AIAA JOURNAL
Vol. 42, No. 2, February 2004

Analysis and Enhancement of Collaborative Optimization for Multidisciplinary Design

JiGuan G. Lin*
LinSys, Lexington, Massachusetts 02421-5801

Collaborative optimization is examined with a focus on its consistency equality constraints and the effect of their vanishing gradients. Attempts are made to enhance its analytical and computational performance. General theoretical investigation and specific analytical study show that when system-level variables satisfy the consistency equality constraints, system-level optimization generally will encounter no-solution and convergence difficulties if it depends on Lagrange multipliers in the Karush–Kuhn–Tucker necessary conditions. Thus, to avoid such difficulties, algorithms and methods that need no Lagrange multipliers at all are used. Alternatively, to avoid multiplier troubles in applying the Karush–Kuhn–Tucker conditions, the consistency constraints are replaced by other constraints that are not sums of squared differences. Specific computational study shows that enhanced application of the penalty-function method can yield approximate solutions very close to the solution of the predecomposed problem. Convergence is obtained with little difficulty when only the least of the freedom allowable by collaborative optimization is taken in using or producing variants of the targets.

Nomenclature

d_i	=	subsystem i discrepancy in using or producing target values
$\boldsymbol{g}_i$	=	vector of (inequality) design constraints on subsystem i
$\boldsymbol{h}_i$	=	vector of analysis equations in subsystem i
J_{sys}	=	system-level objective
$\boldsymbol{s}$	=	vector of all shared design variables
$\boldsymbol{s}_i$	=	vector of subsystem i design variables that are also needed by other subsystems
$\boldsymbol{x}_i$	=	vector of subsystem i design variables that are used solely by subsystem i
$\boldsymbol{y}$	=	vector of all subsystem outputs, including all $\boldsymbol{z}_i$
$\boldsymbol{y}_i$	=	vector of outputs from subsystem i
$\boldsymbol{z}_i$	=	vector of outputs from other subsystems that are needed by subsystem i as input
$\partial\boldsymbol{\eta}_i/\partial\boldsymbol{x}_i$	=	Jacobian of vector $\boldsymbol{\eta}_i$ with respect to vector $\boldsymbol{x}_i$
$\boldsymbol{\eta}_i$	=	vector of subsystem i variants of output $\boldsymbol{y}_i$
λ_i	=	Lagrange multiplier associated with ith consistency constraint
$\boldsymbol{\mu}_i$	=	vector of Lagrange multipliers associated with subsystem i design constraints $\boldsymbol{g}_i$
$\boldsymbol{\sigma}_i$	=	vector of subsystem i variants of shared design variables $\boldsymbol{s}_i$
$\boldsymbol{\zeta}_i$	=	vector of subsystem i variants of input $\boldsymbol{z}_i$

Subscript

$*$	=	subsystem-level optimum

Superscript

$*$	=	system-level optimum

Received 17 July 2002; presented as Paper 2002-5503 at the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA, 4–6 September 2002; revision received 24 March 2003; accepted for publication 25 August 2003.

*Principal Engineer; linjiguan@email.com. Member AIAA.

I. Introduction

COLLABORATIVE optimization[1,2] (CO) is a bilevel approach to multidisciplinary optimization (MDO) that decomposes a design problem into subproblems along the lines of constituent disciplines. It tries to preserve autonomy in discipline-specific calculations at the subsystem level while enforcing interdisciplinary consistency through equality constraints on subsystem-minimized discrepancies at the system level. The system level passes out to all subsystems a set of targets for shared design variables and subsystem outputs. Any subsystem, for example, the ith, is allowed to disagree on the targets by using or producing different values if its total discrepancy d_i is minimized as the sole objective. Discrepancy d_i is defined, for example, as sum of squared differences (SSD) between target values and subsystem variants. The system level optimizes the targets with respect to a system objective subject to the constraint that all subsystem-minimized discrepancies d_{i*} equal zero. Such a concept of preserving disciplinary autonomy, especially, the freedom it allows, is very attractive; for instance, in Ref. 3, an entire long paragraph was devoted to describing its beauty. Moreover, CO was the only bilevel approach selected for starting the MDO method evaluation in Refs. 4 and 5 and for comparison with conventional optimization techniques in a reusable launch-vehicle design study in Ref. 6, despite many other MDO approaches and variants being included in a critical review and extensive survey.[7,8] Besides being widely applied (see listing in Ref. 9), the approach has been expanded, extended, evaluated, or modified by many.[3–5,7,10,11]

Kodiyalam[5] conducted a systematic evaluation of CO and two other MDO methods but found that CO had failure in five of six test problems selected from NASA's MDO Test Suite to converge to a point satisfying first-order necessary optimality conditions. Among these five, four were class II problems with engineering contents, exhibiting some of the salient features of realistic MDO problems.[4] Alexandrov and Lewis[12] then investigated rigorously the CO formulation. Their careful mathematical analysis and clear numerical demonstration revealed that some analytical properties of CO could lead to serious computational difficulties when applying constrained optimization algorithms: Vanishing Jacobian of the consistency constraints could cause failure to converge to a solution, etc. Earlier, Braun et al.[13] pointed out convergence was not achieved for a CO problem due to the line search algorithm in NPSOL. They resolved the difficulty by modifying the consistency constraints to inequalities (with no tolerances). Even earlier, Thareja and Haftka[14] used inequality discrepancy constraints (also with no tolerances) on the system level of a CO-like two-level formulation of a structural design problem with a two-level solution approach. Their limited

results could be one of the earliest indications of convergence difficulty with CO. On the other hand, the equality constraints used in their two-level formulation with single-level solution approach were different from the consistency constraints used in CO, and their constraint gradients did not necessarily vanish when those equality constraints were satisfied. Cormier et al.[6] applied CO to a complicated multidisciplinary reusable launch-vehicle design with inequality discrepancy constraints. Their convergence rate was fairly sensitive to the tolerances allowed on the discrepancy constraints, and they had no success obtaining any kind of results using a "cold" starting point that was not necessarily an already converged design configuration. They pointed out an undesirable side effect of allowing tolerances: The errors in the subsystems tended to gather in a few of the most sensitive of the design variables. Alexandrov and Lewis[12] also applied inequality discrepancy constraints to two simple CO problems but were not satisfied with the results even though the tolerances were varied parametrically.

This paper examines the CO approach further, with focus on the consistency equality constraints and the effect of their vanishing gradients. To enhance the approach, the study attempts to gain greater insight by using three different kinds of examination: general theoretical investigations, specific analytical studies, and specific computational studies. First, in Sec. II the CO formulation is restated. An anatomy of the consistency constraints is given in Sec. III, with focus on theoretical aspects of both system and subsystem levels in the formulation. For gaining specific qualitative insights, analytical derivation of CO solutions and examination of the associated postoptimality sensitivity analysis are given in Sec. IV. Computational studies with three different subsystem-level formulations are reported in Sec. V, where the penalty-function method is used so that the investigation can focus on convergence without the instant-lockup effect of enforcing the consistency equality constraints.

II. Collaborative-Optimization Problem Formulation

Suppose a design problem, for example, of an airplane wing, is decomposed into multiple disciplinary subsystems, such as aerodynamics, structures, performance, etc. Divide the design variables of subsystem i into two exclusive groups: vector $\boldsymbol{s}_i$ representing global (shared) design variables that are also needed by other subsystems and vector $\boldsymbol{x}_i$ representing local design variables that are used by subsystem i only. Denote by vector $\boldsymbol{z}_i$ those outputs from other subsystems that are needed by subsystem i for input and by vector $\boldsymbol{y}_i$ all outputs from subsystem i. Consider all shared design variables $\boldsymbol{s}$ and all subsystem outputs $\boldsymbol{y}$ as independent optimization variables of the system level. Note that $\boldsymbol{y}$ includes all $\boldsymbol{z}_i$. Freeze all system-level variables $(\boldsymbol{s}, \boldsymbol{y})$ in each subsystem optimization and treat them as parameters (or dummy variables) with specific values issued by the system level as targets.

Individual subsystems often do not have sufficient local design variables to satisfy their own design constraints and analysis equations, much less enough for matching the targets passed down from the system level. The CO formulation is particularly helpful in such a situation. It permits each subsystem to disagree with other subsystems on the target values by using or producing local variants and, thus, enables concurrent, or separate, discipline-specific calculations.

Denote by vector $\boldsymbol{\sigma}_i$ subsystem i variants of shared design variables $\boldsymbol{s}_i$, vector $\boldsymbol{\zeta}_i$ variants of $\boldsymbol{z}_i$ as input, and vector $\boldsymbol{\eta}_i$ variants of $\boldsymbol{y}_i$ as output. For each given vector of target values, define subsystem i discrepancy in using or producing them as

$$d_i = (\boldsymbol{\sigma}_i - \boldsymbol{s}_i)^T(\boldsymbol{\sigma}_i - \boldsymbol{s}_i) + (\boldsymbol{\zeta}_i - \boldsymbol{z}_i)^T(\boldsymbol{\zeta}_i - \boldsymbol{z}_i) + (\boldsymbol{\eta}_i - \boldsymbol{y}_i)^T(\boldsymbol{\eta}_i - \boldsymbol{y}_i) \tag{1}$$

CO then requires that each subsystem minimize discrepancy d_i as the sole objective while trying to satisfy its design constraints and analysis equations. The system level, on the other hand, ensures all minimized discrepancies d_i* vanish while optimizing the choice of targets $(\boldsymbol{s}, \boldsymbol{y})$ with respect to system-level objective J_{sys}. The two-level CO formulation[2] in a general setting is given next. For subsystem i, given $\boldsymbol{s}, \boldsymbol{y}$, find $\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i$:

$$\text{minimize} \quad d_i \tag{2a}$$

the discrepancy objective, subject to

$$\boldsymbol{g}_i(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i, \boldsymbol{\eta}_i) \le 0 \tag{2b}$$

with $\boldsymbol{\eta}_i$ solved from

$$\boldsymbol{h}_i(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i, \boldsymbol{\eta}_i) = 0 \tag{2c}$$

For system level, given $d_{1*}, d_{2*} \ldots$ (minimized discrepancies), find $\boldsymbol{s}, \boldsymbol{y}$:

$$\text{minimize} \quad J_{\text{sys}}(\boldsymbol{s}, \boldsymbol{y}) \tag{3a}$$

subject to

$$d_{i*}(\boldsymbol{s}, \boldsymbol{y}) = 0 \quad \text{all} \quad i \tag{3b}$$

(consistency constraints). Note that of each subsystem i, analysis outputs $\boldsymbol{\eta}_i$ are functions of subsystem optimization variables $(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ and minimized discrepancies d_{i*} are functions of targets $(\boldsymbol{s}_i, \boldsymbol{z}_i, \boldsymbol{y}_i)$ from the system level. Also note that for targets $(\boldsymbol{s}, \boldsymbol{y})$ to be system-level feasible all consistency constraints (3b) must be satisfied.

III. General Anatomy of Consistency Constraints

The vanishing Jacobian of the consistency constraints could cause convergence difficulty at the system level.[12] The questions are why and how. Let us examine the system and subsystem levels from general theoretical viewpoints. First, differentiating the ith consistency constraint function d_{i*} with respect to system-level variables $(\boldsymbol{s}, \boldsymbol{y})$, we readily get from Eq. (1)

$$\frac{\partial d_{i*}}{\partial(\boldsymbol{s}, \boldsymbol{y})} = 2(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)^T\left[\frac{\partial \boldsymbol{\sigma}_{i*}}{\partial(\boldsymbol{s}, \boldsymbol{y})} - \frac{\partial \boldsymbol{s}_i}{\partial(\boldsymbol{s}, \boldsymbol{y})}\right] + 2(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)^T \times\left[\frac{\partial \boldsymbol{\zeta}_{i*}}{\partial(\boldsymbol{s}, \boldsymbol{y})} - \frac{\partial \boldsymbol{z}_i}{\partial(\boldsymbol{s}, \boldsymbol{y})}\right] + 2(\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i)^T\left[\frac{\partial \boldsymbol{\eta}_{i*}}{\partial(\boldsymbol{s}, \boldsymbol{y})} - \frac{\partial \boldsymbol{y}_i}{\partial(\boldsymbol{s}, \boldsymbol{y})}\right] \tag{4}$$

where $\partial d_{i*}/\partial(\boldsymbol{s}, \boldsymbol{y})$ denotes the row vector of sensitivity derivatives of subsystem i minimized discrepancy d_{i*}, with respect to targets $(\boldsymbol{s}, \boldsymbol{y})$ as parameters. Its column version is a consistency-constraint gradient. The consistency-constraint Jacobian is the matrix consisting of $\partial d_{i*}/\partial(\boldsymbol{s}, \boldsymbol{y})$ as the ith row. Similarly, $\partial \boldsymbol{\sigma}_{i*}/\partial(\boldsymbol{s}, \boldsymbol{y})$, $\partial \boldsymbol{s}_i/\partial(\boldsymbol{s}, \boldsymbol{y})$, $\partial \boldsymbol{\zeta}_{i*}/\partial(\boldsymbol{s}, \boldsymbol{y})$, etc., are Jacobian matrices representing sensitivity derivatives of vectors $\boldsymbol{\sigma}_{i*}, \boldsymbol{s}_i, \boldsymbol{\zeta}_{i*}, \ldots$, respectively.

Because each d_{i*} is a SSD, constraints (3b) are satisfied if and only if all discrepancy terms vanish, that is, $\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i = 0$, $\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i = 0$, and $\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i = 0$. It follows immediately from Eq. (4) that the consistency-constraint Jacobian necessarily vanishes whenever the consistency equality constraints are satisfied.[12]

A. System-Level Consequences of Satisfying Consistency Constraints

Now, what is the impact of vanishing consistency-constraint gradients on system-level optimization? Let us look into system-level optimality conditions. The equality-constraint version of the Karush–Kuhn–Tucker (KKT) first-order necessary conditions are

$$\frac{\partial J_{\text{sys}}}{\partial(\boldsymbol{s}, \boldsymbol{y})} + \sum_i \lambda_i \frac{\partial d_{i*}}{\partial(\boldsymbol{s}, \boldsymbol{y})} = 0 \tag{5}$$

Because all consistency-constraint gradients vanish whenever targets $(\boldsymbol{s}, \boldsymbol{y})$ are system-level feasible, the KKT conditions simply become $\partial J_{\text{sys}}/\partial(\boldsymbol{s}, \boldsymbol{y}) = 0$ no matter what values Lagrange multipliers λ_i have and what system-level objective function J_{sys} is. Because objective gradient $\partial J_{\text{sys}}/\partial(\boldsymbol{s}, \boldsymbol{y})$ does not necessarily vanish at system-level feasible targets, not even at the optimum solution of the predecomposed problem (example in Sec. IV.B), the KKT conditions cannot be satisfied at any system-level feasible targets at all. (See Ref. 12 for more detailed analysis.) Therefore, the system-level

problem rarely will have a solution satisfying both the consistency constraints and the KKT conditions.

In other words, Lagrange multipliers are not really computable when targets $(\boldsymbol{s}, \boldsymbol{y})$ become system-level feasible and can be arbitrarily large if tried by the KKT conditions (because their multiplicands are vanishing). This becomes the main source of computational difficulties. NPSOL and all other algorithms that employ such Lagrange multipliers in their merit functions have serious difficulty in making decisions about termination, etc., and will likely stop essentially arbitrarily anywhere. (Again, see Ref. 12 for detailed discussions on unfortunate computational consequences of the inability to characterize solutions via the KKT conditions.)

Let us trace back to an origin of the conditions. It was specifically stated in Ref. 15 that the KKT conditions were based on the assumption that all constraints had satisfied constraint qualification[15](CQ). For inequality constraints, there are many ways to satisfy CQ as well as many weakened forms of CQ (e.g., see Refs. 16 and 17). For equality constraints only, the equivalent assumption is regularity,[18] and there are essentially two ways to test for the regularity assumption[18]: (1) linearity test, where the constraint functions are all linear in optimization variables $(\boldsymbol{s}, \boldsymbol{y})$ or (2) normality test, where their gradients are linearly independent. Note that the normality test[18] on equality constraints corresponds to the nondegeneracy condition[16] on inequality constraints and that the latter is one of many sufficient conditions[16,17] for inequality constraints to satisfy CQ.

Because the consistency constraints (3b) are nonlinear, linear independence of the constraint gradients becomes the only test available. Now because all consistency-constraint gradients vanish at system-level feasible targets, they definitely cannot be linearly independent at all and, hence, cannot satisfy the normality test. Therefore, the KKT conditions are inapplicable at any system-level feasible targets. (Even though one could argue that not satisfying a sufficient condition, the normality test, might not necessarily mean failure to satisfy CQ, the failure is confirmed in the study example when CQ is checked directly by its definition; see the Appendix.)

That the consistency constraints are all equality constraints of the SSD type is the real cause of the trouble. Because feasible sets defined by equality constraints only are hollow in general, whereas those defined by inequality constraints have an interior, equality constraints naturally are more likely to fail CQ. (Illustrative failure examples are given in Ref. 18.) Their SSD type simply guarantees the failure.

On the other hand, the finite-dimensional Fritz John condition (see Refs. 17 and 18) is applicable without having to satisfy CQ first. By its equality-constraint version, we readily have Theorem 1.

Theorem 1. There is a nonzero vector of multipliers $\lambda_0, \lambda_1, \ldots,$ with $\lambda_0 \geq 0$ such that

$$\lambda_0 \frac{\partial J_{\text{sys}}}{\partial(\boldsymbol{s}, \boldsymbol{y})} + \sum_i \lambda_i \frac{\partial d_{i*}}{\partial(\boldsymbol{s}, \boldsymbol{y})} = 0 \tag{6}$$

Fritz John condition (6) is satisfied with $\lambda_0 = 0$ when targets $(\boldsymbol{s}, \boldsymbol{y})$ are system-level feasible, whether or not the objective gradient vanishes. (Of course, it is also satisfied with $\lambda_0 = 1$ if the objective gradient vanishes.) Again, whenever the consistency-constraint gradients vanish, the Lagrange multipliers can take on any values, even those unrelated to the objective gradient.

This condition is generally useless for computation when $\lambda_0 = 0$. If it has any use, it is to suggest minimizing the objective function J_{sys} by direct search as follows: Evaluate objective J_{sys} directly at those points where its multiplier λ_0 must be zero and compare those objective values to find the smallest as a local optimum. See end of Sec. IV.C for an illustration.

All of this means that inside the system-level feasible region, system-level optimization should depend on no multiplier conditions, whether KKT or Fritz John. Alternatively, a plausible gradient-based approach is to approximate by system-level infeasible targets, for example, using the penalty-function method. It makes sense simply because consistency-constraint gradients do not necessarily vanish when the consistency constraints are not satisfied.

B. Subsystem-Level Consequences of Satisfying Consistency Constraints

What is the impact on subsystem-level optimization if the consistency constraints are satisfied? Let us look at optimality conditions for subsystem i more closely. Freeze the targets at constant values as passed down from the system level, take derivatives of discrepancy objective d_i with respect to subsystem i input variants and local design variables $(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$, and evaluate the derivatives at subsystem i optimum solution $(\boldsymbol{\sigma}_{i*}, \boldsymbol{\zeta}_{i*}, \boldsymbol{x}_{i*})$. Then, we get

$$\frac{\partial d_{i*}}{\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)} = 2(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)^T \frac{\partial \boldsymbol{\sigma}_{i*}}{\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)} + 2(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)^T \frac{\partial \boldsymbol{\zeta}_{i*}}{\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)} + 2(\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i)^T \frac{\partial \boldsymbol{\eta}_{i*}}{\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)} \tag{7}$$

Lemma 1. Gradient $\partial d_{i*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ vanishes whenever ith consistency constraint is satisfied.

Note that the KKT first-order necessary conditions for subsystem i (KKTi) are

$$\frac{\partial d_i}{\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)} + \boldsymbol{\mu}_i^T \frac{\partial \boldsymbol{g}_i}{\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)} = 0, \qquad \boldsymbol{\mu}_i^T \boldsymbol{g}_i = 0, \qquad \boldsymbol{\mu}_i \geq 0 \tag{8}$$

It follows immediately from Lemma 1 that at subsystem optimum solution $(\boldsymbol{\sigma}_{i*}, \boldsymbol{\zeta}_{i*}, \boldsymbol{x}_{i*})$, KKT$i$ conditions (8) reduce to

$$\boldsymbol{\mu}_i^T \frac{\partial \boldsymbol{g}_{i*}}{\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)} = 0, \qquad \boldsymbol{\mu}_i^T \boldsymbol{g}_{i*} = 0, \qquad \boldsymbol{\mu}_i \geq 0 \tag{9}$$

whenever the ith consistency constraint is satisfied, where $\boldsymbol{g}_{i*}$ denotes subsystem constraints $\boldsymbol{g}_i$ evaluated at optimum solution $(\boldsymbol{\sigma}_{i*}, \boldsymbol{\zeta}_{i*}, \boldsymbol{x}_{i*})$ and $\partial \boldsymbol{g}_{i*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ the corresponding constraint Jacobian. Now this is just opposite to the system-level mess. If multipliers μ_{ik} are not all zero, then gradients $\partial g_{ik*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ of critical (active) design constraints must be linearly dependent by the first part of conditions (9). If nondegeneracy CQ has been assumed before applying the KKTi conditions, then the assumption becomes violated whenever the ith consistency constraint is satisfied.

Theorem 2. Suppose the ith consistency constraint is satisfied. If gradients $\partial g_{ik*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ of critical design constraints are linearly independent, then the amputated KKTi conditions (9) requires that all multipliers be zero: $\boldsymbol{\mu}_i = 0$.

Proof. Note first that the second and third parts of conditions (9) imply that $\mu_{ik} = 0$ if $g_{ik*} < 0$. Thus, the first part simplifies to

$$\sum_k{}' \mu_{ik} \frac{\partial g_{ik*}}{\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)} = 0$$

with the summation ranging over all critical constraints, $g_{ik*} = 0$. If all such gradients are linearly independent, then this equation holds only if all multipliers μ_{ik} associated with critical constraints are zero. Therefore, whether the constraints are critical or not, any associated multiplier μ_{ik} in conditions (9) must be zero. QED

The vanishing of all multipliers μ_{ik} is a good indication that discrepancy d_i has been minimized to zero and that the ith consistency constraint has been satisfied. However, a possible troublesome implication of Theorem 2 is that, whenever ith consistency constraint is satisfied, no design constraints can both be critical and have linearly independent gradients as required by the nondegeneracy condition[16] of CQ. Obviously, such an implication is avoided when the penalty method is used in the system-level optimization because satisfaction of the consistency constraints is only approximate until a system-level optimum solution is found.

C. Sufficient Conditions for Satisfying Consistency Constraints

Next is a simple discovery from the preceding anatomic exercise. It answers the reversed question: When will the ith consistency constraint be satisfied? Equivalently, what makes all discrepancy terms $(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)$, $(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)$, and $(\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i)$ vanish? First, what makes them vanish if gradient $\partial d_{i*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ vanishes? Let us write out gradient $\partial d_{i*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ more specifically than in Eq. (7) and in

the usual column form as follows:

$$\left(\frac{\partial d_{i*}}{\partial \boldsymbol{\sigma}_i}\right)^T = 2(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i) + 2\left(\frac{\partial \boldsymbol{\eta}_{i*}}{\partial \boldsymbol{\sigma}_i}\right)^T (\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i) \quad (10a)$$

$$\left(\frac{\partial d_{i*}}{\partial \boldsymbol{\zeta}_i}\right)^T = 2(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i) + 2\left(\frac{\partial \boldsymbol{\eta}_{i*}}{\partial \boldsymbol{\zeta}_i}\right)^T (\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i) \quad (10b)$$

$$\left(\frac{\partial d_{i*}}{\partial \boldsymbol{x}_i}\right)^T = 2\left(\frac{\partial \boldsymbol{\eta}_{i*}}{\partial \boldsymbol{x}_i}\right)^T (\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i) \quad (10c)$$

It is clear that discrepancy terms $(\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i)$ will then vanish if and only if Jacobian $\partial \boldsymbol{\eta}_{i*}/\partial \boldsymbol{x}_i$ has full row rank, that is, its rows are linearly independent, and that vanishing of both discrepancy terms $(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)$ and $(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)$ will follow. Next, to assume that gradient $\partial d_{i*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ vanishes is, by KKTi conditions (8), to assume that $\boldsymbol{\mu}_i^T \partial \boldsymbol{g}_{i*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ vanishes. Recall that no gradients $\partial g_{ik*}/\partial(\boldsymbol{\sigma}_i, \boldsymbol{\zeta}_i, \boldsymbol{x}_i)$ of critical design constraints should vanish because of the nondegeneracy condition of CQ. What is left is simply to assume that no design constraints are critical. Putting this together, we have already established Theorem 3.

Theorem 3. Given targets $(\boldsymbol{s}, \boldsymbol{y})$, assume that at subsystem i optimum solution $(\boldsymbol{\sigma}_{i*}, \boldsymbol{\zeta}_{i*}, \boldsymbol{x}_{i*})$, Jacobian $\partial \boldsymbol{\eta}_{i*}/\partial \boldsymbol{x}_i$ has full row rank. Then, all discrepancy terms $(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)$, $(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)$, and $(\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i)$ vanish if no design constraints g_{ik*} are critical.

Note that if Jacobian $\partial \boldsymbol{\eta}_{i*}/\partial \boldsymbol{x}_i$ does not have full row rank, satisfaction of the ith consistency constraint is not guaranteed even when subsystem i attains its minimum discrepancy.

One way to avoid trouble with multipliers when applying the KKT conditions is to modify both system and subsystem levels. The underlying idea is to replace the consistency constraints by other constraints that are not equalities or SSD. The following slightly strengthened version of Theorem 3 affords a specific way for doing that.

Theorem 4. Given targets $(\boldsymbol{s}, \boldsymbol{y})$, assume that all subsystems minimize their discrepancies d_i with no constraints other than satisfying their analysis equations. Then, targets $(\boldsymbol{s}, \boldsymbol{y})$ satisfy all consistency constraints if each Jacobian $\partial \boldsymbol{\eta}_i/\partial \boldsymbol{x}_i$ has full row rank.

The consequences are obvious: 1) Consistency constraints (3b) are no longer needed; 2) the system level needs to satisfy all design constraints instead, while optimizing the choice of targets; 3) individual subsystems perform their discipline-specific analyses plus their unconstrained discrepancy minimization based on the passed-down targets $(\boldsymbol{s}, \boldsymbol{y})$; and 4) all subsystems should specify their design constraints at the system level with their unconstrained optimum solutions. The modified bilevel optimization is carried out in Sec. IV.D and results in an additional interesting insight of the CO problem.

D. Postoptimality Sensitivity Derivatives

It is needed for solving the system-level optimization problem by a gradient-based algorithm to evaluate gradients of the consistency-constraint functions $d_{i*}(\boldsymbol{s}, \boldsymbol{y})$. These are sensitivity gradients of subsystem optima and in the literature[19–21] are often called with prefixes optimum or postoptimality to distinguish them from the usual gradients (with respect to design variables, such as $\boldsymbol{x}_i$, etc.) that are needed during the iterative optimization. An early and most common use of postoptimality sensitivity gradients is to estimate the cost on the objective optimum due to small changes in the constraint bounds (called shadow price in economics) by using Lagrange multipliers associated with the inequality constraints. Such a multiplier-based postoptimality sensitivity calculation has been extended and applied to problem parameters other than constraint bounds in engineering design.[19–21] In CO, such gradients are precisely the sensitivity derivatives of the optima of subsystem objectives d_{i*}, with respect to targets $(\boldsymbol{s}, \boldsymbol{y})$ as the problem parameters of the subsystems; see Eq. (4). Let us write them out more explicitly:

$$\frac{\partial d_{i*}}{\partial \boldsymbol{s}_i} = 2(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)^T \left[\frac{\partial \boldsymbol{\sigma}_{i*}}{\partial \boldsymbol{s}_i} - I\right] + 2(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)^T \frac{\partial \boldsymbol{\zeta}_{i*}}{\partial \boldsymbol{s}_i} + 2(\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i)^T \frac{\partial \boldsymbol{\eta}_{i*}}{\partial \boldsymbol{s}_i} \quad (11a)$$

$$\frac{\partial d_{i*}}{\partial \boldsymbol{z}_i} = 2(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)^T \frac{\partial \boldsymbol{\sigma}_{i*}}{\partial \boldsymbol{z}_i} + 2(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)^T \left[\frac{\partial \boldsymbol{\zeta}_{i*}}{\partial \boldsymbol{z}_i} - I\right] + 2(\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i)^T \frac{\partial \boldsymbol{\eta}_{i*}}{\partial \boldsymbol{z}_i} \quad (11b)$$

$$\frac{\partial d_{i*}}{\partial \boldsymbol{y}_i} = 2(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)^T \frac{\partial \boldsymbol{\sigma}_{i*}}{\partial \boldsymbol{y}_i} + 2(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)^T \frac{\partial \boldsymbol{\zeta}_{i*}}{\partial \boldsymbol{y}_i} + 2(\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i)^T \left[\frac{\partial \boldsymbol{\eta}_{i*}}{\partial \boldsymbol{y}_i} - I\right] \quad (11c)$$

where I is an identity matrix of the appropriate dimension. Because subsystem optimum solutions $(\boldsymbol{\sigma}_{i*}, \boldsymbol{\zeta}_{i*}, \boldsymbol{x}_{i*})$ are usually not available in closed form, sensitivity derivatives $\partial \boldsymbol{\sigma}_{i*}/\partial \boldsymbol{s}_i$, $\partial \boldsymbol{\zeta}_{i*}/\partial \boldsymbol{s}_i$, etc., require calculation of the Lagrange multipliers and second-order derivatives of the subsystem constraint functions[19] and, thus, incur a relatively significant computational cost.

Braun and Kroo[2] posed a new postoptimality sensitivity analysis (POSA), with a simple formula that did not even need Lagrange multipliers or second-order derivatives for use with the CO architecture as a very computationally inexpensive way of providing analytic gradients of the consistency constraints. It takes full advantage of the SSD nature of the consistency constraints. Specifically, they proposed that postoptimality sensitivity derivatives of the minimized objective d_{i*} with respect to problem parameters $(\boldsymbol{s}_i, \boldsymbol{z}_i, \boldsymbol{y}_i)$ be calculated simply by[2]

$$\frac{\partial d_{i*}}{\partial \boldsymbol{s}_i} = -2(\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i)^T, \qquad \frac{\partial d_{i*}}{\partial \boldsymbol{z}_i} = -2(\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i)^T, \ldots \quad (12)$$

as if subsystem optimum solutions $(\boldsymbol{\sigma}_{i*}, \boldsymbol{\zeta}_{i*}, \boldsymbol{x}_{i*})$ were independent of the problem parameters $(\boldsymbol{s}_i, \boldsymbol{z}_i, \boldsymbol{y}_i)$, that is, as if

$$\frac{\partial \boldsymbol{\sigma}_{i*}}{\partial \boldsymbol{s}_i} \equiv 0, \ldots, \qquad \frac{\partial \boldsymbol{\zeta}_{i*}}{\partial \boldsymbol{z}_i} \equiv 0, \ldots \quad (13)$$

in Eqs. (11a–11c). Their treatment of problem parameters $(\boldsymbol{s}, \boldsymbol{y})$ is inconsistent with the general understanding in the literature[19,20] when calculating postoptimality sensitivity derivatives, that is, Eq. (13) are actually invalid in general. (For specific examples, see Sec. IV.E). Nonetheless, it is amazing that the ingeniously simple formula (12) is mostly, though not completely, correct. In Sec. IV.E, we calculate the sensitivity derivatives of the consistency constraints first by the standard calculus, that is, using Eq. (11), and then by their POSA formulas, all based on the same closed-form optimum subsystem solutions, and then compare them.

IV. Analytical Case Studies

To gain specific qualitative insights of CO, we work out an example analytically. After obtaining optimum solutions of all subsystem problems in closed form, we will see specifically the following: 1) No system-level feasible targets can satisfy the KKT conditions. 2) The solution of the predecomposed problem can be approached only from outside of the system-level feasible regions and can only be closely approximated. 3) There exist no nonzero multipliers to satisfy the subsystem KKT conditions when consistency constraints are satisfied. 4) When both levels are modified to apply the KKT conditions without difficulty with multipliers, the resulting CO problem has the same solution as the predecomposed problem.

Consider the following simple example as our case study problem. Find scalars s, x_1, x_2:

$$\text{minimize} \quad J_{\text{sys}} = y_1^2 + 10y_2^2 \quad (14a)$$

subject to

$$s + x_1 \le 1 \quad (14b)$$

$$-s + x_2 \le -2 \quad (14c)$$

with (y_1, y_2) solved from

$$-2y_1 + y_2 = x_1 \tag{14d}$$

$$-y_1 + 2y_2 = x_2 \tag{14e}$$

where inequalities (14b) and (14c) are the design constraints of the disciplinary subsystems and Eqs. (14d) and (14e) are their coupled analysis equations. This problem has a unique solution[12]:

$$s^* = 18/11, \qquad x_1^* = -7/11, \qquad x_2^* = -4/11$$

$$y_1^* = 10/33, \qquad y_2^* = -1/33, \qquad J_{\text{sys}}^* = 1/9.9$$

This example, though symbolically different here, is the "convex quadratic programming" problem Alexandrov and Lewis[12] used to investigate and demonstrate convergence difficulties with CO. To get to their problem setting and solutions exactly, simply set $a_1 = -y_1$, $a_2 = y_2$, and half the system objective defined by Eq. (14a). In practical MDO problems, two different disciplines generally have different analysis equations. We switch the sign of a_1, that is, from a_1 to $-y_1$, consistently all over so that both subsystem analysis equations (14d) and (14e) at least do not look so symmetric as they were originally with both local design variables positively proportional to both subsystem outputs in exactly the same way.

Although it does not represent the large scale of practical MDO problems nor the complexity in realistic design constraints and analysis equations, this simple example does represent, at least symbolically, many essential characteristics of the general MDO problems. For instance, it has sharing of global design variables (through s) and intercoupling of analysis outputs (through y_1 and y_2). Kroo and Manning[9] modified it and thus provided a very similar one, but it then had no interdisciplinary coupling through either output y_1 or output y_2.

A. Subsystem-Level Discrepancy-Minimizing Solutions

Because each subsystem has only one local design variable, too few for satisfying a design constraint and an analysis equation simultaneously, the flexibility offered by the CO formulation is useful here. Just as in Ref. 12, assume that subsystem 1 uses variant σ_1 of shared design variable s and produces variant η_1 as output y_1, whereas subsystem 2 uses variant σ_2 of s and produces variant η_2 as y_2. Also assume, as in Ref.12, that subsystem 1 accepts y_2 exactly as passed down from the system level for input and subsystem 2 similarly accepts y_1 as is for input. In other words, $\zeta_1 \equiv z_1 = y_2$ and $\zeta_2 \equiv z_2 = y_1$. In the following, we derive the optimum solutions of these subsystem problems as functions of targets (s, y_1, y_2) passed down from the system level.

Subsystem 1

Minimize $d_1 = (\sigma_1 - s)^2 + (\eta_1 - y_1)^2$ subject to

$$\sigma_1 + x_1 \le 1 \tag{15a}$$

with η_1 solved from

$$-2\eta_1 + y_2 = x_1 \tag{15b}$$

where σ_1 and x_1 are the optimization variables. Define Lagrangian $F = d_1 + \mu_1 g_1$ where μ_1 denotes the Lagrange multiplier associated with design constraint $g_1 = \sigma_1 + x_1 - 1$.

Then the KKT conditions are 1) $\partial F/\partial\sigma_1 = 2(\sigma_1 - s) + \mu_1 = 0$ and 2) $\partial F/\partial x_1 = -(\eta_1 - y_1) + \mu_1 = 0$ where $\partial\eta_1/\partial\sigma_1 = 0$ and $\partial\eta_1/\partial x_1 = -\frac{1}{2}$ from analysis equation (15b) were used.

For case 0, $\mu_1 = 0$, the KKT conditions imply that $\sigma_1 = s$ and $\eta_1 = y_1$. From this and analysis equation (15b), we have $x_1 = -2y_1 + y_2$. Constraint (15a) then requires

$$s - 2y_1 + y_2 - 1 \le 0$$

For case 1, $\mu_1 > 0$, constraint (15a) must be critical; hence, $x_1 = -\sigma_1 + 1$. With analysis equation (15b), we then have $\sigma_1 = 2\eta_1 - y_2 + 1$. The KKT conditions imply that $2(\sigma_1 - s) = -(\eta_1 - y_1)$. From the last two equations, we readily obtain

$$\sigma_1 = (4s + 2y_1 - y_2 + 1)/5, \qquad \eta_1 = (2s + y_1 + 2y_2 - 2)/5$$

Finally, condition 2 requires that $0 < \eta_1 - y_1$, that is,

$$0 < s - 2y_1 + y_2 - 1$$

Given each value of parameters (s, y_1, y_2), this is a simple quadratic programming problem in standard form. The preceding is its optimum solution, provided that parameters (s, y_1, y_2) allow it to exist. Combining both cases 0 and 1, subsystem 1 optimum solution for $s - 2y_1 + y_2 \le 1$ is

$$\sigma_{1*} = s, \qquad \eta_{1*} = y_1, \qquad x_{1*} = -2y_1 + y_2 \tag{15c}$$

and for $1 < s - 2y_1 + y_2$ it is

$$\sigma_{1*} = (4s + 2y_1 - y_2 + 1)/5, \qquad \eta_{1*} = (2s + y_1 + 2y_2 - 2)/5$$

$$x_{1*} = 1 - \sigma_{1*} \tag{15d}$$

Subsystem 2

Minimize $d_2 = (\sigma_2 - s)^2 + (\eta_2 - y_2)^2$ subject to

$$-\sigma_2 + x_2 \le -2 \tag{16a}$$

with η_2 solved from

$$-y_1 + 2\eta_2 = x_2 \tag{16b}$$

The subsystem 2 optimum solution, derived in a similar manner, for $2 \le s + y_1 - 2y_2$ is given by

$$\sigma_{2*} = s, \qquad \eta_{2*} = y_2, \qquad x_{2*} = -y_1 + 2y_2 \tag{16c}$$

and for $s + y_1 - 2y_2 < 2$ it is

$$\sigma_{2*} = (4s - y_1 + 2y_2 + 2)/5, \qquad \eta_{2*} = (2s + 2y_1 + y_2 - 4)/5$$

$$x_{2*} = \sigma_{2*} - 2 \tag{16d}$$

These subsystem results are same as given in Ref. 12 except that they are now expressed more explicitly. Before proceeding, note that each subsystem optimum solution has two different closed forms for two different domains in the target space.

B. System-Level Consequences of Satisfying Consistency Constraints

Using closed-form solutions (15c) and (15d) and (16c) and (16d), we can express the subsystem-minimized discrepancies specifically in closed form as follows.

For $s - 2y_1 + y_2 \le 1$:

$$d_{1*} = 0 \tag{17a}$$

For $1 < s - 2y_1 + y_2$:

$$d_{1*} = (s - 2y_1 + y_2 - 1)^2/5 \tag{17b}$$

For $2 \le s + y_1 - 2y_2$:

$$d_{2*} = 0 \tag{18a}$$

For $s + y_1 - 2y_2 < 2$:

$$d_{2*} = (s + y_1 - 2y_2 - 2)^2/5 \tag{18b}$$

From Eqs. (17a) and (17b), consistency constraint 1 is satisfied if and only if targets (s, y_1, y_2) are such that $s - 2y_1 + y_2 \le 1$, and from Eqs. (18a) and (18b) consistency constraint 2 is satisfied if and only if the targets are such that $2 \le s + y_1 - 2y_2$. It also follows from Eqs. (17) and (18) that both consistency-constraint gradients $\partial d_{1*}/\partial(s, y_1, y_2)$ and $\partial d_{2*}/\partial(s, y_1, y_2)$ vanish whenever both consistency constraints are satisfied.

Let us continue to solve the system-level problem. Minimize

$$J_{\text{sys}} = y_1^2 + 10y_2^2 \tag{19a}$$

subject to

$$d_{1*} = 0, \qquad d_{2*} = 0 \tag{19b}$$

where s, y_1, and y_2 now are the optimization variables.

Define Lagrangian $F = y_1^2 + 10y_2^2 + \lambda_1 d_{1*} + \lambda_2 d_{2*}$ Then the KKT conditions are 1) $\partial F/\partial s = \lambda_1 \partial d_{1*}/\partial s + \lambda_2 \partial d_{2*}/\partial s = 0$, 2) $\partial F/\partial y_1 = 2y_1 + \lambda_1 \partial d_{1*}/\partial y_1 + \lambda_2 \partial d_{2*}/\partial y_1 = 0$, and 3) $\partial F/\partial y_2 = 20y_2 + \lambda_1 \partial d_{1*}/\partial y_2 + \lambda_2 \partial d_{2*}/\partial y_2 = 0$.

For case 00, $d_{1*} = 0$ and $d_{2*} = 0$. Only in both domains for (17a) and (18a) is satisfying both consistency constraints possible, and for such a combination, the constraint Jacobian vanishes. The KKT conditions simplify to $y_1 = 0$ and $y_2 = 0$. Substituting into these domain definitions yields a contradiction: $s \leq 1$ and $2 \leq s$. Therefore, no system-level feasible targets can satisfy even the KKT first-order conditions.

Note that objective gradient $\partial J_{\text{sys}}/\partial(s, y_1, y_2)$ even fails to vanish at optimum solution (s^*, y_1^*, y_2^*) of the predecomposed problem. Specifically,

$$\frac{\partial J_{\text{sys}}}{\partial s} = 0, \qquad \frac{\partial J_{\text{sys}}}{\partial y_1} = 2y_1^* = \frac{20}{33}, \qquad \frac{\partial J_{\text{sys}}}{\partial y_2} = 2y_2^* = -\frac{2}{33}$$

Thus, it is expected from the KKT conditions that no optimum system-level feasible solution exists.

To be system-level feasible, any candidate must be in both domains for (17a) and (18a), a wedge in the s–y_1–y_2 space; however, in this region, objective gradient $\partial J_{\text{sys}}/\partial(s, y_1, y_2)$ fails to vanish as required by KKT first-order conditions 1–3. Only outside this region can a consistency-constraint gradient be nonzero so that objective gradient $\partial J_{\text{sys}}/\partial(s, y_1, y_2)$ need not vanish, but any resulting first-order solution is not system-level feasible. Therefore, we may have to consider the limit of some infeasible first-order solutions as an approximation. Thus, let us continue with different cases of constraint violation.

For case 01, $d_{1*} = 0$ and $d_{2*} \neq 0$. Condition 1 becomes $\partial d_{2*}/\partial s = 2\lambda_2(s + y_1 - 2y_2 - 2)/5 = 0$, which implies $\lambda_2 = 0$. The remaining KKT conditions simplify to $y_1 = 0$ and $y_2 = 0$. The domain definitions for this combination then require $s \leq 1$. Infeasible first-order solution $(s, 0, 0)$ with $s \leq 1$ obviously is too far away from solution (s^*, y_1^*, y_2*).

For case 10, $d_{1*} \neq 0$ and $d_{2*} = 0$. Condition 1 becomes $\partial d_{1*}/\partial s = 2\lambda_1(s - 2y_1 + y_2 - 1)/5 = 0$, which implies $\lambda_1 = 0$. The remaining KKT conditions again simplify to $y_1 = 0$ and $y_2 = 0$. The domain definitions for this combination then require $2 \leq s$. Such an infeasible first-order solution, $(s, 0, 0)$ with $s \geq 2$, also is too far away from solution (s^*, y_1^*, y_2^*).

For case 11, $d_{1*} \neq 0$ and $d_{2*} \neq 0$. The KKT conditions become 1) $\lambda_1 A + \lambda_2 B = 0$, 2) $5y_1 - 2\lambda_1 A + \lambda_2 B = 0$, and 3) $50y_2 + \lambda_1 A - 2\lambda_2 B = 0$, where $A = s - 2y_1 + y_2 - 1$ and $B = s + y_1 - 2y_2 - 2$.

First, domain definitions for (17b) and (18b) require specifically that $A > 0$ and $B < 0$. Condition 1 in turn requires that both λ_1 and λ_2 must be nonzero and have the same sign. Now, combined with condition 1, conditions 2 and 3 simplify, respectively, to

$$5y_1 - 3\lambda_1 A = 0, \qquad 50y_2 + 3\lambda_1 A = 0$$

Together, they require

$$y_2 = -y_1/10 \tag{19c}$$

Upon substitution of Eq. (19c) into A and B, conditions 1 and 2 become

$$(\lambda_1 + \lambda_2)s - (2.1\lambda_1 - 1.2\lambda_2)y_1 - \lambda_1 - 2\lambda_2 = 0$$

$$-3\lambda_1 s + (5 + 6.3\lambda_1)y_1 + 3\lambda_1 = 0$$

From which

$$s = [(2.1\lambda_1 - 1.2\lambda_2)y_1 + \lambda_1 + 2\lambda_2]/(\lambda_1 + \lambda_2) \tag{19d}$$

$$y_1 = 3\lambda_1\lambda_2/(9.9\lambda_1\lambda_2 + 5\lambda_1 + 5\lambda_2) \tag{19e}$$

Finally,

$$A = 5\lambda_2/(9.9\lambda_1\lambda_2 + 5\lambda_1 + 5\lambda_2),$$

$$B = -5\lambda_1/(9.9\lambda_1\lambda_2 + 5\lambda_1 + 5\lambda_2)$$

The domain definitions then require $9.9\lambda_1 + 5\lambda_1/\lambda_2 + 5 > 0$ and $9.9\lambda_2 + 5\lambda_2/\lambda_1 + 5 > 0$.

This means that both Lagrange multipliers λ_1 and λ_2 can be any positive numbers, or, otherwise, any negative numbers such that $-9.9\lambda_1 < 5\lambda_1/\lambda_2 + 5$ and $-9.9\lambda_2 < 5\lambda_2/\lambda_1 + 5$.

Now, to gain a clearer, more specific insight, suppose $\lambda_1 = \lambda_2 = \lambda$. Then the domain definitions simply require that either $\lambda > 0$, or $0 > \lambda > -10/9.9$. In the case of positive λ, observe that

$$y_1 \to 10/33 = y_1^*, \qquad y_2 \to -1/33 = y_2^*$$

$$s \to 18/11 = s^*, \qquad \text{as} \quad \lambda \to \infty$$

that is, the optimum solution of the predecomposed can be closely approximated by system-level infeasible first-order solutions (19c–19e) approaching from outside the system-level feasible region by targets that are totally infeasible, having both $d_{1*}(s, y_1, y_2) \neq 0$ and $d_{2*}(s, y_1, y_2) \neq 0$.

C. Subsystem-Level Consequences of Satisfying Consistency Constraints

The gradient of the subsystem 1 design constraint, $\partial g_{1*}/\partial(\sigma_1, x_1) = (1, 1)$, is linearly independent. By Theorem 2, its multiplier must be zero: $\mu_1 = 0$. Similarly, the gradient of the subsystem 2 design constraint, $\partial g_{2*}/\partial(\sigma_2, x_2) = (-1, 1)$, is linearly independent, and by Theorem 2, its multiplier also must be zero, that is, $\mu_2 = 0$. Thus, when targets (s, y_1, y_2) are system-level feasible, there exists no nonzero multiplier μ_1 to satisfy the KKT1 conditions and no nonzero multiplier μ_2 to satisfy the KKT2 conditions.

D. Applying Sufficient Conditions for Satisfying Consistency Constraints

Jacobian $\partial\eta_i/\partial x_i$ of each subsystem has full row rank because $\partial\eta_1/\partial x_1 = -\frac{1}{2}$ by Eq. (15b) and $\partial\eta_2/\partial x_2 = \frac{1}{2}$ by Eq. (16b). Theorem 4 then suggests that both subsystems better be unconstrained instead. After dropping the design constraints, problems (15) and (16) change as follows.

For subsystem 1, minimize $d_1 = (\sigma_1 - s)^2 + (\eta_1 - y_1)^2$ with η_1 solved from $-2\eta_1 + y_2 = x_1$.

For subsystem 2, minimize $d_2 = (\sigma_2 - s)^2 + (\eta_2 - y_2)^2$ with η_2 solved from $-y_1 + 2\eta_2 = x_2$. Their unconstrained solutions are readily obtained to be

$$\sigma_{1*} = s, \qquad \eta_{1*} = y_1, \qquad x_{1*} = -2y_1 + y_2 \tag{20a}$$

$$\sigma_{2*} = s, \qquad \eta_{2*} = y_2, \qquad x_{2*} = -y_1 + 2y_2 \tag{20b}$$

All of the discrepancy terms now vanish and, hence, both consistency constraints are satisfied.

The system-level optimization should now satisfy design constraints instead of consistency constraints, whereas the subsystems should pass up their unconstrained optimum solutions to specify their design constraints. The modified system-level problem thus is as follows: minimize

$$J_{\text{sys}} = y_1^2 + 10y_2^2$$

subject to

$$s - 2y_1 + y_2 - 1 \leq 0 \tag{20c}$$

$$-s - y_1 + 2y_2 + 2 \leq 0 \tag{20d}$$

Constraints (20c) and (20d) represent the subsystem design constraints (15a) and (16a), after unconstrained subsystem solutions

(20a) and (20b) are substituted in. The system-level solution is readily found to be the same as that of the predecomposed problem:

$$s^* = 18/11 = 1.\overline{63}, \qquad y_1^* = 10/33 = 0.\overline{30}$$

$$y_2^* = -1/33 = -0.\overline{03}, \qquad J_{\text{sys}}^* = 1/9.9 = 0.\overline{10}$$

This modified system-level problem can be interpreted as the original having the consistency equality constraints (19b) replaced by the domain definitions for (17a) and (18a). Therefore, the preceding derivation is an analytical proof that the original CO problem after the modification has the same unique solution as the predecomposed problem. This clarifies that, in this example, CO does not cause convergence difficulty in iterative computation because of nonexistence or nonuniqueness of its solution. The difficulty is rather the general inability of KKT-based analysis and computation, such as NPSOL and other sequential guadratic programming algorithms, to deal with SSD-type equality constraints and is not really unique to CO. It is simply because the gradient of any SSD equality constraint necessarily vanishes when that very constraint is satisfied.

Recall that no system-level optimum solution can be found by applying the KKT conditions as usual. (See case 00 in Sec. IV.B.) Solving the modified system-level problem also can be interpreted as an example of applying Fritz John condition (6). Constraints (20c) and (20d) define those points at which the multiplier λ_0 associated with objective J_{sys} must be zero (because gradients of d_{1*} and d_{2*} vanish). Finding the minimum of objective J_{sys} directly over these points, that is, solving the modified system-level problem subject to constraints (20c) and (20d), gives a local optimum, which is also the global optimum for this problem.

E. Postoptimality Sensitivity Derivatives

Subsystem 1

Sensitivity of the subsystem 1 optimum variants to targets is available from Eqs. (15c) and (15d).

For $s - 2y_1 + y_2 \le 1$:

$$\frac{\partial \sigma_{1*}}{\partial s} = 1, \qquad \frac{\partial \sigma_{1*}}{\partial y_1} = 0, \qquad \frac{\partial \sigma_{1*}}{\partial y_2} = 0$$

$$\frac{\partial \eta_{1*}}{\partial s} = 0, \qquad \frac{\partial \eta_{1*}}{\partial y_1} = 1, \qquad \frac{\partial \eta_{1*}}{\partial y_2} = 0$$

For $1 < s - 2y_1 + y_2$:

$$\frac{\partial \sigma_{1*}}{\partial s} = \frac{4}{5}, \qquad \frac{\partial \sigma_{1*}}{\partial y_1} = \frac{2}{5}, \qquad \frac{\partial \sigma_{1*}}{\partial y_2} = -\frac{1}{5}$$

$$\frac{\partial \eta_{1*}}{\partial s} = \frac{2}{5}, \qquad \frac{\partial \eta_{1*}}{\partial y_1} = \frac{1}{5}, \qquad \frac{\partial \eta_{1*}}{\partial y_2} = \frac{2}{5}$$

Recall that $\zeta_1 \equiv z_1 = y_2$.

Sensitivity derivatives of d_{1*} (s, y_1, y_2) to targets calculated according to Eqs. (11) are all zero for $s - 2y_1 + y_2 \le 1$. Therefore, consistency-constraint gradient $\partial d_{1*}/\partial(s, y_1, y_2)$ vanishes at all such targets. On the other hand, for $1 < s - 2y_1 + y_2$, sensitivity derivatives of $d_{1*}(s, y_1, y_2)$ to targets also calculated according to Eqs. (11) are all nonzero:

$$\frac{\partial d_{1*}}{\partial s} = 2(\sigma_{1*} - s)\left[\frac{\partial \sigma_{1*}}{\partial s} - 1\right] + 2(\eta_{1*} - y_1)\frac{\partial \eta_{1*}}{\partial s}$$

$$= -\frac{2(\sigma_{1*} - s)}{5} + \frac{4(\eta_{1*} - y_1)}{5} = \frac{2(s - 2y_1 + y_2 - 1)}{5} > 0 \tag{21a}$$

$$\frac{\partial d_{1*}}{\partial z_1} = \frac{\partial d_{1*}}{\partial y_2} = 2(\sigma_{1*} - s)\frac{\partial \sigma_{1*}}{\partial y_2} + 2(\eta_{1*} - y_1)\frac{\partial \eta_{1*}}{\partial y_2}$$

$$= -\frac{2(\sigma_{1*} - s)}{5} + \frac{4(\eta_{1*} - y_1)}{5} = \frac{2(s - 2y_1 + y_2 - 1)}{5} > 0 \tag{21b}$$

$$\frac{\partial d_{1*}}{\partial y_1} = 2(\sigma_{1*} - s)\frac{\partial \sigma_{1*}}{\partial y_1} + 2(\eta_{1*} - y_1)\left[\frac{\partial \eta_{1*}}{\partial y_1} - 1\right]$$

$$= \frac{4(\sigma_{1*} - s)}{5} - \frac{8(\eta_{1*} - y_1)}{5} = -\frac{4(s - 2y_1 + y_2 - 1)}{5} < 0 \tag{21c}$$

Subsystem 2

Sensitivity of the subsystem 2 optimum variants to targets is available from Eqs. (16c) and (16d).

For $2 \le s + y_1 - 2y_2$:

$$\frac{\partial \sigma_{2*}}{\partial s} = 1, \qquad \frac{\partial \sigma_{2*}}{\partial y_1} = 0, \qquad \frac{\partial \sigma_{2*}}{\partial y_2} = 0$$

$$\frac{\partial \eta_{2*}}{\partial s} = 0, \qquad \frac{\partial \eta_{2*}}{\partial y_1} = 0, \qquad \frac{\partial \eta_{2*}}{\partial y_2} = 1$$

For $s + y_1 - 2y_2 < 2$:

$$\frac{\partial \sigma_{2*}}{\partial s} = \frac{4}{5}, \qquad \frac{\partial \sigma_{2*}}{\partial y_1} = -\frac{1}{5}, \qquad \frac{\partial \sigma_{2*}}{\partial y_2} = \frac{2}{5}$$

$$\frac{\partial \eta_{2*}}{\partial s} = \frac{2}{5}, \qquad \frac{\partial \eta_{2*}}{\partial y_1} = \frac{2}{5}, \qquad \frac{\partial \eta_{2*}}{\partial y_2} = \frac{1}{5}$$

Recall $\zeta_2 \equiv z_2 = y_1$.

Sensitivity derivatives of d_{2*} (s, y_1, y_2) to targets calculated according to Eqs. (11) are all zero for $2 \le s + y_1 - 2y_2$. Therefore, consistency-constraint gradient $\partial d_{2*}/\partial(s, y_1, y_2)$ vanishes at all such targets. On the other hand, for $s + y_1 - 2y_2 < 2$, sensitivity derivatives of d_{2*} (s, y_1, y_2) to targets also calculated according to Eqs. (11) are all nonzero:

$$\frac{\partial d_{2*}}{\partial s} = 2(\sigma_{2*} - s)\left[\frac{\partial \sigma_{2*}}{\partial s} - 1\right] + 2(\eta_{2*} - y_2)\frac{\partial \eta_{2*}}{\partial s}$$

$$= -\frac{2(\sigma_{2*} - s)}{5} + \frac{4(\eta_{2*} - y_2)}{5} = \frac{2(s + y_1 - 2y_2 - 2)}{5} > 0 \tag{22a}$$

$$\frac{\partial d_{2*}}{\partial z_2} = \frac{\partial d_{2*}}{\partial y_1} = 2(\sigma_{2*} - s)\frac{\partial \sigma_{2*}}{\partial y_1} + 2(\eta_{2*} - y_2)\frac{\partial \eta_{2*}}{\partial y_1}$$

$$= -\frac{2(\sigma_{2*} - s)}{5} + \frac{4(\eta_{2*} - y_2)}{5} = \frac{2(s + y_1 - 2y_2 - 2)}{5} > 0 \tag{22b}$$

$$\frac{\partial d_{2*}}{\partial y_2} = 2(\sigma_{2*} - s)\frac{\partial \sigma_{2*}}{\partial y_2} + 2(\eta_{2*} - y_2)\left[\frac{\partial \eta_{2*}}{\partial y_2} - 1\right]$$

$$= \frac{4(\sigma_{2*} - s)}{5} - \frac{8(\eta_{2*} - y_2)}{5} = -\frac{4(s + y_1 - 2y_2 - 2)}{5} < 0 \tag{22c}$$

Because analytic solutions of the subsystem problems are available in closed form, it is instructive to derive some specific qualitative insights of POSA formula Eq. (12). By POSA, we easily get $\partial d_{1*}/\partial y_2 \equiv 0$ because d_1 has no discrepancy term explicitly involving y_2. Recall the definition of d_1 in Sec. IV.A. Evidently, compared to Eq. (21b), POSA does not provide optimum sensitivity derivative $\partial d_{1*}/\partial y_2$ correctly when it is nonzero. Similarly, by POSA, we again get $\partial d_{2*}/\partial y_1 \equiv 0$. Compared to Eq. (22b), POSA does not provide optimum sensitivity derivative $\partial d_{2*}/\partial y_1$ correctly, either. POSA provides other sensitivity derivatives correctly, such as the end results in Eqs. (21a) and (21c) and (22a) and (22c), even though the calculation is quite different and much simpler. Inasmuch as it is incorrect on two components when nonzero, POSA overall is still not adequate in providing nonvanishing analytic gradients of subsystem optima d_{1*} and d_{2*}.

V. Computational Case Studies

Alexandrov and Lewis[12] demonstrated numerically that CO had serious convergence difficulty due to vanishing consistency-constraint Jacobian when they applied NPSOL. To investigate the convergence difficulty in a different way, we compute CO solutions of the same problem described in Sec. IV by applying, instead of equality constraints, penalties on minimized discrepancies and using, instead of NPSOL, DOMIN[22] for system-level optimization and CONMIN[23] for each subsystem optimization. In other words, we examine CO from different viewpoints. (DOMIN is unconstrained, whereas CONMIN is based on usable-feasible directions; neither depends on Lagrange multipliers or the KKT conditions.)

General theoretical results in Sec. III.A and specific analytical results in Sec. IV.B imply that 1) minimized discrepancies d_{i*} should be kept from constraining to equal zero in the early part of iteration to prevent their gradients from vanishing prematurely and 2) the system-level solution should be approached from outside the system-level feasible region. These mean that the penalty-function method is a suitable vehicle at least for investigating the convergence difficulty.

Recently, DeMiguel and Murray[11] proposed that, before applying the penalty-function method to CO, discrepancies d_i be defined by sum of absolute, instead of squared, discrepancy terms for subsystem optimization. Minimized values of such absolute discrepancies would then be totaled, without being squared first, to form a penalty function. Accordingly, nonsmoothness and multiple local solutions would then become new difficulties to deal with; however, bundle methods were known to behave poorly in practice. Therefore, they proposed such modified subsystem problems be perturbed by introducing barrier, that is, interior penalty, terms and that each such perturbed problem be solved repeatedly for a decreasing sequence of barrier parameters until all barrier parameters finally could approach zero. The need to deal with the ill conditioning inherently associated with barrier functions became an additional difficulty.[11]

In this paper, we not only choose the usual well-established penalty-function method, still using the SSD definition of discrepancies, but also explore the potential enhancements. The penalty-function approach usually needs increasing an penalty to drive approximate solutions close to exact solutions. In enhanced applications, we step up the penalty factor p progressively. The reported results correspond to the last penalty factor (within a given limit) that has caused a noticeable change in the computed solution.

As shown at the end of Sec. IV.E, POSA unfortunately was inadequate in providing analytic gradients of the consistency constraints for this study problem. So that the inadequacy of POSA introduces no additional convergence difficulty, for this investigation we decide to use finite difference gradients in both system-level and subsystem-level optimization, even though we prefer analytic gradients. The system level is, thus, expected to make many calls (at additional cost) for subsystem-level optimization just for computing the finite difference gradients.

Computational results of gradient-based optimization are often dependent on the starting points. We select five quite different points for the system-level optimization to test its convergence: One is system-level feasible but four are not. Each is rather arbitrary, although each is also particular in a way according to analytical results (17) and (18). Specifically, starting point 1 ($s=-3$, $y_1=3$, $y_2=-3$) satisfies both constraints $d_{1*}=0$ and $d_{2*}=0$; starting point 2 ($s=y_1=y_2=0$) satisfies only $d_{1*}=0$ but not $d_{2*}=0$; starting point 3 ($s=1.1111$, $y_1=-1.1111$, $y_2=-2.2222$) does not satisfy $d_{1*}=0$ but does $d_{2*}=0$; starting point 4 ($s=3.3333$, $y_1=-2.2222$, $y_2=1.1111$) satisfies neither $d_{1*}=0$ nor $d_{2*}=0$; and starting point 5 ($s=1$, $y_1=-1$, $y_2=1$) also satisfies neither $d_{1*}=0$ nor $d_{2*}=0$.

Note that both consistency-constraint gradients have vanished at point 1 and that one of them has vanished at point 2 or 3. None of these starting points are near exact solution (s^*, y_1^*, y_2^*) because in practice we often are not so fortunate anyway. All subsystems start from zero: $\sigma_i=x_i=0$, $i=1, 2$.

Alexandrov and Lewis[12] observed the worst sort of behavior when starting from a system-level feasible point. Starting from point 1, NPSOL terminated at $s=-2.806$, $y_1=5.658$, $y_2=0.301$, excessively far from exact solution (s^*, y_1^*, y_2^*), having a large componentwise miss of -4.442 in s, 5.355 in y_1, and 0.3313 in y_2. Let us calculate the error relative to the exact solution by

$$\frac{\sqrt{(s-s^*)^2+\left(y_1-y_1^*\right)^2+\left(y_2-y_2^*\right)^2}}{\sqrt{s^{*2}+y_1^{*2}+y_2^{*2}}} \tag{23}$$

Then, the error of their solution is 418.49%. According to them, both consistency constraints were satisfied in all iterations from the beginning to the end. Thus, the consistency-constraint Jacobian was identically zero for all of these iterations, and POSA would have provided such (identically zero) gradients adequately. (They computed sensitivity derivatives for system-level optimization via POSA and those for subsystem-level optimization analytically.)

Starting from point 5, NPSOL was reported[12] to have found the solution with no iterations in which the consistency constraints were satisfied. In other words, no consistency constraint gradients vanished, although POSA would not have provided the constraint gradients adequately. (One component of each gradient provided by POSA would vanish erroneously consistently.) This makes it rather interesting also to try this starting point on a combination of finite difference gradients, the penalty-function method, and different optimization algorithms.

We try three different CO formulations for the same predecomposed problem: the original plus two extremes in the subsystem level.

A. Formulation 1

First, consider the CO formulation with subsystem problems formulated by Eqs. (15) and (16).

Basic Application 1

The penalty-function method is applied in the usual manner, each consistency constraint function d_{i*} being squared before summing to form a quadratic penalty function with factor p. The system level, thus, minimizes

$$y_1^2+10y_2^2+p\left(d_{1*}^2+d_{2*}^2\right) \tag{24}$$

A heavy penalty, $p=1000$, is applied immediately in a single step.

The results in Table 1 resemble what Alexandrov and Lewis obtained with equality constraints per CO formulation in that the solutions starting from points 1 and 5 are the worst and the best, respectively, and in that the effects of vanishing consistency-constraint gradients are very similar. Starting from point 1, with both constraint gradients already vanished, the result is a large miss in all components. Two misses are smaller in magnitude (marked with a v on the right of the numbers) than in Ref. 12, 4.310 vs 4.442 in

Table 1 Basic application 1: heavy, quadratic, variant

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		1000.00
End	−2.6739	2.8131	−0.7258	13.18	1000.00
Miss	−4.3102v	2.5100v	−0.6955^		
Error	302.56% Z1				
Start	0.0000	0.0000	0.0000		1000.00
End	2.0315	0.4766	0.0057	0.23	1000.00
Miss	0.3951	0.1736	0.0360		
Error	26.02%				
Start	1.1111	−1.1111	−2.2222		1000.00
End	0.0950	0.5586	−0.5096	2.91	1000.00
Miss	−1.5414	0.2556	−0.4793		
Error	98.19% Z2				
Start	3.3333	−2.2222	1.1111		1000.00
End	1.3336	0.4342	−0.0391	0.20	1000.00
Miss	−0.3028	0.1312	−0.0088		
Error	19.83% A2				
Start	1.0000	−1.0000	1.0000		1000.00
End	1.9110	0.3870	−0.0344	0.16	1000.00
Miss	0.2747	0.0840	−0.0041		
Error	17.26% A1				

Table 2 Basic application 2: heavy, quadratic, local

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		1000.00
End	1.8404	0.3363	−0.0616	0.15	1000.00
Miss	0.2041	0.0333	−0.0313		
Error	12.56% A2				
Start	0.0000	0.0000	0.0000		1000.00
End	2.5740	0.7732	0.1392	0.79	1000.00
Miss	0.9376	0.4702	0.1695		
Error	63.83% Z1				
Start	1.1111	−1.1111	−2.2222		1000.00
End	1.8894	0.4298	0.0470	0.21	1000.00
Miss	0.2531	0.1268	0.0773		
Error	17.63%				
Start	3.3333	−2.2222	1.1111		1000.00
End	1.6159	0.2720	−0.0350	0.09	1000.00
Miss	−0.0205	−0.0310	−0.0047		
Error	2.25% A1				
Start	1.0000	−1.0000	1.0000		1000.00
End	1.9338	0.4192	−0.0076	0.18	1000.00
Miss	0.2974	0.1162	0.0227		
Error	19.23% Z2				

s and 2.510 vs 5.355 in y_1, but one is larger (marked with a caret) 0.6955 vs 0.3313 in y_2. The error, also calculated by Eq. (23), is 302.56%, which is as bad as what Alexandrov and Lewis obtained with NPSOL. It is the worst result (marked Z1) in Table 1.

On the other hand, starting from point 5, with both constraint gradients not yet vanished, the approximate solution appears to have been able to get fairly close to the exact. The error is 17.26%, the best (marked A1) in Table 1. The approximate solution starting from point 4, where neither consistency-constraint gradients has yet vanished, is similar to that starting from point 5. It is the second best (marked A2), with an error of 19.83%.

Basic Application 2

Each subsystem problem in basic application 1 has variants σ_i and η_i placed ahead of local design variable x_i in the vector of optimization variables. Table 2 reports the results of placing x_i before both σ_i and η_i.

The solution starting from point 1 is better than that in Table 1. The error is now 12.56%, impressively smaller than 302.56%. Both the largest and the smallest errors among the five starting points decrease, too. The worst is 63.83%, almost five times smaller than the worst in Table 1. A different order in the optimization variables can make a difference. The reason is unclear. It may depend on the specific optimization package, namely, CONMIN, used. We point it out so that other researchers may be aware of the effect.

Enhanced Application 1

Tables 3 lists results from repeating basic application 2 but applying penalty factor p in two successive steps, initially $p = 100$ then $p = 1000$, and using a warm start of DOMIN after the initial step. DOMIN would take data from the previous run for initializing the optimization. The largest error is reduced from 63.83 to 40.02%.

Enhanced Application 2

Results in Table 3 indicate that lighter initial penalty can be beneficial in moving approximate solutions closer to the exact. Table 4 has results from repeating enhanced application 1 with even lighter initial penalty and even slower increase, that is, beginning with only $p = 10$ and increasing only by $\sqrt{(10)}$ up to $p \le 500$. In addition, discrepancies d_{i*} are not resquared, that is, Eq. (24) is replaced by linear penalty

$$y_1^2 + 10y_2^2 + p(d_{1*} + d_{2*}) \tag{25}$$

(Both d_{1*} and d_{2*} are already nonnegative and, thus, sufficient for forming a penalty function.[24]) The largest error is 29.73% and the smallest now only 0.61%.

Table 3 Enhanced application 1: lighter, quadratic

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		100.00
End	1.6446	0.2781	−0.0069	0.08	1000.00
Error	2.12% A1				
Start	0.0000	0.0000	0.0000		100.00
End	2.2760	0.4824	0.0190	0.24	100.00
Error	40.02% Z1				
Start	1.1111	−1.1111	−2.2222		100.00
End	1.6311	0.1979	−0.0181	0.04	100.00
Error	6.37% A2				
Start	3.3333	−2.2222	1.1111		100.00
End	1.5335	0.1801	−0.0284	0.04	100.00
Error	9.63% Z2				
Start	1.0000	−1.0000	1.0000		100.00
End	1.4847	0.3365	−0.0308	0.12	1000.00
Error	9.33%				

Table 4 Enhanced application 2: slow, linear

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		10.00
End	1.1560	0.3885	−0.1135	0.28	10.00
Error	29.73% Z1				
Start	0.0000	0.0000	0.0000		10.00
End	1.5936	0.2913	−0.0307	0.09	10.00
Error	2.66%				
Start	1.1111	−1.1111	−2.2222		10.00
End	1.4089	0.4301	−0.0684	0.23	316.23
Error	15.82% Z2				
Start	3.3333	−2.2222	1.1111		10.00
End	1.6522	0.2944	−0.0686	0.13	100.00
Error	2.54% A2				
Start	1.0000	−1.0000	1.0000		10.00
End	1.6348	0.3044	−0.0204	0.10	10.00
Error	0.61% A1				

Table 5 Enhanced application 3: slow, linear, rooted

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		10.00
End	1.6100	0.2605	−0.0600	0.10	31.62
Error	3.50%				
Start	0.0000	0.0000	0.0000		10.00
End	1.9281	0.4521	−0.0547	0.23	316.23
Error	19.74% Z1				
Start	1.1111	−1.1111	−2.2222		10.00
End	1.6393	0.2966	−0.0324	0.10	316.23
Error	0.44% A1				
Start	3.3333	−2.2222	1.1111		10.00
End	1.6649	0.3063	−0.0201	0.10	31.62
Error	1.83% A2				
Start	1.0000	−1.0000	1.0000		10.00
End	1.5484	0.2409	−0.0526	0.09	10.00
Error	6.61% Z2				

A simple explanation follows. When discrepancy d_{i*} is less than 1, squaring them has the effect of reducing the impact of the penalty. For instance, when $d_{i*} = 2$, squaring it emphasizes the impact because $d_{1*}^2 = 4 > 2$. On the other hand, when $d_{i*} = 0.2$, squaring it deemphasizes the impact because $d_{1*}^2 = 0.04 < 0.2$, reducing the effect of penalty by five times.

Enhanced Application 3

Table 5 results from continuing the same setup as enhanced application 2 except that each subsystem minimizes, instead of d_i, the square root of d_i (but still returns d_{i*} to the system level). The largest error is further reduced to 19.74%. The 0.44% approximate solution now looks quite like the exact.

A likely reason is subsystem optimization compares very small objective values, considerably below 1, for termination. Suppose

Table 6 Enhanced application 4: slow, linear, rooted, weighted

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		1.00
End	1.6563	0.2921	−0.0411	0.10	2.00
Error	1.51% Z1				
Start	0.0000	0.0000	0.0000		1.00
End	1.6401	0.3145	−0.0248	0.11	32.00
Error	0.80% Z2				
Start	1.1111	−1.1111	−2.2222		1.00
End	1.6393	0.2964	−0.0336	0.10	16.00
Error	0.48%				
Start	3.3333	−2.2222	1.1111		1.00
End	1.6383	0.2982	−0.0291	0.10	16.00
Error	0.32% A1				
Start	1.0000	−1.0000	1.0000		1.00
End	1.6308	0.3081	−0.0285	0.10	32.00
Error	0.47% A2				

the absolute difference between variants and targets, for example, between η_i and y_i, is 0.03. Assume for sake of argument that other discrepancy terms are negligible. Then, d_i is $\approx 0.03^2$. One criterion used by CONMIN to terminate is when the absolute difference with the previous objective is smaller than the default tolerance of 0.001. If the previous discrepancy is 1.8×10^{-3}, then CONMIN will terminate because $1.8 \times 10^{-3} - 0.03^2 = 0.0009 < 0.001$. Now, suppose the objective is the square root of d_i instead. Then, the current objective is now ≈ 0.03, and the difference with the previous objective is now $(1.8 \times 10^{-3})^{1/2} - 0.03 = 0.012426 > 0.001$. Thus, CONMIN will not terminate yet.

Enhanced Application 4

Table 6 lists results of continuing the same setup as enhanced application 3 except for two modifications: 1) multiplying the second term in discrepancy d_1 by 10 and that of discrepancy d_2 by 100 and 2) starting the penalty with $p = 1$ and increasing only by twofold until $p = 32$. Specifically, discrepancy d_1 is modified to weight the error in output y_1: $d_1 = (\sigma_1 - s)^2 + 10(\eta_1 - y_1)^2$. Similarly, discrepancy d_2 is to weight the error in output y_2: $d_2 = (\sigma_2 - s)^2 + 100(\eta_2 - y_2)^2$, where an extra factor of 10 reflects the relative importance of y_2 over y_1 in system-level objective J_{sys}. (By the same token, the error in s receives no weight increase.) This application is equivalent to enhanced application 3 insofar as initial penalty on constraint d_{1*} is concerned when $(\sigma_1 - s)^2$ is negligible compared to $(\eta_1 - y_1)^2$.

The worst approximate solution has an error of only 1.51% and the best, 0.32%!

B. Formulation 2

Next assume that each subsystem takes the least freedom in using or producing variants of the targets. Each subsystem accepts the shared design variable and the other subsystem's output as they are passed down from the system level. The only variants now are η_1 of y_1 that subsystem 1 produces as output and η_2 of y_2 that subsystem 2 produces as output. Subsystem problems (15) and (16) now reduce to the following.

For subsystem 1, minimize

$$d_1 = (\eta_1 - y_1)^2 \tag{26a}$$

subject to

$$s + x_1 \le 1 \tag{26b}$$

with η_1 solved from

$$-2\eta_1 + y_2 = x_1 \tag{26c}$$

For subsystem 2, minimize

$$d_2 = (\eta_2 - y_2)^2 \tag{27a}$$

subject to

$$-s + x_2 \le -2 \tag{27b}$$

Table 7 Basic application, formulation 2: heavy, quadratic

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		1000.00
End	1.6125	0.2501	−0.0250	0.07	1000.00
Error	3.50%				
Start	0.0000	0.0000	0.0000		1000.00
End	1.6125	0.2501	−0.0250	0.07	1000.00
Error	3.50%				
Start	1.1111	−1.1111	−2.2222		1000.00
End	1.6125	0.2501	−0.0250	0.07	1000.00
Error	3.50%				
Start	3.3333	−2.2222	1.1111		1000.00
End	1.6125	0.2501	−0.0250	0.07	1000.00
Error	3.50%				
Start	1.0000	−1.0000	1.0000		1000.00
End	1.6125	0.2501	−0.0250	0.07	1000.00
Error	3.50%				

Table 8 Enhanced application, formulation 2: slow, linear

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		10.00
End	1.6366	0.3028	−0.0297	0.10	316.23
Error	0.04%				
Start	0.0000	0.0000	0.0000		10.00
End	1.6366	0.3029	−0.0296	0.10	316.23
Error	0.04%				
Start	1.1111	−1.1111	−2.2222		10.00
End	1.6364	0.3026	−0.0299	0.10	316.23
Error	0.04%				
Start	3.3333	−2.2222	1.1111		10.00
End	1.6364	0.3027	−0.0298	0.10	316.23
Error	0.04%				
Start	1.0000	−1.0000	1.0000		10.00
End	1.6366	0.3028	−0.0297	0.10	316.23
Error	0.04%				

with η_2 solved from

$$-y_1 + 2\eta_2 = x_2 \tag{27c}$$

Subsystem 1 has only x_1 as optimization variable and subsystem 2 only x_2.

Basic Application

The penalty-function method is applied as basic application 1 but with discrepancy d_{i*} in Eq. (24) redefined by Eqs. (26a) and (27a). The results are given in Table 7. All approximate solutions have an error of 3.5%.

Enhanced Application

Results in Table 8 are from repeating enhanced application 2 with discrepancy d_{i*} in Eq. (25) redefined by Eqs. (26a) and (27a). All solutions have an error of only 0.04%.

C. Formulation 3

Finally assume that each subsystem takes the full freedom of using or producing variants of the targets, similar to examples shown in Ref.13. In addition to the variants in formation 1, subsystem 1 now uses variant ζ_1 of y_2 for input and subsystem 2 variant ζ_2 of y_1 for input. Because $z_1 = y_2$ and $z_2 = y_1$, problems (15) and (16) now expand as follows.

For subsystem 1, minimize

$$d_1 = (\sigma_1 - s)^2 + (\zeta_1 - y_2)^2 + (\eta_1 - y_1)^2 \tag{28a}$$

subject to

$$\sigma_1 + x_1 \le 1 \tag{28b}$$

with η_1 solved from

$$-2\eta_1 + \zeta_1 = x_1 \tag{28c}$$

Table 9 Basic application, formulation 3: heavy, quadratic

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		1000.00
End	−1.4609	1.7164	−0.6647	7.36	1000.00
Error	208.06% Z1				
Start	0.0000	0.0000	0.0000		1000.00
End	1.7954	−0.6482	−1.5270	23.74	1000.00
Error	106.97% Z2				
Start	1.1111	−1.1111	−2.2222		1000.00
End	2.4314	0.6278	−0.0265	0.40	1000.00
Error	51.60%				
Start	3.3333	−2.2222	1.1111		1000.00
End	1.7393	0.3340	0.0507	0.14	1000.00
Error	8.08% A1				
Start	1.0000	−1.0000	1.0000		1000.00
End	2.0606	0.4629	−0.0301	0.22	1000.00
Error	27.24% A2				

Table 10 Enhanced application, formulation 3: slow, linear

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		10.00
End	1.6740	0.3262	−0.0142	0.11	100.00
Error	2.83% A1				
Start	0.0000	0.0000	0.0000		10.00
End	1.7776	0.3753	−0.0140	0.14	100.00
Error	9.58% Z1				
Start	1.1111	−1.1111	−2.2222		10.00
End	1.6777	0.3307	−0.0099	0.11	31.62
Error	3.23% A2				
Start	3.3333	−2.2222	1.1111		10.00
End	1.5640	0.2544	−0.0633	0.10	31.62
Error	5.60%				
Start	1.0000	−1.0000	1.0000		10.00
End	1.5668	0.2381	−0.0842	0.13	100.00
Error	6.57% Z2				

For subsystem 2, minimize

$$d_2 = (\sigma_2 - s)^2 + (\zeta_2 - y_1)^2 + (\eta_2 - y_2)^2 \tag{29a}$$

subject to

$$-\sigma_2 + x_2 \le -2 \tag{29b}$$

with η_2 solved from

$$-\zeta_2 + 2\eta_2 = x_2 \tag{29c}$$

Subsystem 1 has σ_1, ζ_1, and x_1 as optimization variables and subsystem 2 has σ_2, ζ_2, and x_2.

Basic Application

The penalty-function method is applied as basic application 1 but with discrepancy d_{i*} in Eq. (24) redefined by Eqs. (28a) and (29a). The results are given in Table 9. Compared to Table 1, both largest and smallest errors are somewhat reduced.

Enhanced Application

Results in Table 10 are from repeating enhanced application 2 with discrepancy d_{i*} in Eq. (25) redefined by Eqs. (28a) and (29a). Compared to Table 4, the largest error is reduced, but the smallest increased.

D. Discussions

Formulation 1

Table 1 shows that the approximate solution starting from point 1 is the worst, just as in Ref. 12, although it is closer to the exact (solution of the predecomposed problem) than the would-be exact solution Alexandrov and Lewis[12] obtained through equality constraints according to the CO formulation. It is possible to improve the computational results when applying the penalty-function method. Four enhanced applications can improve the approximate solutions over those obtained from the usual application (basic application 1 or 2). For instance, approximate solution starting from point 1 can be improved from having an error of 302.56% to 1.51% and that starting from point 5 from having an error of 17.26% to 0.47%. One enhanced application can improve the worst of the solutions starting from different points to having error of only 1.51% and the best to having error of only 0.32%.

Although the penalty-function method may suffer from practically the same convergence difficulty when consistency-constraint gradients vanish, it can mitigate the difficulty by postponing or slowing down their disappearance. From the penalty viewpoint, CO with equality constraints on minimized discrepancies essentially is the case where a too heavy penalty on the discrepancies is imposed at the beginning of iteration and hence a lockup is enforced too soon. Thus, one might as well tradeoff convergence difficulty in equality-constrained optimization by using an enhanced penalty-function method at the expense of additional computation.

We have shown that approximate solutions computed by the penalty-function method can be reasonably close to the exact despite possible difficulty due to the vanishing of consistency-constraint gradients. We have achieved it, not by starting the iterative computation from nearly the exact solution, nor by searching for the right starting point, but rather by trying enhancements on the computational method itself. In other words, we do not recommend applying the penalty-function method to CO in the usual way as in basic application 1 or 2; we recommend considering enhanced applications 2–4.

Light–gradual–warm penalty adjustment and weighting individual discrepancy terms make the major enhancements, but using square roots and not resquaring each contribute significantly, too. For comparison, Tables 11 and 12 show results of using the same penalty adjustment and the same weighting as enhanced application 4 but otherwise repeating enhanced applications 1 and 2, respectively. Not resquaring reduces the largest error from 9.53% in

Table 11 Enhanced application 1a: slow, quadratic, weighted

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		1.00
End	1.6383	0.2113	−0.0250	0.05	16.00
Error	5.52% A1				
Start	0.0000	0.0000	0.0000		1.00
End	1.6646	0.1667	−0.0167	0.03	4.00
Error	8.40%				
Start	1.1111	−1.1111	−2.2222		1.00
End	1.6300	0.1723	−0.0192	0.03	4.00
Error	7.89% A2				
Start	3.3333	−2.2222	1.1111		1.00
End	1.5079	0.2416	−0.0552	0.09	32.00
Error	8.69% Z2				
Start	1.0000	−1.0000	1.0000		1.00
End	1.5841	0.1553	−0.0050	0.02	2.00
Error	9.53% Z1				

Table 12 Enhanced application 2a: slow, linear, weighted

Test	s	y_1	y_2	J_{sys}	p
Start	−3.0000	3.0000	−3.0000		1.00
End	1.6296	0.2919	−0.0440	0.10	16.00
Error	1.14% A1				
Start	0.0000	0.0000	0.0000		1.00
End	1.5992	0.2605	−0.0398	0.08	2.00
Error	3.44%				
Start	1.1111	−1.1111	−2.2222		1.00
End	1.6890	0.2846	0.0045	0.08	1.00
Error	3.95% Z2				
Start	3.3333	−2.2222	1.1111		1.00
End	1.6092	0.2671	−0.0554	0.10	4.00
Error	3.10% A2				
Start	1.0000	−1.0000	1.0000		1.00
End	1.7196	0.2023	−0.0149	0.04	1.00
Error	7.90% Z1				

Table 11 to 7.90% in Table 12 and similarly the smallest from 5.52% to 1.14%. Next, a comparison of Tables 12 and 6 shows that using square roots reduces the largest error from 7.90% to 1.51% and the smallest from 1.14% to 0.32%.

It is rather straightforward to extend these enhanced applications to large real engineering MDO problems, although three techniques need further development: light initial penalty, gradual increase in penalty factor, and weighting of individual discrepancy terms. We are still working on them and will conduct more thorough investigation with more realistic CO problems later for their generalization or adaptive combination. Computational efficiency then will be compared to standard CO approaches. Right now they are simply some useful tips for tailoring the general penalty-function method for more intelligent application to CO problems.

Different Subsystem-Level Formulations

Because each subsystem discrepancy in formulation 2 consists of a single term, weighting it differently is meaningless and taking its square root may cause a divide-by-zero floating error when both system and subsystem levels start from zero. For a fair comparison among the three subsystem-level formulations, assume no different weighting on the individual discrepancy terms and no subsystems using square roots in discrepancy minimization. Then, compare the three formulations with respect to results of basic application 1 (Tables 1, 7, and 9) and those of enhanced application 2 (Tables 4, 8, and 10).

Formulation 2 is the best: That is, if each subsystem restricts its freedom to the least in disagreeing on the passed-down targets, the approximate solution is very close to the exact. Even with a heavy one-shot penalty ($p = 1000$) that has a similar effect to instant lockup when the equality consistency constraints are enforced immediately, the error is 3.5% for all starting points; with a lighter initial penalty and a slower increase, the error is merely 0.04% for all starting points. It is difficult to call an error of 3.5%, although not necessarily acceptable depending on the specific accuracy requirement, a convergence difficulty. On the other hand, compared to formulation 2, taking up more freedom in disagreement as in both formulations 1 and 3 becomes unnecessary, and the results are worse. With the same heavy one-shot penalty as on formulation 2, both the worst and second-worst solutions of formulations 1 and 3 have unacceptably large errors, around 100% or larger (Tables 1 and 9). With the same lighter initial penalty and same slower increase as used in formulation 2, both the worst and second-worst solutions of formulations 1 and 3 have much smaller errors, around 10% or larger (Tables 4 and 10). A 10% error is much less acceptable than a 3.5% error.

Equality Constraints

The explicit effect of equality discrepancy constraints on CO is the vanishing of their gradients, which then cause convergence difficulty in the system-level optimization. It is because the equality constraints are of the SSD type. Equality discrepancy constraints of the component type, such as $\boldsymbol{\sigma}_{i*} - \boldsymbol{s}_i = 0$, $\boldsymbol{\zeta}_{i*} - \boldsymbol{z}_i = 0$, $\boldsymbol{\eta}_{i*} - \boldsymbol{y}_i = 0$, etc., used in the CO_1 formulation also investigated by Alexandrov and Lewis,[12] do not have the same kind of vanishing effect and convergence difficulty.

The equality constraints whose effect on a two-level formulation of a structural design problem Thareja and Haftka examined[14] are different from those examined in this paper. They are similar to the itemwise "interdisciplinary consistency constraints" in "distributed analysis optimization"[25] and "individual discipline feasible"[26] formulations and will not cause their gradients to vanish, either. Consequently, Thareja and Haftka applied the penalty-function method together with extended interior penalty, this is, barrier, functions to a different two-level formulation from the CO.

VI. Conclusions

System-level feasible targets cannot satisfy the KKT first-order necessary conditions because the consistency-constraint Jacobian vanishes whenever the consistency constraints are satisfied. Unconditional failure of the consistency constraints to satisfy CQ is the underlying cause. System-level feasible targets can only satisfy the Fritz John first-order conditions.

A vanishing consistency-constraint Jacobian can cause difficulty in the analytical derivation of CO solutions. When trying to derive CO solutions analytically for the study example, difficulty is encountered at the system level because the vanishing of consistency-constraint Jacobian combined with satisfying the consistency equality constraints implies, through the KKT conditions, that no CO solution exists. System-level targets can approximate the predecomposed solution but only if approaching from outside the system-level feasible region by totally infeasible targets.

At the subsystem level, if gradients of design constraints are assumed to be linearly independent for applying the KKTi conditions, then all associated Lagrange multipliers must vanish when the targets are system-level feasible, contradicting the assumption of linear independence.

A modification in the CO formulation that will make the KKT conditions applicable without multiplier troubles is to require the subsystems fit their optimization variables to their analysis equations with no constraints and to require the system level take charge of satisfying all of the design constraints specified through unconstrained subsystem solutions. Such a modified CO problem and the predecomposed problem have the same solution.

Basic application of the penalty-function method may encounter convergence difficulty due to vanishing gradients of the consistency constraints similar to direct application of an equality-constrained method, but enhanced application can greatly improve the worst and the best of the approximate solutions. Enhanced application can mitigate the difficulty by postponing or slowing down the disappearance of the consistency-constraint gradients. From the penalty viewpoint, CO with consistency equality constraints essentially is the case of imposing too high penalty on the discrepancies too early and, hence, enforcing a lockup too soon. Computational results are inevitably approximate. Even satisfying constraints is often approximate. Thus, why not let consistency-constraint satisfaction be approximate or asymptotic?

Taking the least of the freedom allowable by the CO formulation in using or producing variants of the passed-down targets has the advantage of little difficulty in convergence.

Appendix: Direct Check of CQ by Definition

Here, we check constraint qualification (CQ) directly by definition. We cite the original definition in the present context, with vector $\boldsymbol{X}$ representing system-level variables $(\boldsymbol{s}, \boldsymbol{y})$ and apply it to the system-level optimization problem.

First let $\boldsymbol{X}^0$ belong to the boundary of the constraint set of points satisfying $d_{i*}(\boldsymbol{X}) = 0$ for all i. For CQ,[15] it is assumed for each $\boldsymbol{X}^0$ of the constraint boundary that any vector $\boldsymbol{\xi}$ satisfying linear equations

$$\left[\frac{\partial d_{i*}}{\partial \boldsymbol{X}}\right]^0 \boldsymbol{\xi} = 0 \qquad \text{for all} \qquad i \tag{A1}$$

is tangent to an arc contained in the constraint set; in other words, to any vector $\boldsymbol{\xi}$ satisfying Eq. (A1) there corresponds a differentiable arc $\boldsymbol{\gamma}(t)$, $0 \le t \le 1$, contained in the constraint set, with $\boldsymbol{\gamma}(0) = \boldsymbol{X}^0$ and $[\mathrm{d}\boldsymbol{\gamma}/\mathrm{d}t]^0 = k\boldsymbol{\xi}$ for some positive scalar k.

Then an application to system-level problem (19) is demonstrated. The set of points satisfying consistency constraints (19b) is given by all points (s, y_1, y_2) satisfying the domain definitions for Eqs. (17a) and (18a), that is, satisfying

$$s - 2y_1 + y_2 - 1 \le 0, \qquad -s - y_1 + 2y_2 + 2 \le 0 \tag{A2}$$

The boundary points are readily given by $y_1^0 = s - \frac{4}{3}$ and $y_2^0 = s - \frac{5}{3}$ for any s. Because $(\partial d_{i*}/\partial s, \partial d_{i*}/\partial y_1, \partial d_{i*}/\partial y_2) = 0$ for both $i = 1$ and 2 at such boundary points (see Sec. IV.E), any vector $\boldsymbol{\xi}$ can satisfy the linear equations corresponding to Eq. (A1). Take $\boldsymbol{\xi} = (0, 1, 2)$ as one of the many possibilities. Define $\boldsymbol{\gamma}(t) = \boldsymbol{X}^0 + t\boldsymbol{\xi}$, $0 \le t \le 1$, where $\boldsymbol{X}^0$ is any of the boundary points (s, y_1^0, y_2^0). Then,

$$(s, y_1, y_2) = \boldsymbol{\gamma}(t) = \left(s + t\boldsymbol{\xi}_1, s - \tfrac{4}{3} + t\boldsymbol{\xi}_2, s - \tfrac{5}{3} + t\boldsymbol{\xi}_3\right)$$
$$= \left(s, s - \tfrac{4}{3} + t, s - \tfrac{5}{3} + 2t\right)$$

It is not difficult to verify that any point (s, y_1, y_2) defined by $\gamma(t)$ with $t > 0$ cannot satisfy the second inequality in (A2). Thus, for no positive t is arc $\gamma(t)$ contained in the constraint set. Therefore, CQ cannot be satisfied.

References

[1]Kroo, I. M., Altus, S., Braun, R., Gage, P., and Sobieski, I., "Multidisciplinary Optimization Methods for Aircraft Preliminary Design," AIAA Paper 94-4325, Sept. 1994.

[2]Braun, R. D., and Kroo, I. M., "Development and Application of the Collaborative Optimization Architecture in a Multidisciplinary Design Environment," *International Congress on Industrial and Applied Mathematics*, Aug. 1995, URL: http://techreports.larc.nasa.gov/ltrs/dublincore/1995/NASA-95-iciam.rdb.html.

[3]Balling, R. J., and Wilkinson, C. A., "Execution of Multidisciplinary Design Optimization Approaches on Common Test Problems," *AIAA Journal*, Vol. 35, 1997, pp. 178–186.

[4]Alexandrov, N. M., and Kodiyalam, S., "Initial Results of an MDO Method Evaluation Study," AIAA Paper 98-4884, Sept. 1998.

[5]Kodiyalam, S., "Evaluation of Methods for Multidisciplinary Design Optimization, Phase I," NASA CR-1998-208716, Sept. 1998.

[6]Cormier, T., Scott, A., Ledsinger, L., McCormick, D., Way, D., and Olds, J., "Comparison of Collaborative Optimization to Conventional Design Techniques for Conceptual RLV," AIAA Paper 2000-4885, Sept. 2000.

[7]Balling, R. J., and Sobieszczanski-Sobieski, J., "Optimization of Coupled Systems: A Critical Overview of Approaches," *AIAA Journal*, Vol. 34, 1996, pp. 6–17.

[8]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; also AIAA Paper 96-0711, Jan. 1996.

[9]Kroo, I. M., and Manning, V. M., "Collaborative Optimization: Status and Directions," AIAA Paper 2000-4721, Sept. 2000.

[10]Tappeta, R. V., and Renaud, J. E., "Multiobjective Collaborative Optimization," *Journal of Mechanical Design*, Vol. 119, 1997, pp. 403–411.

[11]DeMiguel, A.-V., and Murray, W., "An Analysis of Collaborative Optimization Methods," AIAA Paper 2000-4720, Sept. 2000.

[12]Alexandrov, N. M., and Lewis, R. M., "Analytical and Computational Aspects of Collaborative Optimization," NASA TM-2000-210104, April 2000; also *AIAA Journal*, Vol. 40, No. 2, 2002, pp. 301–309.

[13]Braun, R. D., Gage, P., Kroo, I. M., and Sobieski, I., "Implementation and Performance Issues in Collaborative Optimization," AIAA Paper 96-4017, Sept. 1996.

[14]Thareja, R., and Haftka, R. T., "Numerical Difficulties Associated with Using Equality Constraints to Achieve Multi-Level Decomposition in Structural Optimization," AIAA Paper 86-0854, May 1986.

[15]Kuhn, H. W., and Tucker, A. W., "Nonlinear Programming," *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, Univ. of California Press, Berkeley, CA, 1951, pp. 481–492.

[16]Arrow, K. J., Hurwicz, L., and Uzawa, H., "Constraint Qualifications in Maximization Problems," *Naval Research Logistics Quarterly*, Vol. 8, No. 2, 1961, pp. 175–191.

[17]Mangasarian, O. L., *Nonlinear Programming*, McGraw–Hill, New York, 1969, Chaps. 5, 7, 11.

[18]Hestenes, M. R., *Optimization Theory: The Finite Dimensional Case*, Wiley-Interscience, New York, 1975, Chaps. 3, 4.

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

[20]Schmit, L. A., and Chang, K. J., "Optimum Design Sensitivity Based on Approximation Concepts and Dual Methods," *International Journal for Numerical Methods in Engineering*, Vol. 20, Jan. 1984, pp. 39–75.

[21]Braun, R. D., Kroo, I. M., and Gage, P. J., "Post-Optimality Analysis in Aerospace Vehicle Design," AIAA Paper 93-3932, Aug. 1993.

[22]Spellucci, P., "DOMIN's User Guide," 1995, URL: http://www.mathematik.tu-darmstadt.de/ags/ ag8/Mitglieder/spellucci_en.html.

[23]Vanderplaats, G. N., "CONMIN—a FORTRAN Program For Constrained Function Minimization, User's Manual," Addendum to NASA TM X-62282, May 1978.

[24]Luenberger, D. G., *Introduction to Linear and Nonlinear Programming*, Addison–Wesley, Reading, MA, 1973, Chap. 12.

[25]Alexandrov, N. M., and Lewis, R. M., "Algorithmic Perspectives on Problem Formulations in MDO," AIAA Paper 2000-4719, Sept. 2000.

[26]Cramer, E. J., Dennis, J. E., Jr., Frank, P. D., Lewis, R. M., and Shubin, G. R., "Problem Formulation for Multidisciplinary Optimization," *SIAM Journal on Optimization*, Vol. 4, No. 4, 1994, pp. 754–776.

A. Messac
Associate Editor