the design in two steps: first by concurrent optimizations of the BBs using the design variables $\boldsymbol{X}$ and holding $\boldsymbol{Z}$ constant, next by means of a system-level optimization that utilizes variables $\boldsymbol{Z}$. We begin with the BB-level optimization.

A. BB-Level (Discipline or Subsystem) Optimizations

The basis of the algorithm is the formulation of an objective function unique for each BB such that constrained minimization of that function in each BB results in the minimization of the system objective function (SOF). The SOF is computed as a single output item in one of the BBs; without loss of generality we assume that it is the first BB, BB$_1$, so that

$$\Phi = y_{1,i} \quad (2)$$

is one of the elements of the behavior variable output vector $\boldsymbol{Y}_1$.

The derivatives of $\boldsymbol{Y}$ with respect to $x_{r,j}$, $\partial \boldsymbol{Y}/\partial x_{r,j}$, are computed according to Sobieszczanski-Sobieski[3] by solving a set of simultaneous, algebraic equations GSE, for a particular $x_{r,j}$:

$$[A]\left\{\frac{\partial \boldsymbol{Y}}{\partial x_{r,j}}\right\} = \left\{\frac{\partial f}{\partial x_{r,j}}\right\} \quad (3)$$

where the right-hand side term is a partial derivative of the functional relationship $\boldsymbol{Y}_r = f(x_{r,j}, \boldsymbol{Y}_{r,s}(x_{r,j}))$, and A is a square matrix whose dimension equals the length of the $\boldsymbol{Y}$ vector ($NY \times NY$). The matrix

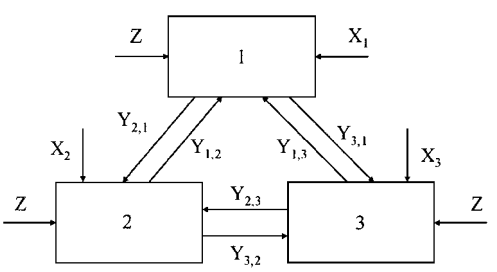

Fig. 1 System of coupled BB.

A is composed of submatrices forming the pattern

$$\left[\begin{array}{c|c|c} I & A_{1,2} & A_{1,3} \\ \hline A_{2,1} & I & A_{2,3} \\ \hline A_{3,1} & A_{3,2} & I \end{array}\right] \tag{4}$$

where I is identity matrix, $NY_r \times NY_r$, and $A_{r,s}$ are matrices of the derivatives that capture sensitivity of output to input in a BB. For example,

$$A_{2,3} = -\frac{\partial \boldsymbol{Y}_2}{\partial \boldsymbol{Y}_3}, NY_2 \times NY_3, \qquad A_{3,2} = -\frac{\partial \boldsymbol{Y}_3}{\partial \boldsymbol{Y}_2}, NY_3 \times NY_2 \tag{5}$$

The derivative terms such as $\partial f/\partial x_{r,j}$, and $\partial \boldsymbol{Y}_2/\partial \boldsymbol{Y}_3$ in Eqs. (3) and (5) may be obtained in a BB sensitivity analysis (BBSA).[13] Note that Eq. (3) can be efficiently solved for many different $x_{r,j}$ using techniques available for linear equations with many right-hand sides.

Having $\partial \boldsymbol{Y}/\partial x_{r,j}$ computed from Eq. (3) for all $x_{r,j}$, we can express Φ as a function of $\boldsymbol{X}$ by the linear part of the Taylor series:

$$\Phi = y_{1,i} = (y_{1,i})_0 + \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{X}_1}\right)^T \Delta \boldsymbol{X}_1 + \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{X}_2}\right)^T \Delta \boldsymbol{X}_2 + \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{X}_3}\right)^T \Delta \boldsymbol{X}_3 \tag{6}$$

where the derivative terms are vectors of length NX_r. We see from Eq. (6) that

$$\Delta\Phi = \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{X}_1}\right)^T \Delta \boldsymbol{X}_1 + \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{X}_2}\right)^T \Delta \boldsymbol{X}_2 + \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{X}_3}\right)^T \Delta \boldsymbol{X}_3 \tag{7}$$

the three terms showing explicitly how the local design variables from each of the three BBs contribute to a change in the system objective function $\Delta\Phi$.

It is apparent that to minimize $\Delta\Phi$ we need to charge each BB with the task of minimizing its own objective. Using BB_2 as an example, objective ϕ_2 is

$$\phi_2 = \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{X}_2}\right)^T \Delta \boldsymbol{X}_{2,j}, \qquad j = 1 \longrightarrow NX_2 \tag{8}$$

The preceding equations state mathematically the fundamentally important concept that in a system optimization the contributing disciplines should not optimize themselves for a traditional, discipline-specific objective such as the minimum aerodynamic drag or minimum structural weight. They should optimize themselves for a synthetic objective function that measures the influence of the BB_r design variables $\boldsymbol{X}_r$ on the entire system objective function.

Another way to look at it is to observe that

$$\phi_2 = \left(\frac{\partial y_{1,i}}{\partial x_{2,1}}\right)^T \Delta x_{2,1} + \left(\frac{\partial y_{1,i}}{\partial x_{2,2}}\right)^T \Delta x_{2,2} + \left(\frac{\partial y_{1,i}}{\partial x_{2,j}}\right)^T \Delta x_{2,j} + \cdots, \qquad j = 1 \longrightarrow NX_2 \tag{9}$$

and so it may be regarded as a composite objective function commonly used in multiobjective optimization. One may say, therefore, that in a coupled system the local disciplinary or subsystem optimizations should be multiobjective with a composite objective function. The composite objective should be a sum of the local design variables weighted by their influence on the single objective of the whole system. It should be emphasized that this is true also in that particular BB_r where Φ is being computed. In the aircraft example it is $\Phi = y_{1,i}$ in BB_1 according to Eq. (2). However, the BB_1 optimization objective is not $\phi_1 = y_{1,i}$. Instead, it is ϕ_1 from an equation analogous to Eq. (9).

The local optimization problem may be stated formally for BB_2:

$$\text{given } \boldsymbol{X}_2, \boldsymbol{Z}, \text{ and } \boldsymbol{Y}_{2,1}, \boldsymbol{Y}_{2,3}, \qquad \text{find } \Delta \boldsymbol{X}_2; \text{ length } NX_2$$

$$\text{minimize } \phi_2 = \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{X}_2}\right)^T \Delta \boldsymbol{X}_2$$

$$\text{satisfy } G_2 \leq 0, \qquad \text{including side constraints} \tag{10}$$

Incidentally, we adhere to the convention that calls for minimization of the objective function. If the application requires that function be maximized, as it does in the example of aircraft range, we convert the objective, for example, $\Phi = -(\text{range})$.

The optimization problems for BB_1 and BB_3 are analogous. All three problems, being independent of each other, may be solved concurrently. This is an opportunity for concurrent engineering and parallel processing. By solving Eq. (10) for all three BBs, we have improved the system because, according to Eqs. (6) and (7), we have reduced Φ by $\Delta\Phi$ while satisfying constraints in each BB.

B. System-Level Optimization

So far we have improved the system by manipulating local $\boldsymbol{X}$ in the presence of a constant $\boldsymbol{Z}$. We can score further improvement by exploiting $\boldsymbol{Z}$ as variables. To do so we need to know how $\boldsymbol{Z}$ influences $\Phi = y_{1,i}$. That is, we need $\partial y_{1,i}/\partial \boldsymbol{Z}$.

At this point, the BLISS algorithm forks into two alternatives: BLISS/*A* and BLISS/*B*.

1. *BLISS/A*

This version of BLISS computes the derivatives of $\boldsymbol{Y}$ with respect to $\boldsymbol{Z}$ by a new, generalized version of GSE, Eq. (3). The GSE generalization accounts for the optimization of a BB turning its $\boldsymbol{X}$ into a function of $\boldsymbol{Y}$ and $\boldsymbol{Z}$ that enter that particular BB as parameters. The generalization takes the following form:

$$[M]\left\{\begin{array}{c} \dfrac{d\boldsymbol{Y}}{dz_k} \\ \dfrac{d\boldsymbol{X}}{dz_k} \end{array}\right\} = \left\{\begin{array}{c} \dfrac{\partial f_y}{\partial z_k} \\ \dfrac{\partial f_x}{\partial z_k} \end{array}\right\} \tag{11}$$

termed GSE/optimized subsystems (GSE/OS), where the right-hand-side terms are partial derivatives of the functional relationships $\boldsymbol{Y}_r = f_y(z_k, \boldsymbol{Y}_{r,s}(z_k))$, and $\boldsymbol{X}_r = f_x(z_k, \boldsymbol{Y}_{r,s}(z_k))$. Recall that the r subscript indicates association with BB_r whereas the s subscript indicates information flow from BB_s. The GSE/OS yields a vector $d\boldsymbol{Y}/dz_k$ and $d\boldsymbol{X}/dz_k$, and because Φ is one of the elements of $\boldsymbol{Y}$, $\Phi = y_{1,i}$, we get the desired derivative $d\Phi/dz_k$. Derivation and details of the GSE/OS structure, including the definition of the matrix M, are in the Appendix. The matrix of coefficients in GSE/OS is populated with $\partial \boldsymbol{Y}_r/\partial \boldsymbol{Y}_s$, $\partial \boldsymbol{Y}_r/\partial \boldsymbol{X}_r$, and $\partial \boldsymbol{X}_r/\partial \boldsymbol{Y}_s$. These terms and the right-hand-side terms of $\partial \boldsymbol{Y}/\partial z_k$ and $\partial \boldsymbol{X}/\partial z_k$ are obtained from the following sources: $\partial \boldsymbol{Y}_r/\partial \boldsymbol{Y}_s$, $\partial \boldsymbol{Y}_r/\partial \boldsymbol{X}_r$, and $\partial \boldsymbol{Y}/\partial z_k$ from BBSA; and $\partial \boldsymbol{X}_r/\partial z_k$, and $\partial \boldsymbol{X}_r/\partial \boldsymbol{Y}_s$ from BB optimum sensitivity analysis, BBOSA.

The terms $\partial \boldsymbol{X}_r/\partial z_k$ and $\partial \boldsymbol{X}_r/\partial \boldsymbol{Y}_s$ are the derivatives of optimum with respect to parameters that, in principle, may be obtained by differentiation of the Kuhn–Tucker conditions, for example, an algorithm described by Sobieszczanski-Sobieski et al.[14] That approach, however, requires second-order derivatives of behavior, too costly in most large-scale applications. Therefore, BLISS/*A* uses an approximate algorithm adapted from Vanderplaats and Cai.[15] In that algorithm, parameters are perturbed by a small increment, one at a time, and the BB optimization is repeated by linear programming (LP) starting from the optimal point. Derivatives of optimal $\boldsymbol{X}$ and $\boldsymbol{Y}$ with respect to parameters are then computed by finite differences.

2. *BLISS/B*

This version of BLISS avoids calculation of $\partial \boldsymbol{X}_r/\partial z_k$ and $\partial \boldsymbol{X}_r/\partial \boldsymbol{Y}_s$ altogether by using an algorithm that yields $\partial\Phi/\partial \boldsymbol{P}$, where $\boldsymbol{P}$ includes both $\boldsymbol{Y}$ and $\boldsymbol{Z}$. The algorithm, described in the literature (e.g., Ref. 16) is based on the well-known notion that the Lagrange multipliers may be interpreted as the prices, stated in the units of Φ, for the constraint changes caused by incrementing p_i. For a general case of the objective $F = F(\boldsymbol{P})$ and $\boldsymbol{G}_0 = \boldsymbol{G}_0(\boldsymbol{P})$, the algorithm gives the following formula for $(dF/dp_i)_0$:

$$\left(\frac{dF}{dp_i}\right)_0 = \frac{\partial F}{\partial p_i} + L^T\left(\frac{\partial \boldsymbol{G}_0,}{\partial p_i}\right)$$

To use the preceding equation in BLISS, consider that in $\boldsymbol{P}$ we have an independent $\boldsymbol{Z}$ and $\boldsymbol{Y} = \boldsymbol{Y}(\boldsymbol{Z})$ so that the partial terms require chain differentiation. Hence, the preceding general formula tranforms to

$$\left(\frac{\mathrm{d}y_{1,i}}{\mathrm{d}z_k}\right)_0^T = \left(L^T\left(\frac{\partial \boldsymbol{G}_0}{\partial z_k}\right)\right)_1 + \left(L^T\left(\frac{\partial \boldsymbol{G}_0}{\partial z_k}\right)\right)_2 + \left(L^T\left(\frac{\partial \boldsymbol{G}_0}{\partial z_k}\right)\right)_3 + \left[\left(L^T\left(\frac{\partial \boldsymbol{G}_0}{\partial \boldsymbol{Y}}\right)\right)_1 + \left(L^T\left(\frac{\partial \boldsymbol{G}_0}{\partial \boldsymbol{Y}}\right)\right)_2 + \left(L^T\left(\frac{\partial \boldsymbol{G}_0}{\partial \boldsymbol{Y}}\right)\right)_3\right]\left(\frac{\partial \boldsymbol{Y}}{\partial z_k}\right) + \left(\frac{\partial y_{1,i}}{\partial z_k}\right)^T \tag{12}$$

where subscripts 1, 2, and 3 identify the BBs 1, 2, and 3. The terms in the given equation originate from the following sources: $\partial \boldsymbol{G}_0/\partial z_k$ and $\partial \boldsymbol{G}_0/\partial \boldsymbol{Y}$ from (BBSA) performed on isolated BB_r, $\boldsymbol{L}$ obtained for BB_r at the end of BB optimization (BBOPT), and $\partial \boldsymbol{Y}/\partial z_k$—from GSE in SSA.

BLISS/*B* is substantially simpler in implementation than BLISS/*A* and it eliminates the computational cost of one LP per parameter $y_{r,i}$ and z. Optimizers that yield $\boldsymbol{L}$ as a byproduct of optimization are available for use in BBOPT, or $\boldsymbol{L}$ may be obtained as described by Haftka and Gurdal.[17]

3. Optimization in Z Space

Once $\mathrm{d}y_{1,i}/\mathrm{d}z_k$ have been computed from either Eq. (11) or as $(\mathrm{d}y_{1,i}/\mathrm{d}z_k)_0$ from Eq. (12), we can further improve the system objective by executing the following optimization, using any suitable optimizer:

$$\text{given } \boldsymbol{Z} \text{ and } \Phi_0, \qquad \text{find } \Delta \boldsymbol{Z}$$

$$\text{minimize } \Phi = \Phi_0 + \left(\frac{\partial y_{1,i}}{\partial \boldsymbol{Z}}\right)^T \Delta \boldsymbol{Z}$$

$$\text{satisfy } \boldsymbol{Z}L \le \boldsymbol{Z} + \Delta \boldsymbol{Z} \le \boldsymbol{Z}U, \qquad \Delta \boldsymbol{Z}L \le \Delta \boldsymbol{Z} \le \Delta \boldsymbol{Z}U \tag{13}$$

where $(\partial y_{1,i}/\partial \boldsymbol{Z})$ is a vector that collects all the derivatives $\mathrm{d}y_{1,i}/\mathrm{d}z_k$, and Φ_0 is inherited from the previous cycle analysis (SA) for $\boldsymbol{X}$ and $\boldsymbol{Z}$ (initialized if it is the first cycle). It is recommended to handle the $\boldsymbol{Z}$ constraints by means of a trust-region technique, e.g., Ref. 18. In Eq. (13) $\partial y_{1,i}/\partial \boldsymbol{Z}$ is a constrained derivative that protects $\boldsymbol{G}_0 = 0$ in all BBs. Therefore, the optimization is unconstrained except of the side constraints and move limits.

However, some BBs may have constraints that depend on system and behavior variables, $\boldsymbol{Z}$ and $\boldsymbol{Y}$, more strongly than on $\boldsymbol{X}$ (in the extreme case some constraints may not be functions of $\boldsymbol{X}$ at all, only of $\boldsymbol{Y}$ and $\boldsymbol{Z}$). Such constraints, denoted G_{yz}, may be difficult (or impossible) to satisfy by manipulating only $\boldsymbol{X}$ in BBOPT. To satisfy them, one must add to the $\boldsymbol{Z}$-space optimization in Eq. (13) their extrapolated values

$$\boldsymbol{G}_{yz} = \boldsymbol{G}_{yz,0} + \left(\frac{\partial \boldsymbol{G}_{yz}}{\partial \boldsymbol{Z}} + \frac{\partial \boldsymbol{G}_{yz}}{\partial \boldsymbol{Y}}\frac{\partial \boldsymbol{Y}}{\partial \boldsymbol{Z}}\right)\Delta \boldsymbol{Z} \le 0 \tag{14}$$

where $\partial \boldsymbol{G}_{yz}/\partial \boldsymbol{Z}$, and $\partial \boldsymbol{G}_{yz}/\partial \boldsymbol{Y}$ are obtained from the BBSA. In this instance, the $\boldsymbol{Z}$-space optimization becomes a constrained one.

IV. Iterative Procedure

The two operations, the local optimizations in the BBs and the system-level optimization, described in Sec. III, result in a new system, altered because of the increments on $\boldsymbol{X}$ and $\boldsymbol{Z}$. This means that inputs to and outputs from the computations at both the system and BB levels [SA, BB analysis (BBA), BBSA, SSA, BBOPT, BBOSA in BLISS/*A*, and SOPT] all need to be updated and the sequence of these operations repeated with the new values of all quantities involved. This includes new values of all of the derivatives because they would change if there were any nonlinearities in the system (as there usually are).

In a large-scale application where execution of each BLISS cycle may require significant resources and time, the engineering team may wish to review the results before committing to the next cycle. That intervention may entail a problem reformulation, such as overriding the variable values, deleting and adding variables, constraints, or even BBs.

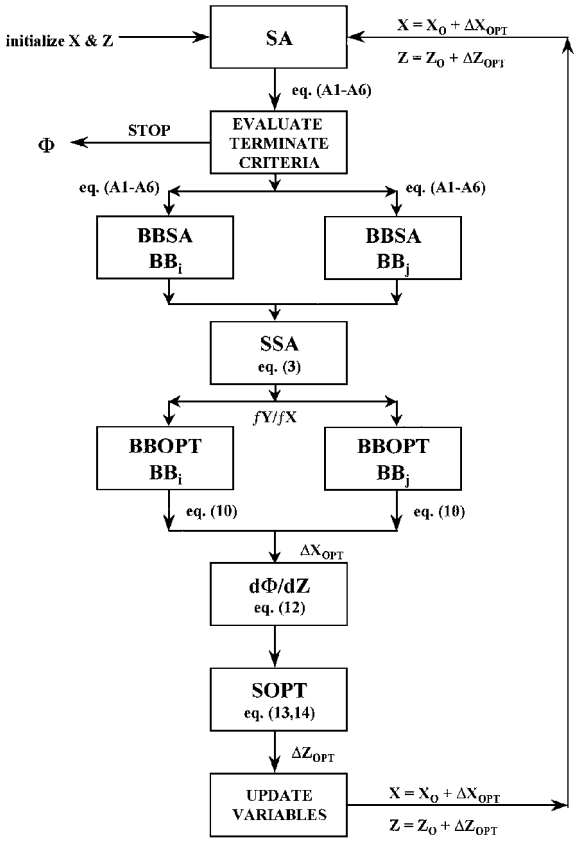

Fig. 2 BLISS/*B* flowchart.

Thus, the following procedure emerges (also flowchart Fig. 2) for BLISS/*B* with the BLISS/*A* operations, if different, noted in brackets.

0) Initialize $\boldsymbol{X}$ and $\boldsymbol{Z}$.

1) Use SA to get $\boldsymbol{Y}$ and $\boldsymbol{G}$; this includes BBA for all BBs.

2) Examine termination criteria, exercise judgment to override the results, modify the problem formulation, and continue or stop.

3) Use BBSA to obtain $\partial \boldsymbol{Y}/\partial \boldsymbol{X}$, $\partial \boldsymbol{Y}_{r,s}/\partial \boldsymbol{Y}_s$, $\partial \boldsymbol{G}/\partial \boldsymbol{Z}$, and $\partial \boldsymbol{G}/\partial \boldsymbol{Y}$, and SSA [Eq. (3)], to get $\partial \boldsymbol{Y}/\partial \boldsymbol{X}$ (and $\partial \boldsymbol{Y}/\partial \boldsymbol{Z}$). Here is an opportunity for concurrent processing.

4) Use BBOPT for all BBs [Eq. (10)] using ϕ formulated individually for each BB [Eq. (8)], get ϕ_{opt} and $\Delta \boldsymbol{X}_{\mathrm{opt}}$; obtain Lagrange multipliers $\boldsymbol{L}$ for $\boldsymbol{G}_0$ [skip $\boldsymbol{L}$]. Here is an opportunity for concurrent processing.

5) Obtain $\mathrm{d}\Phi/\mathrm{d}\boldsymbol{Z}$ as in Eq. (12). [Execute BBOSA to obtain $\partial \boldsymbol{X}/\partial \boldsymbol{Z}$ and $\partial \boldsymbol{X}/\partial \boldsymbol{Y}$, and form and solve GSE/OS[15] to generate $\partial \boldsymbol{Y}/\partial \boldsymbol{Z}$]. Here is an opportunity for concurrent processing.

6) Use SOPT to get $\boldsymbol{Z}_{\mathrm{opt}}$ by Eqs. (13) and (14).

7) Update all quantities, and repeat from step 1. $\boldsymbol{X} = \boldsymbol{X}_0 + \Delta \boldsymbol{X}_{\mathrm{opt}}$; $\boldsymbol{Z} = \boldsymbol{Z} + \Delta \boldsymbol{Z}_{\mathrm{opt}}$.

Note that termination is placed as step 2 after SA to ensure that the full analysis results document the final system design, as opposed to having it documented only by the extrapolated quantities. Also, at this point the engineering team may decide whether to intervene by modifying the variable values, and adding or deleting the design variables and constraints.

When started from a feasible design, the procedure will result in an improved system, while the local constraints are kept satisfied within extrapolation accuracy, even when terminated before convergence.

In case of an infeasible design start, the improvement will be in the sense of reductions in the constraint violations, while the objective may exhibit an increase, at least initially. The procedure achieves the improvement by virtue of optimization alternating between the domain of *NB* $\boldsymbol{X}$ spaces (step 4), and the single $\boldsymbol{Z}$ space (step 6).

There is a caveat: Because in BLISS/*B* the extrapolation of Φ in Eq. (13) is based on the use of the Lagrange multipliers in Eq. (12), its accuracy depends on the BBOPT yielding a feasible solution and on the active constraints $\boldsymbol{G}_0$ remaining active for updated $\boldsymbol{Z}$, for

example, see Barthelemy and Sobieszczanski-Sobieski.[19] If some constraints leave the active set $\boldsymbol{G}_0$, or new constraints enter, a discontinuous change of the extrapolation error may result. For example, consider the wing aspect ratio (AR) as a $\boldsymbol{Z}$ variable and suppose that for AR = 3 it is the stress due to the wing bending that is one of the active constraints in the structures BB. If optimization in the $\boldsymbol{Z}$ space took the design to AR = 4, the next cycle may reveal that the stress constraint is satisfied but a flutter constraint becomes critical. Past experience[20] shows that this discontinuity may slow, but not prevent, the process convergence and the latter may be controlled by adjusting the move limits and by the user's intervention in the process. Numerical experiments cited in Ref. 21 showed no need for the optimization termination criteria to be tighter than normal for the extrapolation based on the optimum sensitivity derivatives to have a useful accuracy.

V. Numerical Tests and Examples

BLISS/*A* was tested on a sample of test problems from Ref. 22 and on a design of an electronic package. BLISS/*B* was exercised on the latter and also on a very simplified aircraft configuration problem. Both versions of BLISS performed as intended in all of the tests. The sole purpose of these initial numerical experiments was to test and to demonstrate the BLISS procedure logic and data flow, therefore, the BBs were merely surrogates of the numerical processes that need to be used in real applications.

A. Aircraft Optimization

The aircraft test was an optimum cruise segment of a supersonic business jet based on the 1995–1996 AIAA Student Competition. This problem was selected because of its available data base and the availability of the black boxes written in VISUAL BASIC in the form of Excel™ spreadsheets. The supersonic business jet was modeled as a coupled system of structures (BB_1), aerodynamics (BB_2), propulsion (BB_3), and aircraft range (BB_4). All of the disciplines were represented by modules comprising an analysis level typical for an early conceptual design stage. For example, wing weight was computed by a statistical equation based on complex regression analysis of applicable historical data. Reference 23 presents this equation as

$$W_{\text{wing}} = 0.0051(W_T N_Z)^{0.557} S_{\text{REF}}^{0.649} \text{AR}^{0.5} (t/c)^{-0.4} (1+\lambda)^{0.1} \times (0.1875 S_{\text{REF}})^{0.1} / \cos\Lambda \tag{15}$$

Additionally, to ensure continuity in the optimization space, equations for specific fuel consumption and an upper constraint bound on throttle setting were surface fits representing engine deck data.

The aircraft optimization was a maximization of the range computed through the Breguet range equation. For testing purposes, additional design and state variables were introduced in BBs 1–3, and functional relationships not present in the original BBs were supplied to reflect what is commonly known about the typical functions involved in design. For example, stress is expected to fall as a reciprocal of the increase of the skin thickness in a wing box. Such relationships were represented by polynomial functions. One plot of such a function is shown in Fig. 3, portraying the wing twist as a function of the wing box cross-sectional dimensions scale factor and the wing lift. Each polynomial function is of the form

$$PF = A_0 + A_i \times \boldsymbol{S}^T + \tfrac{1}{2} \times \boldsymbol{S} \times A_{ij} \times \boldsymbol{S}^T \tag{16}$$

where $\boldsymbol{S}$ is the vector of independent variables and A_0, A_i, and A_{ij} are coefficient terms.

It should be noted that BB_2 contains a constraint that does not depend on its $\boldsymbol{X}$ or $\boldsymbol{Y}$ input, thus the $\boldsymbol{Z}$-space optimization is a constrained one, per Eqs. (13) and (14). Side constraints on $\boldsymbol{Z}$ were judiciously selected to guard against conditions not accounted for in the BBAs. For example, a lower bound of 2.5 on AR stemmed from the subsonic performance considerations. The BBs are coupled by the output-to-input data transfers (design structure matrix) depicted in Fig. 4. Note that BB_4 is an analysis-only module and does not feedback any data to other BBs.

Table 1 Aircraft results for 20% move limit

Variable	Cycle[a] 1	2	3	4	5
			Range (SSA)		
	535.79	1581.67	3425.35	3961.41	3963.98
Extpl. error	−535.79	−536.67	−431.63	−56.26	−3.43
BB1 extpl.	17.17	−0.16	−3.26	−0.86	0.00
BB2 exptl.	16.85	0.00	0.00	0.00	0.00
BB3 extpl.	26.00	110.92	−76.84	0.00	0.00
X extpl.	60.02	110.75	−80.10	−0.86	0.00
Z extpl.	449.19	1301.30	559.90	0.00	0.00
			Range (Extpl.)		
	1045.00	2993.72	3905.15	3960.55	3963.98
λ	0.25	0.14951	0.17476	0.25775	0.38757
x	1	0.75	0.75	0.75	0.75
C_f	1	0.75	0.75	0.75	0.75
T	0.5	0.1676	0.20703	0.15624	0.15624
t/c	0.05	0.06	0.06	0.06	0.06
h, ft	45000	54000	60000	60000	60000
M	1.6	1.4	1.4	1.4	1.4
AR	5.5	4.4	3.3	2.5	2.5
Λ, deg	55	66	70	70	70
S_{ref}, ft^2	1000	1200	1400	1500	1500

[a] One cycle is one pass through the BLISS procedure.

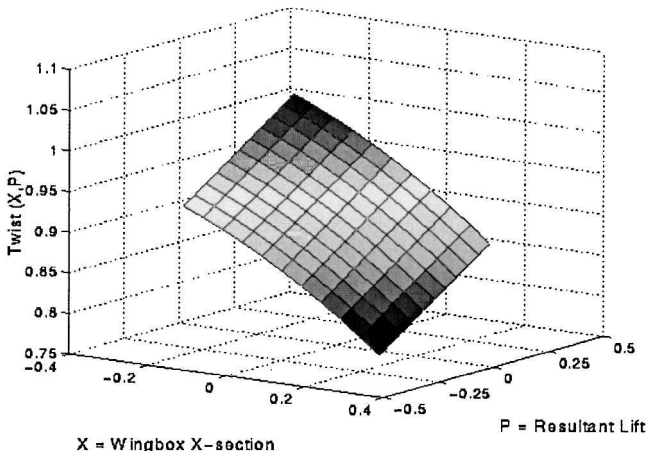

Fig. 3 Polynomial representation of wing twist.

This test was conducted entirely using MATLAB® 5 and its Optimization Toolbox.[24] To establish a benchmark, the system was first optimized using an all-in-one approach in which the MATLAB optimizer was coupled directly to SA and saw no distinction between the $\boldsymbol{X}$ and $\boldsymbol{Z}$ variables. Next, the test case was executed using BLISS/*B*, starting at different infeasible initial points chosen by varying the six design variables that are not arguments in the polynomial functions. The choice of initial values for variables that are arguments of the polynomial functions was limited due to the nature of the polynomial formulation. This limitation is not a characteristic of the BLISS method itself, as the polynomial functions would not be required in a large-scale optimization problem. With the move limits ranging from 10 to 70%, the procedure convergence was satisfactory through the move limits of 60% for all initial points tested. However, in nearly all cases, no additional improvement in convergence rate was recorded for move limits greater than 20%. For instance, the objective function was advanced to within 1% of the benchmark in five passes for move limits 20 and 30%. Onset of an erratic behavior was observed with move limits increased past 60%, the procedure converged or diverged dependent on the starting point.

Table 1 displays a sample of typical results for the move limits value of 20%. It shows that the initial design range was extremely poor, only 536 nm. BLISS/*B* improvements advanced the range to 3964 nm. The range converged monotonically, although in some cases small amplitude oscillations were observed. Comparison of the extrapolated and actual values of the objective and constraints shows reasonable accuracy and conservatism of the extrapolations.

Table 2 Normalized *Y* derivatives with respect to *X* and *Z*[a]

	Density									
Number	λ	x	C_f	T	t/c	h	M	AR	Sweep	S_{ref}
W_T	0.01146	1.71536	0.01981	−0.15744	0.12714	−0.33931	0.31958	0.08208	0.2537	0.55182
W_F	0	0	0	0	0.72626	0	0	−0.36043	0	1.09211
Θ	−0.03342	0.19971	$3.31E-15$	$-1.73E-14$	$-2.10E-14$	$-1.93E-13$	$-6.15E-14$	−0.10766	$3.77E-14$	−0.10766
L	0.01146	1.71536	0.01981	−0.15744	0.12714	−0.33931	0.31958	0.08208	0.2537	0.55182
D	$-4.19E-05$	0.00581	0.12457	−0.00049	0.68108	−2.1339	2.00984	$3.37E-06$	−0.83983	0.99838
L/D	0.0115	1.7095	−0.1046	−0.15694	−0.54935	1.84108	−1.6507	0.08207	1.10064	−0.43675
SFC	$1.98E-20$	$-5.07E-18$	$-2.70E-17$	0.08544	0	0.12946	0.05555	$2.31E-17$	$-1.86E-16$	0
W_E	$-4.40E-05$	0.0061	0.13083	−1.03986	0.71531	−2.24115	2.11086	$3.54E-06$	−0.88204	1.04857
ESF	$-4.19E-05$	0.00581	0.12457	−0.99059	0.68108	−2.1339	2.00984	$3.37E-06$	−0.83983	0.99838
R	−0.00077	−0.12692	−0.12581	−0.07299	0.10115	2.07616	−1.04784	−0.39618	0.82904	0.15535

[a]SFC, specific fuel consumption; ESF, engine scale factor.

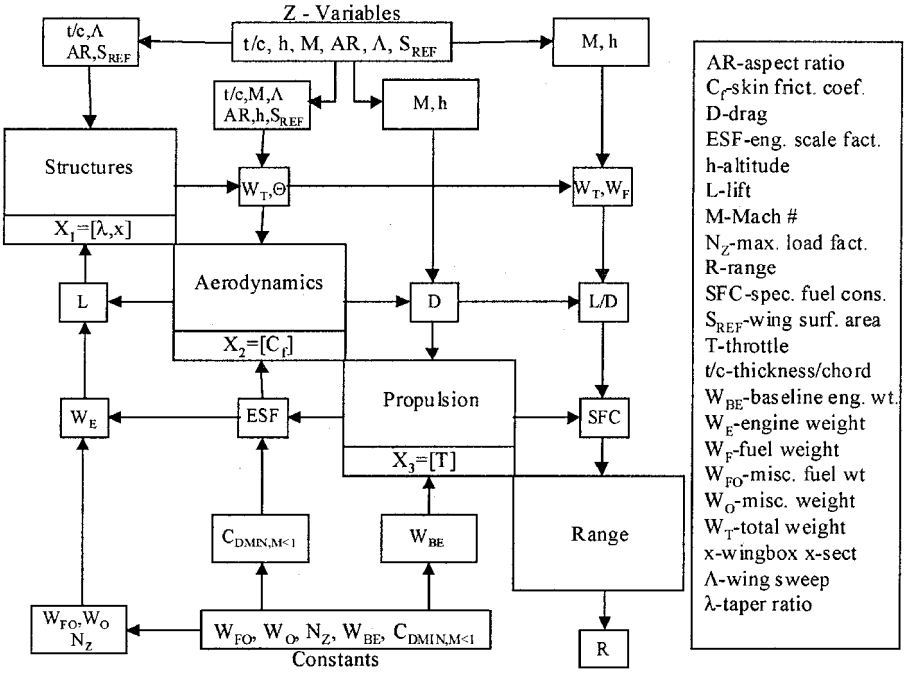

Fig. 4 Data dependencies for range optimization.

The optimal values of the design variables reflect numerous tradeoffs typical for aircraft design. For instance, optimal thickness to chord ratio (t/c) resulted, in part, from a tradeoff between the wave drag and structural weight. Table 2 shows normalized (logarithmic) derivatives of all $\boldsymbol{Y}$, including the range, with respect to all of the $\boldsymbol{X}$ and $\boldsymbol{Z}$ variables, sampled in cycle 1 to illustrate sensitivity of the system solution to design variables.

Figure 5 illustrates the range histogram, and depicts the extrapolation error as being effectively controlled by the move limits. Range sensitivities to $\boldsymbol{X}$ and $\boldsymbol{Z}$ variables are shown in Fig. 6. As expected, altitude and Mach number have the largest effect on the objective function, whereas taper ratio has the smallest. Figure 7 shows the individual BB and system contributions to the range objective in each cycle. Here it is observed that, in this particular case, the contribution of system level variables is significantly larger than that of the local variables in the extrapolation of range.

This test case was also implemented in a software package for system analysis and optimization called iSIGHT.[25] The iSIGHT and MATLAB results cross check was completely satisfactory.

B. Electronic Package Optimization

The electronic packaging was introduced as an MDO problem by Renaud (see Ref. 26). Its electrical and thermal subsystems are coupled because component resistance is influenced by operating temperatures and the temperatures depend on resistance.

The objective of the problem is to maximize the watt density for the electronic package subject to constraints. The constraints require the operation temperatures for the resistors to be below a threshold temperature and the current through the two resistors to be equal. The system diagram in Fig. 8 shows the data dependencies for two BBs, representing electrical resistance analysis and thermal analysis. As Fig. 8 indicates, there are no natural $\boldsymbol{Z}$ in this case. Therefore, $\boldsymbol{Z}$ were created as targets imposed on each of the $\boldsymbol{Y}$ and the BBOPTs were required to match the $\boldsymbol{Y}$ values to those $\boldsymbol{Z}$ targets (similar to how it is done in the collaborative optimization method). Details of the electronic packaging problem are given by Padula et al.[27§]

This test case was implemented in iSIGHT using BLISS/*A* and *B*. A benchmark result was obtained by executing an all-in-one (A-in-O column) optimization from various starting points.

BLISS/*A* and *B* were started from the same points. Table 3 displays the benchmark and the BLISS/*A* and *B* results as showing a good agreement. Table 3 also indicates a comparison of the computational labor (the Work column) measured by the number of BB evaluations necessary to converge the fixed-point iterations in the BBAs and in the SA, all repeated as needed to compute derivatives by finite differences in a gradient-guided optimization. As Table 3 shows, the BLISS/*B* computational labor was substantially lower than the benchmark in all cases.

VI. BLISS Status, Assessment, and Concluding Remarks

A method for engineering system optimization was developed to decompose the problem into a set of local optimizations (large number of detailed design variables) and a system-level optimization (small number of global design variables). Optimum sensitivity

§Accessible at http://fmod-w.w.w.larc.nasa.gov/mdob/MDOB/index.html.

Table 3 Electronic packaging data

Case	Initial design objective	Initial design max constraint violation	Final design objective	Final design max constraint violation	Work
			A-in-O		
1	7.79440*E*+01	2.16630*E*−08	6.39720*E*+05	1.22*E*−03	498
2	6.83630*E*+03	−2.89560*E*−01	6.39720*E*+05	1.22*E*−03	264
3	1.51110*E*+03	−4.29240*E*−02	6.36540*E*+05	1.45*E*−03	264
4	1.46070*E*+01	−1.02490*E*−03	6.36940*E*+05	1.42*E*−03	175
			BLISS/A		
1	7.79440*E*+01	2.16630*E*−08	6.39700*E*+05	1.20*E*−03	436
2	6.83630*E*+03	−2.89560*E*−01	6.39050*E*+05	1.18*E*−03	508
3	1.51110*E*+03	−4.29240*E*−02	6.39050*E*+05	−4.89*E*−04	174
4	1.46070*E*+01	−1.02490*E*−03	6.39290*E*+05	3.70*E*−04	313
			BLISS/B		
1	7.79440*E*+01	2.16630*E*−08	6.39720*E*+05	1.22*E*−03	365
2	6.83630*E*+03	−2.89560*E*−01	6.39720*E*+05	1.22*E*−03	207
3	1.51110*E*+03	−4.29240*E*−02	6.39720*E*+05	1.22*E*−03	114
4	1.46070*E*+01	−1.02490*E*−03	6.39720*E*+05	1.22*E*−03	105

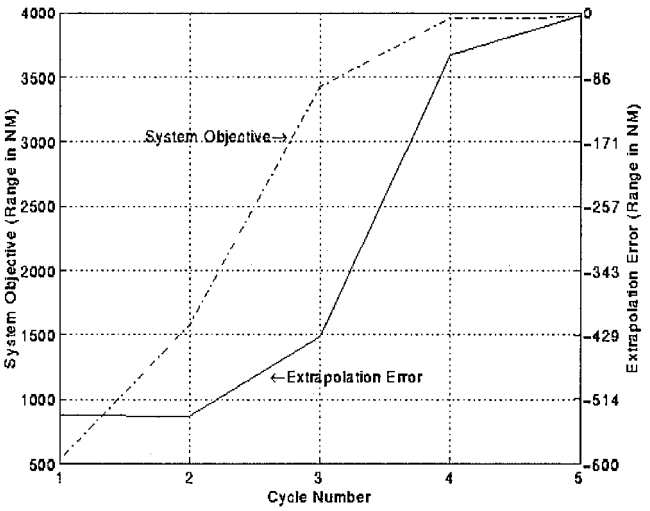

Fig. 5 Range and extrapolation error histogram.

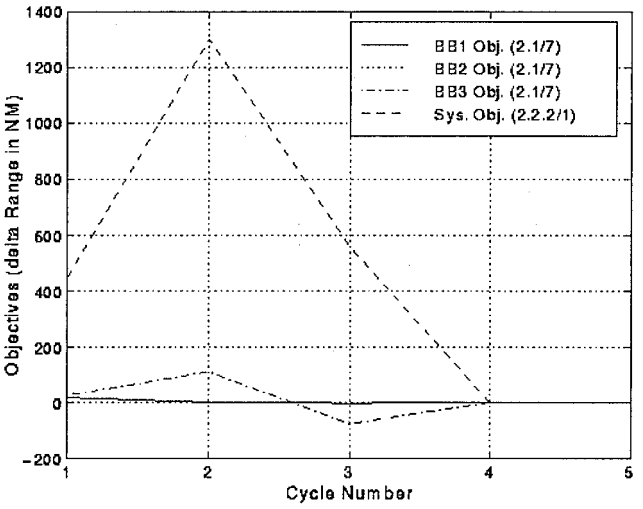

Fig. 7 BB and system contributions to range.

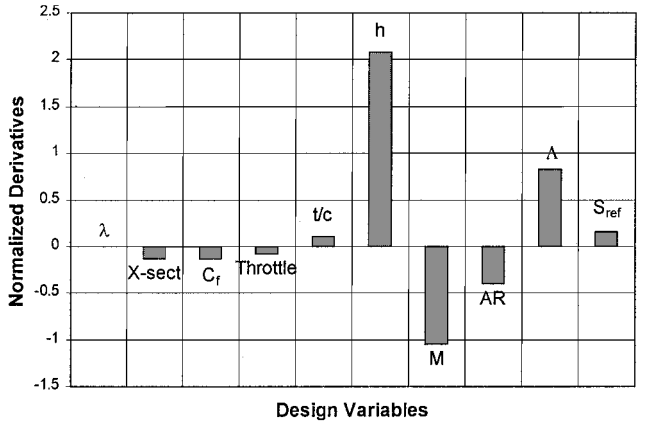

Fig. 6 Range sensitivities (first cycle).

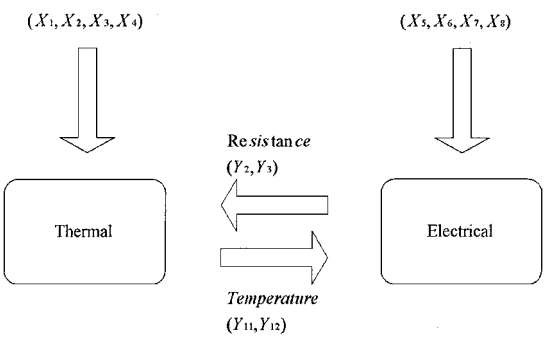

Fig. 8 Electronic packaging data dependencies.

data link the subsystem and system level optimizations. There are two variants of the method, BLISS/*A* and BLISS/*B*, that differ by the details of that linkage. In the paper, the method algorithm was laid out in detail for a system of three subdomains (modules). Its generalization to *NB* subdomains is straightforward. The same algorithm may be used to decompose any of the local optimizations, hence optimization may be conducted at more than two levels.

MATLAB and iSIGHT programming languages were used to implement and test the method prototype on a simplified, conceptual level supersonic business jet design, and on a detailed design of an electronic device. Dimensionality and complexity of the preliminary test cases were intentionally kept very low for an expeditious assessment of the method potential before more resources are invested in further development. Favorable agreement with the benchmark results and a satisfactory convergence observed in the tests provided motivation for such development and future testing in larger applications.

Assessment of BLISS at the development status is as follows. BLISS relies on linearization of a generally nonlinear optimization, therefore, its effectiveness depends on the degree of nonlinearity. As any gradient-guided method, it guarantees a cycle-to-cycle improvement, but if the problem is nonconvex, its convergence to the global optimum depends on the starting point and may strongly depend on the move limits. In this regard, BLISS's strong points are in the procedure being open to human intervention between the cycles

and in the autonomy of the subdomain optimizations in local variables. These optimizations may be conducted by any means deemed to be most suitable by disciplinary experts, hence nonconvexity, and strong nonlinearities in terms of the local variables often encountered in subdomains (e.g., the local buckling in thin-walled structures), are isolated and prevented from slowing down the system-level optimization convergence. On the other hand, the optimization robustness may be adversely affected by the local constraints leaving and entering the active constraint set. The effect of this on BLISS/*A* is much less than on BLISS/*B*. This is probably the only reason to continue the development of BLISS/*A* alongside with BLISS/*B*, even though BLISS/*B* has a distinct advantage in simplicity and a much lower computational cost. Once there is more information on the relative merits and demerits of both variants, the better variant may be selected.

The demand BLISS puts on the computer storage is the same the subdomains would require for their own, stand-alone optimizations, with the exception of the generation and solution of the GSE. If there is a pair of BBs that exchange large number of the $y_{r,s,i}$ quantities, dimensionality of the corresponding matrices that store the derivatives, and computational cost of these derivatives needed to form GSE, may become prohibitive. Some relief may be provided here by application of condensation techniques and by deleting from GSE those derivative matrices that are known to have negligible effect on the system behavior.

The principal advantage of BLISS appears to lie in its separating overall system design considerations from the considerations of the detail. This makes the resulting mapping of its algorithm fit well on diverse, and potentially dispersed, human organizations. This advantage remains to be demonstrated in further development toward large-scale, complex applications.

Appendix: Generalized GSE to Include Optimized Subsystems

This appendix details a new generalization of the GSE/OS.

Consider a system in Fig. 1 in which each BB has been optimized. The optimization has turned the $\boldsymbol{X}$ and $\boldsymbol{Y}$ output variables into functions of the $\boldsymbol{Z}$ and $\boldsymbol{Y}$ inputs. Hence the following functions exist:

$$\boldsymbol{Y}_{1,2} = f_{y,1,2}(\boldsymbol{Z}, \boldsymbol{X}_2, \boldsymbol{Y}_{2,1}, \boldsymbol{Y}_{2,3}) \tag{A1}$$

$$\boldsymbol{Y}_{1,3} = f_{y,1,3}(\boldsymbol{Z}, \boldsymbol{X}_3, \boldsymbol{Y}_{3,1}, \boldsymbol{Y}_{3,2}) \tag{A2}$$

$$\boldsymbol{Y}_{2,1} = f_{y,2,1}(\boldsymbol{Z}, \boldsymbol{X}_1, \boldsymbol{Y}_{1,2}, \boldsymbol{Y}_{1,3}) \tag{A3}$$

$$\boldsymbol{Y}_{2,3} = f_{y,2,3}(\boldsymbol{Z}, \boldsymbol{X}_3, \boldsymbol{Y}_{3,1}, \boldsymbol{Y}_{3,2}) \tag{A4}$$

$$\boldsymbol{Y}_{3,1} = f_{y,3,1}(\boldsymbol{Z}, \boldsymbol{X}_1, \boldsymbol{Y}_{1,2}, \boldsymbol{Y}_{1,3}) \tag{A5}$$

$$\boldsymbol{Y}_{3,2} = f_{y,3,2}(\boldsymbol{Z}, \boldsymbol{X}_2, \boldsymbol{Y}_{2,1}, \boldsymbol{Y}_{2,3}) \tag{A6}$$

$$\boldsymbol{X}_1 = f_{x,1}(\boldsymbol{Z}, \boldsymbol{Y}_{1,2}, \boldsymbol{Y}_{1,3}) \tag{A7}$$

$$\boldsymbol{X}_2 = f_{x,2}(\boldsymbol{Z}, \boldsymbol{Y}_{2,1}, \boldsymbol{Y}_{2,3}) \tag{A8}$$

$$\boldsymbol{X}_3 = f_{x,3}(\boldsymbol{Z}, \boldsymbol{Y}_{3,1}, \boldsymbol{Y}_{3,2}) \tag{A9}$$

The same implicit function theorem that is the basis of the original GSE derivation may be applied to the preceding equations to obtain $\partial \boldsymbol{Y}/\partial \boldsymbol{Z}$. For example, by applying chain differentiation to $\boldsymbol{Y}_{2,1}$ treated as a subset of $\boldsymbol{Y}_2$ so that $f_{y,2,1}$ becomes a subset of $f_{y,2}$ we obtain

$$\frac{\partial \boldsymbol{Y}_2}{\partial z_k} = \partial f_{y,z}/\partial z_k + \frac{\partial \boldsymbol{Y}_2}{\partial \boldsymbol{X}_2}\frac{\partial \boldsymbol{X}_2}{\partial z_k} + \frac{\partial \boldsymbol{Y}_2}{\partial \boldsymbol{Y}_1}\frac{\partial \boldsymbol{Y}_1}{\partial z_k} + \frac{\partial \boldsymbol{Y}_2}{\partial \boldsymbol{Y}_3}\frac{\partial \boldsymbol{Y}_3}{\partial z_k} \tag{A10}$$

and for $\boldsymbol{X}_2$, again as one example,

$$\frac{\partial \boldsymbol{X}_2}{\partial z_k} = \frac{\partial f_{x,2}}{\partial z_k} + \frac{\partial \boldsymbol{X}_2}{\partial \boldsymbol{Y}_1}\frac{\partial \boldsymbol{Y}_1}{\partial z_k} + \frac{\partial \boldsymbol{X}_2}{\partial \boldsymbol{Y}_3}\frac{\partial \boldsymbol{Y}_3}{\partial z_k} \tag{A11}$$

where the $\partial \boldsymbol{X}/\partial z_k$ and $\partial \boldsymbol{Y}/\partial z_k$ terms are the derivatives we seek, whereas the remaining are partial derivatives of two different kinds. The derivatives of $\boldsymbol{Y}_r$ with respect to $\boldsymbol{Y}_s$ and $\boldsymbol{Y}_r$ with respect to $\boldsymbol{X}_r$ are obtained from BBSA$_r$, using any sensitivity analysis algorithm appropriate for the particular BB$_r$ (including the option of finite differencing). The derivatives of $\boldsymbol{X}_r$ with respect to z_k and $\boldsymbol{X}_r$ with respect to $\boldsymbol{Y}_s$ are produced by an analysis of optimum for sensitivity to parameters, BBOSA$_r$, explained in later in this appendix.

The chain-derivative expressions for $\boldsymbol{Y}_1$, $\boldsymbol{Y}_3$, $\boldsymbol{X}_1$, and $\boldsymbol{X}_3$ look similar to Eqs. (A10) and (A11); the differences are only in the subscripts. When the entire set of six chain-derivative expressions is written, it forms a set of simultaneous, algebraic equations in which the derivatives such as $\partial \boldsymbol{Y}_2/\partial z_k$ and $\partial \boldsymbol{X}_2/\partial z_k$ appear as unknowns. For the case of the three-BB system, these equations may be presented in a matrix format as

$$[M_{yy}]\left\{\frac{\partial \boldsymbol{Y}}{\partial z_k}\right\} + [M_{yx}]\left\{\frac{\partial \boldsymbol{X}}{\partial z_k}\right\} = \frac{\partial f_y}{\partial z_k}$$

$$[M_{xy}]\left\{\frac{\partial \boldsymbol{Y}}{\partial z_k}\right\} + [M_{xx}]\left\{\frac{\partial \boldsymbol{X}}{\partial z_k}\right\} = \frac{\partial f_x}{\partial z_k} \tag{A12}$$

The internal structure of the M-matrices in Eq. (A12) as follows.

For $[M_{yy}]$:

$$\left[\begin{array}{c|c|c} I & -\dfrac{\partial \boldsymbol{Y}_1}{\partial \boldsymbol{Y}_2} & -\dfrac{\partial \boldsymbol{Y}_1}{\partial \boldsymbol{Y}_3} \\ \hline -\dfrac{\partial \boldsymbol{Y}_2}{\partial \boldsymbol{Y}_1} & I & -\dfrac{\partial \boldsymbol{Y}_2}{\partial \boldsymbol{Y}_3} \\ \hline -\dfrac{\partial \boldsymbol{Y}_3}{\partial \boldsymbol{Y}_1} & -\dfrac{\partial \boldsymbol{Y}_3}{\partial \boldsymbol{Y}_2} & I \end{array}\right]$$

For $[M_{yx}]$:

$$\left[\begin{array}{c|c|c} -\dfrac{\partial \boldsymbol{Y}_1}{\partial \boldsymbol{X}_1} & 0 & 0 \\ \hline 0 & -\dfrac{\partial \boldsymbol{Y}_2}{\partial \boldsymbol{X}_2} & 0 \\ \hline 0 & 0 & -\dfrac{\partial \boldsymbol{Y}_3}{\partial \boldsymbol{X}_3} \end{array}\right]$$

For $[M_{xy}]$:

$$\left[\begin{array}{c|c|c} 0 & -\dfrac{\partial \boldsymbol{X}_1}{\partial \boldsymbol{Y}_2} & -\dfrac{\partial \boldsymbol{X}_1}{\partial \boldsymbol{Y}_3} \\ \hline -\dfrac{\partial \boldsymbol{X}_2}{\partial \boldsymbol{Y}_1} & 0 & -\dfrac{\partial \boldsymbol{X}_2}{\partial \boldsymbol{Y}_3} \\ \hline -\dfrac{\partial \boldsymbol{X}_3}{\partial \boldsymbol{Y}_1} & -\partial \boldsymbol{X}_3, \partial \boldsymbol{Y}_2 & 0 \end{array}\right]$$

For $[M_{xx}]$:

$$\left[\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & I & 0 \\ \hline 0 & 0 & I \end{array}\right]$$

Again, here all $\boldsymbol{Y}_{r,s}$ (and the corresponding $f_{y,r,s}$) are folded into $\boldsymbol{Y}_r$ (and $f_{y,r}$) for compactness, and the terms fall into the categories introduced earlier as follows: M_{yy}, M_{yx}, and $\partial f_y/\partial z_k$ from BBSA; and M_{xy} and $\partial f_x/\partial z_k$ from BBOSA. Generalization of the pattern shown for three BBs to a system with n BBs is straightforward. As in GSE, one may obtain $\partial \boldsymbol{Y}_2/\partial z_k$ and $\partial \boldsymbol{X}_2/\partial z_k$ for all z_k, $k = 1 \rightarrow NZ$ by means of one of the efficient techniques for linear equations with many right-hand sides.

Acknowledgments

Srinivas Kodiyalam of the Engineous Co. provided the test results for Aircraft Optimization and the Electronic Package Optimization using a software package iSIGHT. His contribution is gratefully acknowledged. The engine data were obtained from the Attachment to AIAA/UTC/Pratt–Whitney Undergraduate Individual Aircraft Design Competition 1995/96 issued by AIAA.

References

[1]Balling, R. J., and Sobieszczanski-Sobieski, J., "Optimization of Coupled Systems: A Critical Overview of Approaches," *AIAA Journal*, Vol. 34, No. 1, 1996, pp. 6–17.

[2]Sobieszczanski-Sobieski, J., and Haftka, R. T., "Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments," *Structural Optimization*, Vol. 14, No. 1, 1997, pp. 1–23.

[3]Sobieszczanski-Sobieski, J., "Sensitivity of Complex, Internally Coupled Systems," *AIAA Journal*, Vol. 28, No. 1, 1990, pp. 153–160.

[4]Olds, J., "The Suitability of Selected Multidisciplinary Design and Optimization Techniques to Conceptual Aerospace Vehicle Design," AIAA Paper 92-4791, 1992.

[5]Olds, J., "System Sensitivity Analysis Applied to the Conceptual Design of a Dual-Fuel Rocket SSTO," AIAA Paper 94-4339, 1994.

[6]Sobieszczanski-Sobieski, J., "Optimization by Decomposition: A Step from Hierarchic to Non-Hierarchic Systems," CP-3031, NASA, Pt. 1, 1988; NASA TM-101494. 1988.

[7]Renaud, J. E., and Gabriele, G. A., "Sequential Global Approximation in Non-Hierarchic System Decomposition and Optimization," *Advances in Design Automation and Design Optimization*, edited by G. Gabriele, ASME Publication DE-Vol. 32-1, American Society of Mechanical Engineers, 1991, pp. 191–200.

[8]Renaud, J. E., and Gabriele, G. A., "Improved Coordination in Nonhierarchic System Optimization," *AIAA Journal*, Vol. 31, No. 12, 1993, pp. 2367–2373.

[9]Renaud, J. E., and Gabriele, G. A., "Approximation in Nonhierarchic System Optimization," *AIAA Journal*, Vol. 32, 1994, pp. 198–205.

[10]Stelmack, M., and Batill, S., "Neural Network Approximation of Mixed Continuous/Discrete Systems in Multidisciplinary Design," AIAA Paper 98-0916, 1998.

[11]Braun, R. D., and Kroo, I. M., "Development and Application of the Collaborative Optimization Architecture in a Multidisciplinary Design Environment," *Multidisciplinary Design Optimization State of the Art*, edited by N. Alexandrov and Y. Husseini, Proceedings in Applied Mathematics 80, Society for Industrial and Applied Mathematics, 1997.

[12]Sobieski, I. P., and Kroo, I. M., "Collaborative Optimization Using Response Surface Estimation," AIAA Paper 98-0915, 1998.

[13]Adelman, H. M., and Haftka, R. T., "Sensitivity Analysis of Discrete Systems," *Structural Optimization: Status and Promise*, edited by M. P. Kamat, AIAA, Washington, DC, 1993, pp. 291–313.

[14]Sobieszczanski-Sobieski, J., Barthelemy, J.-F. M., and Riley, K. M., "Sensitivity of Optimum Solutions to Problems Parameters," *AIAA Journal*, Vol. 20, No. 9, 1982, pp. 1291–1299.

[15]Vanderplaats, G. N., and Cai, H. D., "Alternative Methods for Calculating Sensitivity of Optimized Designs to Problem Parameters," CP-2457, NASA, Sept. 1986.

[16]Barthelemy, J.-F., and Sobieszczanski-Sobieski, J., "Optimum Sensitivity Derivatives of Objective Functions in Nonlinear Programming," *AIAA Journal*, Vol. 21, No. 5, 1983, pp. 797–799.

[17]Haftka, R. T., and Gurdal, Z., *Elements of Structural Optimization*, Kluwer, Dordrecht, The Netherlands, 1992, p. 172.

[18]Alexandrov, N., "Robustness Properties of a Trust Region Framework for Managing Approximations in Engineering Optimization," *Proceedings of the AIAA/NASA/USAF 6th Multidisciplinary Analysis and Optimization Symposium*, AIAA, Reston, VA, 1996, pp. 1056–1059.

[19]Barthelemy, J.-F., and Sobieszczanski-Sobieski, J., "Extrapolation of Optimum Design Based on Sensitivity Derivatives," *AIAA Journal*, Vol. 21, No. 6, 1983, pp. 913–915.

[20]Sobieszczanski-Sobieski, J., James, B., and Dovi, A., "Stuctural Optimization by Multi-Level Optimization," *AIAA Journal*, Vol. 23, No. 11, 1985, pp. 1775–1782.

[21]Sobieszczanski-Sobieski, J., "Optimization by Decomposition in Structural and Multidisciplinary Applications," *Optimization of Large Structural Systems*, Kluwer, Dordrecht, The Netherlands, 1993.

[22]Hock, W., and Schittkowski, K., "Test Examples for Nonlinear Programming Codes," *Lecture Notes in Economics and Mathematical Systems*, edited by M. Beckmann and H. P. Kunzi, Heidelberg, Germany, 1981.

[23]Raymer, D. P., *Aircraft Design: A Conceptual Approach*, 2nd ed., AIAA Education Series, AIAA, Washington, DC, 1992.

[24]MATLAB Manual, Ver. 5.0, 1997, and MATLAB Optimization Toolbox, User's Guide, MathWorks, Inc., Natick, MA, 1996.

[25]iSIGHT, Designers and Developers Manual, Ver. 3.1, Engineous Software, Inc., Morrisville, NC, 1998.

[26]Lokunathan, A. N., Brockman, J. B., and Renaud, J. E., "Concurrent Design of Manufacturable Integrated Circuits," *Proceedings of the AIAA/NASA/USAF 6th Multidisciplinary Analysis and Optimization Symposium*, AIAA, Reston, VA, 1996, pp. 1010–1018.

[27]Padula, S. L., Alexandrov, N., and Green, L. L., "MDO Test Suite at NASA Langley Research Center," AIAA Paper 96-4028, 1996.

A. D. Belegundu
Associate Editor