AIAA JOURNAL
Vol. 38, No. 10, October 2000

# Collaborative Optimization Using Response Surface Estimation

I. P. Sobieski* and I. M. Kroo†
*Stanford University, Stanford, California 94305*

**The use of response surface estimation in collaborative optimization, an architecture for large-scale multidisciplinary design is described. Collaborative optimization preserves the autonomy of individual disciplines while providing a mechanism for coordinating the overall design problem and progressing toward improved designs. Collaborative optimization is a two-level optimization architecture, with discipline-specific optimizations free to specify local designs, and a global optimization that ensures that all of the discipline designs eventually agree on a single value for those variables that are shared in common. Results demonstrate how response surface models of subproblem optimization results improve the performance of collaborative optimization. The utility of response surface estimation in collaborative optimization depends on the generation of inexpensive accurate response surface models and the refinement of these models over several fitting cycles. Special properties of the subproblem optimization formulation are exploited to reduce the number of required subproblem optimizations to develop a quadratic model from $\mathcal{O}(n^2)$ to $\mathcal{O}(n/2)$. Response surface refinement is performed using ideas from trust region methods. Results for the combined approaches are compared through the design optimization of a tailless unmanned air vehicle in 44 design variables.**

## Nomenclature

$B$ = coefficients in response surface fit
$b$ = span
$C$ = coefficients in response surface fit
$C_l$ = section lift coefficient
$C_m$ = pitching moment coefficient
$c_i$ = local constraints
$f_{\text{rs}}$ = response surface approximation of $f$
$h_k$ = number of locally computed parameters with system-level targets
$h_m$ = number of local design variables with corresponding system-level targets
$m$ = number of unknown coefficients in fit
$N$ = number of collaborative optimization (CO) subproblems
$n$ = number of targets to CO subproblem
$q$ = dynamic pressure
$R$ = computed aircraft range
$R_0$ = target aircraft range
$t_i$ = spar cap thicknesses
$W$ = weight fraction local design variable
$W_{r,t}$ = trim ballast weight, tip and root
$W_0$ = target weight fraction ($W_{\text{initial}}/W_{\text{final}}$)
$w_i$ = penalty weight
$x_i$ = local design variables with a corresponding system-level specified target, $z_i$
$\bar{x}$ = strictly local design variables without corresponding system-level specified targets
$y_i$ = local computed state variable, a function of $x$ or $\bar{x}$, with a corresponding target $z_i$
$z_i$ = system-level targets
$\alpha$ = maneuver angle of attack
$\delta_e$ = elevon deflection angle
$\eta$ = propeller efficiency
$\theta_i$ = section jig twist

## Introduction: Multilevel Decomposition and Collaborative Optimization

THE solution of large engineering design problems often benefit from decomposition of the large problem into smaller tasks. Decomposition theory was originally formulated in the operations research field[1] and was motivated by the desire to align engineering tasks with the natural disciplinary groups found in large problems and to exploit the computational benefits that usually arise from the concurrent execution of analyses. Several reviews of the development of multilevel decomposition theory and optimization have been previously published.[2,3] The order, or schedule, in which tasks are performed has been shown to influence substantially the efficiency of engineering analysis.[4] The most straightforward multilevel decomposition procedure is to match an optimization algorithm to an integrated set of analyses. The efficiency of this approach is greatly enhanced through the use of system sensitivities that predict the changes in computed parameters as a function of optimization design variables.[5] Improvements in efficiency are further possible through the addition of design variables in the optimization that represent the intermediate computed quantities used by subsequent tasks; in this way, all tasks may be executed concurrently.[6] However, neither of these approaches provide autonomy to the design tasks themselves. Concurrent subspace optimization (CSSO) is one approach developed to address this difficulty.[7] CSSO partitions the authority to specify design variable values among the different disciplines. Each discipline is free to perform a local optimization on its own set of design variables. A coordination problem is performed to ensure that constraints are satisfied and progress is made toward an optimum. A variety of improvements have been made to CSSO including the use of expert systems to set move limits,[8] inclusion of discrete variables,[9] and the use of response surface approximations.[10,11] Collaborative optimization is an alternative architecture for distributed optimization that decomposes the design process. This paper describes a version of this architecture that improves system robustness and performance.

## Collaborative Optimization

Collaborative optimization (CO) is a multidisciplinary design method that preserves disciplinary-level design freedom while providing a coordinating mechanism that ensures progress toward an optimum and compatibility between the disciplinary designs. Descriptions of CO have been the subject of several conference papers[12–14] and archival articles.[15–17] A key benefit of the collaborative architecture is its similarity to existing design processes. Aircraft design and other large real-world engineering design problems are performed in parallel, by disciplinary teams. Apportioning responsibility for specifying the value of variables that affect, or are computed by, more than one discipline is often not obvious. The current approach for resolving these system-level design decisions is often ad hoc and inefficient. CO provides a formal mechanism for posing this problem, resolving dispute, and leading to improved

Received 20 July 1999; revision received 16 February 2000; accepted for publication 11 March 2000. 

*Postdoctoral Researcher, Department of Aeronautics and Astronautics.
†Professor, Department of Aeronautics and Astronautics. Associate Fellow AIAA.

designs. It does this while preserving disciplinary autonomy and design organization structure.

Each discipline is given complete freedom to specify a local design. This means that the disciplinary designer is free to specify the values of variables that are unique to the discipline $\{\bar{x}\}$, but also those that might be inputs, or computed outputs, to another discipline's analyses $\{x\}$. The disciplinary design goal is to satisfy local constraints $\{c\}$ while finding a design that minimizes the conflict between the local design values of parameters that are also specified, or computed, by other disciplines ($J = \|\{x\} - \{z\}\|$). In this version of CO, the subproblem design is given by

$$\min J_i = \sum_{1}^{h_m} (z_j - x_j)^2 + \sum_{h_m+1}^{h_k} (z_j - y_j)^2$$

$$X = \{x_1, \ldots, x_m\}, \{\bar{x}_1, \ldots, \bar{x}_k\}, \qquad \text{s.t. } c_i(X, y) \geq 0 \qquad (1)$$

For example, the structures design team may be tasked with specifing a wing design. Their discipline is the only one that uses or specifies the stringer thicknesses. These are variables that are unique to the structures discipline, $\{\bar{x}\}$. Other variables that the structures discipline may use, such as wing span or planform area, are used or specified by other disciplines such as aerodynamics; these are shared variables. However, CO enables all of the disciplines, structures in this case, to modify the value of shared variables used in the analyses and design. In this example, the value of wing span used in the structural design may be different from the value used by the aerodynamics design. Obviously, there can be only one value of span in the end, but the strength of CO is that it provides a rational way of ensuring that the disciplines agree on values for shared variables. This coordination is performed by a system-level design optimization problem. The system-level problem establishes target values for all shared design parameters, those that are either computed or used as inputs in more than one discipline. These variables are designated system-level targets and represented by $\{z\}$. The task of the disciplinary subproblem optimizations is to find a local design that comes as close to that specified by the system-level optimizer as possible. This is expressed in Eq. (1) as minimizing the value of $J$ that represents the square of the difference between local values of design parameters and their corresponding system-level targets. Note that all variables do not have a system-level target, only those variables that are used by one discipline and are used, or computed, by another. This disciplinary subproblem is posed in such a way that its solution is a locally feasible design, that is, all local constraints will be satisfied. However, there is no guarantee that for some arbitrary vector of targets $\{z\}$ the resulting optimal value of $J$, that is, $J^*$, will be zero. Because of local constraints or the functional relationship between $\{y\}$ and $\{x\}$, $J^*$ may be nonzero over a large part of the domain of $\{z\}$. This is a situation often found in real-world design, where many disciplines vie with one another over the values of design parameters that affect both disciplines. To ensure that all disciplines agree on values of interdisciplinary, for example, shared, variables, a system-level optimization problem is specified as

$$\text{minimize:} \quad z_n$$

$$\text{with respect to } Z = \{z_1, \ldots, z_n\}$$

$$\text{s.t. } J_i^*(Z) = 0 \qquad \forall_{\text{subproblems}_i} \qquad (2)$$

This system-level optimizer adjusts values of the target vector $\{z\}$ with the goal of improving a measure of the overall design. This problem is subject to the constraint that all of the disciplines agree with one another, that is, in the end, the $J_i^*$ values all equal zero. No general convergence proof has been developed for the CO architecture represented by Eqs. (1) and (2). However, it is trivial to demonstrate that the solution to a single-level formulation is also a solution to the collaborative problem. The converse has not been proven; however, numerous numerical experiments have found identical solutions when performed with both a single-level and collaborative formulation.[12,13,17]

Equations (1) and (2) represent a two-level optimization architecture, where each evaluation of the system-level constraint $J_i^*$ requires a complete subproblem optimization given by Eq. (1). Because a subproblem optimization may involve substantial analysis, this relationship puts a heavy burden on the efficiency of the system-level optimization algorithm. As described in Ref. 12, problems that are well suited for CO are those that have few interdisciplinary variables relative to strictly local variables and constraints. Even with such problems, the structure of the system-level problem may cause some optimizers to require prohibitively large numbers of executions of the disciplinary analyses.[17] To reduce this burden, we introduce the use of response surface models that approximate $J_i^*$ as a function of $\{z\}$.

## Motivation for Using Response Surfaces in CO

Response surfaces have several properties that make them attractive for use with optimization in general and with CO in particular:

1) They provide a natural way of implementing coarse-grained parallelization.

2) They are computationally inexpensive to evaluate and therefore avoid potential numerical difficulties that some optimizers experience near the solution point.

3) They represent noisy analysis with an inherently smooth model.

In CO, multiple processors can be used to execute the subproblem optimization given by Eq. (1) for the set of different target vectors required to generate the response surface fit. In this way, using multiple processors, a full response surface may be generated in the time required for a single subproblem optimization. The conventional CO formulation (i.e., the subproblem optimization problems coupled directly with the system optimizer) involves repeated serial optimization of the subproblems.[13]

In addition, some optimization algorithms used at the system level have been observed to have difficulty with the form of the compatibility constraints, especially near the solution.[12,17] Use of response surface models of the system-level constraints avoids the computational expense associated with successive evaluations of the compatibility constraints. Instead, the subproblem optimization is only performed if and when the response surface is regenerated. In general, the use of response surface models of the subproblem optimization results reduces the efficiency burden on the system-level optimization algorithm.

In most problems of practical interest, such as the one used as an example in this paper, a single response surface model is not sufficient to represent the full design domain. Rather, the new design point predicted by the model is evaluated using the full analysis and, if necessary, a new response surface is generated and the process repeated.

## Using Response Surface Models

Response surfaces are typically used to model disciplinary analyses. Figure 1a shows using response surfaces (RS) in this way within a CO framework. An RS of the subproblem analysis is generated by solving the analysis for a set of design vectors $\{x\}$ and obtaining a corresponding set of computed state variables $\{y\}$. The

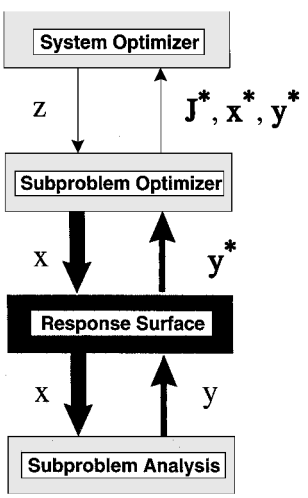


**Fig. 1a RS models analysis.**

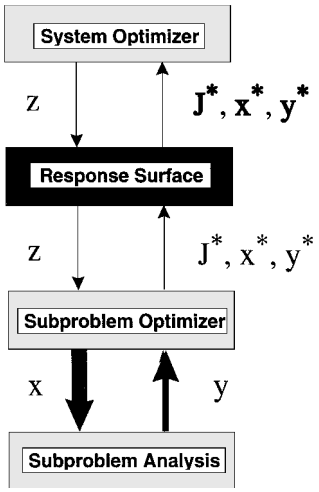


**Fig. 1b RS models optimization.**

subproblem optimizer uses the RS approximation, in lieu of the analysis, to minimize the subproblem objective function $J$, for example, system-level compatibility constraint. This approximated value of $J^*$ is passed back to the system-level optimizer.

Use of RS in this manner presents difficulties for problems of high dimensionality. To generate even a relatively low-order quadratic RS requires $\mathcal{O}(n^2)$ function evaluations. Typical design analyses often involve hundreds of variables, resulting in an impractical number of required function evaluations.[12,14]

The alternate approach proposed here is shown in Fig. 1b. The RS is generated by solving the subproblem optimization for a set of target variable design vectors $\{z\}$ and obtaining a corresponding vector of subproblem objective function values $\{J^*\}$. The system-level optimizer then uses this RS in lieu of the subproblem optimization, to approximate $J_i^*$ as a function of the target vector $\{z\}$. This approach is similar to that used in modeling the subspace objective and constraints in CSSO.[10,11] However, unlike CSSO the RS models of the subspaces need to represent only the subspace objective; all constraint information is maintained at the local level and does not need to be communicated back to the system level.

Problems well suited for CO are already divided along disciplinary boundaries that require relatively little interdisciplinary communication. Although the design subproblems may include hundreds of design variables, they typically include a much smaller number of interdisciplinary variables. Response surface approximations of design results (Fig. 1b) reduces the required dimensionality of the model. This, coupled with the techniques presented in this paper to substantially reduce the number of design points required to generate a quadratic fit, allows practical regeneration of the RS model through the course of a system-level optimization.

## Modeling Optimal Subproblem Designs

This section considers approaches for determining a quadratic fit to the subproblem optimal design results. The first directly models the subproblem optimal objective function $J^*$ as a function of the target variable vector $\{z\}$. Postoptimality gradient information is shown to reduce the number of required points to generate this fit by order of $n$. The second method models the optimal subproblem design vectors $\{x^*\}$ and $\{y^*\}$, as a function of $\{z\}$, with a similarly small number of required optimizations. Finally, information inherent in each subproblem solution is exploited to reduce the number of required subproblem optimizations, using either method, by another factor of 2.

### Estimating the Subproblem Objective $J^*$

The general form of a quadratic RS fit in $n$ variables is

$$J^*(z) = C_0 + \sum_{i}^{n} C_i z_i + \sum_{1 \le i \le j \le n}^{n} C_{ij} z_i z_j \qquad (3)$$

The number of unknown coefficients in Eq. (3) is $m = (n+2)(n+1)/2$. Determining these coefficients requires $m$ different values of $J^*$. These may be obtained by executing the subproblem optimization for $m$ target design vectors, $\{z\}_1, \{z\}_2, \ldots, \{z\}_m$, selected so that they do not lie in the same $n-1$ dimensional hyperplane. However, more information is obtained from each solution of the subproblem optimization than simply the value of $J^*$, and thus a quadratic model may be developed in fewer subproblem optimizations.

### Fit Using Postoptimality Gradients

The subproblem design solution $J^*$ is the optimal value of $J$ as a function of the local design variables with the system level target vector $\{z\}$ held constant. As such, the vector $\{z\}$ is similar to any other constant parameter in the optimization problem, and the sensitivities of the optimal result $J^*$ with respect to $\{z\}$ can be calculated without further analysis using the methods of postoptimal sensitivity. The ability to extract the gradient of $J^*$ with respect to $\{z\}$ has been introduced as a means of substantially improving the efficiency of CO in its conventional implementation.[14] By the accounting for redundant information, the use of postoptimal sensitivities allows a quadratic model to be developed with just $m = n + 1$ subproblem optimizations.

### Modeling Subproblem Optimal Design: $\{y^*\}$ and $\{x^*\}$

The RS model can represent $J^*$ as an explicit quadratic function of the system level targets vectors $\{z\}$, $J^* = f(z)$. An alternate approach is to create RS models that predict the optimal subproblem design parameters $\{x^*\}$ and $\{y^*\}$ as functions of $\{z\}$:

$$\{x^*\} = f(z), \qquad \{y^*\} = g(z) \qquad (4)$$

and substitute these models of $\{x^*\}$ and $\{y^*\}$ into Eq. (1) to determine $J^*$. After choosing a linear functional relationship for the model, Eq. (4) becomes

$$x_j^* = B_{j0} + \sum_{i=1}^{n} B_{ji} z_i, \qquad y_g^* = C_{g0} + \sum_{i=1}^{n} C_{gi} z_i \qquad (5)$$

where $x_j$ and $y_g$ are elements of $\{x\}$ and $\{y\}$. The resulting model of $J^*$ [see Eq. (1)] is still of second order, but now has different properties from the earlier explicit fit of $J^*$. First, the resulting model is always nonnegative, whereas the method described in the preceding section allowed the model of $J^*$ to take on negative values, which we know, from the form of Eq. (1), is actually positive for all $\{z\}$. Second, no explicit gradient information is required to generate this fit. As will be shown in a later example, this allows for a looser convergence tolerance in solving the subproblem optimization than is required if gradient information is used to generate the model. Third, because there are now only $n + 1$ coefficients to compute and one independent equation is obtained for each subproblem optimization, the number of subproblem optimizations required to generate the RS is still $m = n + 1$. The number of points required to generate this fit is the same as that required to generate an explicit quadratic fit of $J^*$ with postoptimal gradient information and a factor of $n$ fewer points than required to fit $J^*$ using function values alone.

### Fit Using Implicit Extra-Point Information

Special information in each subproblem optimization is exploited to further reduce the number of required subproblem optimizations by 50%. A given subproblem optimization yields values of $\{x^*\}$, $\{y^*\}$, and $J^*$, for a given $\{z\}$. The key observation is that if a second subproblem optimization were performed, where the target values $\{z\}$ were set equal to the optimal solution from the first subproblem optimization ($\{x^*\}, \{y^*\} \in \{z\}$), the resulting $J^*$ would be equal to zero. This is because the formulation of the subproblem optimization requires that all local constraints be satisfied and that the computed state variables $\{y^*\}$ be consistent with the local design variables $\{x^*\}$. Because the result is known a priori, there is no need to actually perform another subproblem optimization; rather an extra subproblem optimization solution is obtained implicitly, with no additional analysis. That is, for each subproblem optimization, one learns the value of $J^*$ that corresponds to the given $\{z\}$ and another point where $J^*(\{z\}) = 0$.

When $\{x^*\}$ and $\{y^*\}$ are modeled, the implicit point provides one additional equation per subproblem solution, which reduces

**Table 1 Number of points required for full quadratic fit**

| Dimension | Function evaluations only | Fit using $J$ and $\nabla J$ or $\{x^*\}$ and $\{y^*\}$ | Extra-point information |
|---|---|---|---|
| 1 | 3 | 2 | 1 |
| 2 | 6 | 3 | 2 |
| 3 | 10 | 4 | 2 |
| 4 | 15 | 5 | 3 |
| 5 | 21 | 6 | 3 |
| 10 | 66 | 11 | 6 |
| 20 | 231 | 21 | 11 |
| 30 | 496 | 31 | 16 |

the number of required subproblem optimizations to generate a quadratic response surface to $m = (n + 1)/2$.

When modeling $J^*$ directly, each subproblem optimization yields one additional solution $J^*$ and its full gradient (equal to zero), for a total of $n + 1$ additional equations. The total number of subproblem optimizations required also becomes $m = (n + 1)/2$.

There are limitations as to when the implicit point information is available. There is no general guarantee that the $\{x^*\}$ that corresponds to $J^*$ for a particular $\{z\}$ will be independent (not lie on the same $n - 1$ dimensional hyperplane) from the set of $\{x^*\}$ generated by the subproblem solutions for other target vectors $\{z\}$. The implicit point information also does not exist for cases where the solution to the subproblem optimization is already zero, for example, where $J^*(\{z\}) = 0$. For this result, the subproblem has successfully matched the target design variables $\{z\}$, and there is no extra point. Whether implicit information is present in the subproblem solutions may be discerned during the course of the overall system-level optimization and additional points computed as needed.

The recognition of the existence of this implicit solution allows a second-order RS model of the subproblem optimization results to be generated with between $m = (n + 1)/2$ and $(n + 1)$ subproblem optimizations.

## Summary and Comparison of Fitting Approaches

Table 1 shows the computational requirements for developing a full quadratic model of the subspace optimal design using the various approaches described here. Use of postoptimal sensitivities eliminates the quadratic growth in required function evaluations with the number of design variables $n$. This same reduction is found when the individual components of the subproblem discrepancy functions involving $\{x^*\}$ and $\{y^*\}$ are individually modeled.

The use of implicit extra-point information further reduces the number of subproblem optimizations in many cases. The combination of these methods provides a method for inexpensive generation of quadratic RS models of the subproblem discrepancy function $J^*$ and substantially improves the computational efficiency of collaborative optimization.

These are general approaches to fitting the optimization results of CO subproblems. However, note that, in its practical implementation, CO allows the local discipline to choose the form of its RS. The only requirement of the system-level optimizer is that it receive a value for $J^*$ when it provides the disciplines a target vector $\{z\}$.

## RS Domain and Refinement

Techniques have been shown that allow for the generation of quadratic RS models of subproblem optimal results in as few as $\mathcal{O}(n/2)$ optimizations. However, for any disciplinary design problem where $J^*$ is not a quadratic function of $\{z\}$, the RS model will not be accurate over the full domain of $\{z\}$. Thus, the use of RS estimation in collaborative optimization requires that the full disciplinary design optimization be executed at the predicted system-level optimum $\{z^*\}$ and the resulting $J^*$ value be compared with the value $J^*_{\text{rs}}$ predicted using RS. If this error is unacceptable, new RS are generated about the $\{z^*\}$ point, and the system-level optimization is executed using these new models. Additionally, the domain over which a quadratic representation of $J^*$ is valid will be problem dependent, but also a function of the spacing of points evaluated using the real analysis to generate the model. Thus, in generating new RS models the spacing of the points needs to be able to accommodate variations in the accuracy of the base quadratic model.

Both of these goals are accomplished by borrowing ideas from trust region optimization methods. A general algorithm for minimizing the function $F$ using collaborative optimization and RS estimates follows.[18,19] Here, the algorithm adjusts the spacing of the points used to generate the RS and the need to regenerate the models based on the relative accuracy of the predicted design point.

1) Specify an initial design vector $\{z\}_k$.

2) Define a region of trust $\Delta$.

3) Generate multiple vectors $\{z\}$ as required to solve for the coefficients of the RS.

4) For each vector $\{z\}$ perform a subproblem optimization, minimizing the difference between the target vector $\{z\}$ and local values of $\{x\}$ and computed quantities $\{y\}$.

5) Solve for the unknown coefficients of the model to generate $f_{\text{rs}}(z)$.

6) Find the minimizer $\{z^*\}$ of the model $f_{\text{rs}}(z)$.

7) Evaluate the actual analysis at $f(z^*)$.

8) Compare $f(z^*)$ with $f_{\text{rs}}(z)$ and determine if $\{z^*\}$ is an acceptable point.

9) Adjust trust region size and/or initial point ($\{z\}_{k+1} = \{z^*\}$) and repeat from step 3.

The initial trust region size is chosen to correspond to the range over which the RS is judged to be accurate and is, thus, problem specific. If this initial guess is incorrect, as evaluated by the improvement in the solution, then the trust region size is modified. Three update approaches are considered and differ from each other in the way the error assessment, steps 8 and 9, are performed. The first accepts the new point only if the actual objective function value at this point is better. That is, if $f(\{z^*\}) < f(\{z\})$, where $z^*$ is found on the basis of the RS model and $\{z\}$ is the point around which the model is built, then accept $\{z^*\}$ as the new point and repeat the process. The second approach examines the relative accuracy of the model in predicting the objective function value. This approach is similar to that described in Ref. 20 and evaluates the new point by determining how well the predicted value compares with the actual value. Both of these methods adjust the size of the trust region symmetrically. The third update method is similar to the first except that the region is adjusted asymmetrically. If the new solution lies on the boundary of the trust region the region is expanded in that dimension. The size of the expansion is based on the size of the current trust region. In this work, the trust region size was doubled when the solution fell on a region boundary and cut in half when it fell within the region. This asymmetric growth procedure is used in the example problem that follows.

## Application to Example Problem

The RS estimation procedure outlined is applied to an example aircraft design problem involving 44 design variables and two disciplinary subproblems. This problem is solved both by means of the standard CO approach and with the methods described in this paper.

### Tailless Unmanned Air Vehicle Design Problem Description

The problem is to design a 454-kg (1000-lb), propeller-powered, tailless aircraft for maximum range. The planform, gross weight, and engine performance parameters are fixed, whereas jig twist, spanwise skin thicknesses, elevon deflection angle, maneuver angle of attack, and the mass of wing tip and wing root ballasts are free design variables. The optimizer uses these free design variables to meet trim constraints during cruise and a 4-$g$ maneuver. It also adjusts these variables to satisfy the structural stress constraints while maximizing the computed range.

The design problem is specified by

$$\begin{aligned} &\text{maximize:} && \text{range} \\ &\text{design variables:} && \{\theta\},\ \{t\},\ \alpha,\ \delta_e,\ W_r,\ W_t \\ &\text{subject to:} && n_{\text{maneuver}} = 4, \qquad C_{m_{\text{cruise}}} = 0, \qquad C_{m_{\text{maneuver}}} = 0 \\ & && \{C_l\}_{\text{maneuver}} < 1.45, \qquad \{\sigma\}_{\text{maneuver}} < 275\ \text{MPa} \end{aligned} \tag{6}$$

**Table 2 UAV fixed parameters**

| Parameter | Fixed value |
|---|---|
| Wing area | 18.59 m$^2$ |
| Aspect ratio | 15 |
| Airfoil thickness | 0.12 |
| Taper, $c_t/c_r$ | 0.5 |
| Sweep | 15 deg |
| Fuselage drag area | 0.139 m$^2$ |
| Takeoff weight | 453.5 kg |
| SFC | 0.086 kg/h/N |
| $\eta_{\text{propeller}}$ | 0.8 |
| $C_{d_{Pw}}$ | 0.009 |
| $W_{\text{fixed}}$ | 136.1 kg |

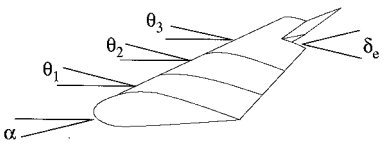


**Fig. 2 Design variables defining tailless UAV.**

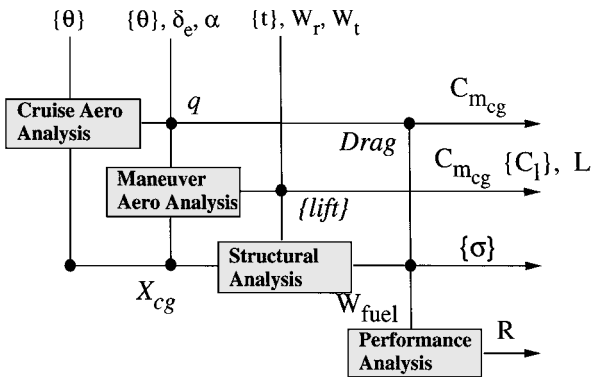


**Fig. 3 Structure of analyses in UAV problem.**

The specified planform, takeoff gross weight, and engine parameters for the design are given in Table 2. The design variables defining the tailless unmanned air vehicle (UAV) are shown in Fig. 2.

Figure 3 shows the structure of the analyses used to design the tailless UAV. The design variables are shown at the top of Fig. 3 and calculated output quantities, such as stress and range, are shown on the right. Dots indicate that a computed output is used by another analysis as an input. The same aerodynamic analysis is used to compute a lift distribution, drag, and overall lift and moment coefficients for two different flight conditions: a 4-$g$ maneuver and cruise. The 4-$g$ maneuver is performed at the cruise speed but at a lower altitude and higher dynamic pressure. The optimizer adjusts the elevon deflection angle and maneuver angle of attack to ensure that the generated lift is four times the takeoff gross weight, and the aircraft is trimmed in both flight conditions ($C_m = 0$).

Thickness design variable values are an input to the structural analysis, which computes a spanwise stress distribution for the given maneuver lift distribution. The position of the aircraft center of gravity is computed accounting for the fixed weights specified in Table 2, possible root or tip ballast weights, and the structural weight of the wing spar. As seen in Fig. 3 this information is used by the aerodynamic analysis for trim. Maximum fuel weight is also calculated in the structures discipline; this value, and the lift and drag computed by aerodynamics, is used by the performance analysis to compute range.

### UAV Analyses

The aerodynamic analysis uses a definition of wing incidence and the fixed planform information of Table 2, to compute the pitching moment coefficient, drag, and lift distribution. The wing is modeled using a vortex panel method with 20 equally sized panels.[21,22] Because the wing planform shape is fixed, the aerodynamic influence coefficients are calculated just once. The linear system is solved for the moment, lift, and drag coefficients. Total drag is the sum of the induced drag, the parasite drag of the fuselage, and the wing parasite drag, proportional to planform area. The computed $C_L$ and the known fixed weight determine the dynamic pressure. For the maneuver case, the same linear system is solved with a new incidence distribution, which is the combination of the maneuver angle of attack and, for the outboard four panels, the elevon deflection angle. The output includes the total lift, $C_{m_{cg}}$, and section lift distribution.

The wing is modeled structurally with as an I beam loaded in bending by the distributed lifting load computed in the aerodynamics analysis. The thickness of the I-beam spar caps is specified at 20 spanwise locations. The width of the beam is 30% of the local mean chord and 75% the local mean wing section thickness. The bending moment and section direct stress is calculated at the inboard end of each panel using a finite element method (FEM).[23] The wing bending weight is proportional to the material volume of the structural spar multiplied by 150% of the density of aluminum. The load-independent wing weight is included in the fixed weight parameter specified in Table 2. The root ballast is positioned at 25% of the root chord while the fixed weight acts at 60% of the root chord. The tip weight is located at the tip quarter chord. The total c.g. is computed using the location and size of the tip and root ballast masses, the position and size of the fixed weight, and the wing c.g. location. Fuel is assumed distributed uniformly along the wing so as not to affect c.g. location. Range is computed using Breguet's equation, and the fixed values of engine efficiency, specific fuel consumption (sfc), and takeoff gross weight are from Table 2.

### Optimal Design

The example problem was designed to be complex enough to represent an interdisciplinary problem but also simple enough so that the analyses could be integrated and solved using conventional black box optimization. These results establish a known baseline solution against which the performance of CO and the use of RS may be measured. An optimal tailless UAV design with a range of 9109 km (5660 mile) was found; the optimal design variable values and computed results are listed in Table 3. The lift and twist distributions are shown as the correct answer in Figs. 4 and 5.

### Collaborative Implementation

The example problem described here is chosen to demonstrate the implementation of CO using RS estimation. However, this problem does not represent the kind of design problems best suited for CO, which are, instead, real-world design problems with large teams, disciplinary specific analysis tools, and thousands of local variables and constraints. The problem is deliberately designed to be simple

**Table 3 UAV optimal design results**

| Parameter/ variable | Correct optimal value |
|---|---|
| $W_r$ | 0.0 kg |
| $W_t$ | 11.47 kg |
| $\alpha$ | 8.4 deg |
| $\delta_e$ | 12.2 deg |
| $L/D$ | 21.6 |
| Range | 10,480 km |

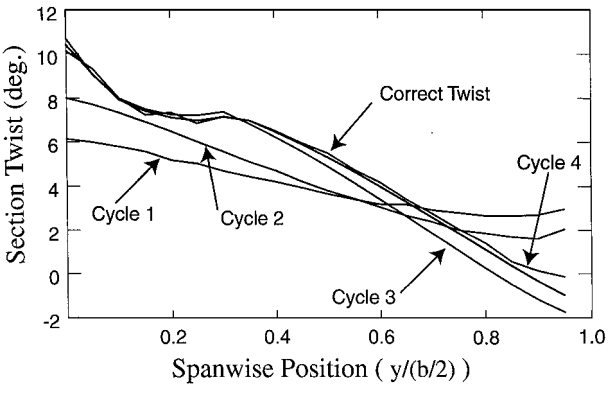


**Fig. 4 Spanwise twist vs trust region cycle.**

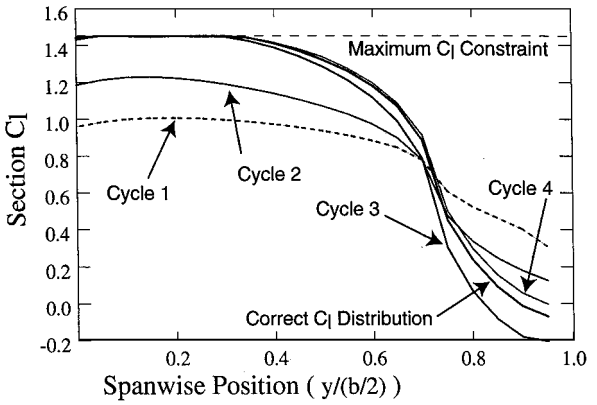


**Fig. 5 Section spanwise $C_l$ distribution.**

enough to execute as an integrated optimization (preceding section), in a conventional CO implementation, and using RS.

The single-level analysis was split along the disciplinary boundary between aerodynamics and structures. The variables shown in italics in Fig. 3 become system-level targets, and the variables at the top of Fig. 3 are strictly local subproblem design variables. The resulting aerodynamics subproblem is expressed by

$$\begin{aligned} &\text{minimize:} && J_1 \\ &\text{design variables:} && \{\theta\},\ \alpha,\ \delta_e,\ X_{\text{cg}} \\ &\text{subject to:} && C_{m_{\text{cruise}}} = 0, \qquad C_{m_{\text{maneuver}}} = 0 \\ & && \{C_l\}_{\text{maneuver}} < 1.45, \qquad n_{\text{maneuver}} = 4 \end{aligned} \tag{7}$$

The structures subproblem, which also includes the range calculation, is

$$\begin{aligned} &\text{minimize:} && J_2 \\ &\text{design variables:} && \{t\},\ W_r,\ W_t,\ a_1,\ a_3,\ \text{drag} \\ &\text{subject to:} && \{\sigma\}_{\text{maneuver}} < 275\ \text{MPa} \end{aligned} \tag{8}$$

The system-level optimization problem is

$$\begin{aligned} &\text{minimize:} && -R_0 + w_1 J_1 + w_2 J_2 \\ &\text{design variables:} && a_{1_0},\ a_{3_0},\ \text{drag}_0,\ X_{\text{cg}_0},\ R_0 \\ &\text{subject to:} && J_1 = 0, \qquad J_2 = 0 \end{aligned} \tag{9}$$

where the system objective is a penalty function of the aircraft range and the two compatibility constraints. The lift is represented by a reduced basis model using two Fourier coefficients: $a_1$ and $a_3$. The Fourier coefficient representation of the lift has been shown to be an effective means of reducing the subproblem dimensionality.[13] The resulting system-level optimization problem involves five variables and two constraints. The correct optimal solution is found in this manner but requires a large number of system-level iterations, suggesting that the RS methods introduced in this paper may be of use.

Figure 6 shows the behavior of the conventional implementation of CO on the tailless UAV problem. Note that a large portion of the major iterations are used to define a compatible design in the region of the optimal range. Each major iteration is equivalent to the parallel development of an RS model. Comparing this behavior to that shown in Fig. 7 is an illustration of the utility of RS estimation in CO.

#### Collaborative Results with RS

The tailless UAV problem is implemented using the RS modeling described by Eq. (5) in conjunction with the update method discussed earlier. Local RS are generated to model $J_1$ and $J_2$ from Eqs. (7) and (8) as a function of the target system-level variables of Eq. (9). The models of $J_1$ and $J_2$ are then used by the system-level optimizer to solve the problem specified by Eq. (9). Figure 8 shows

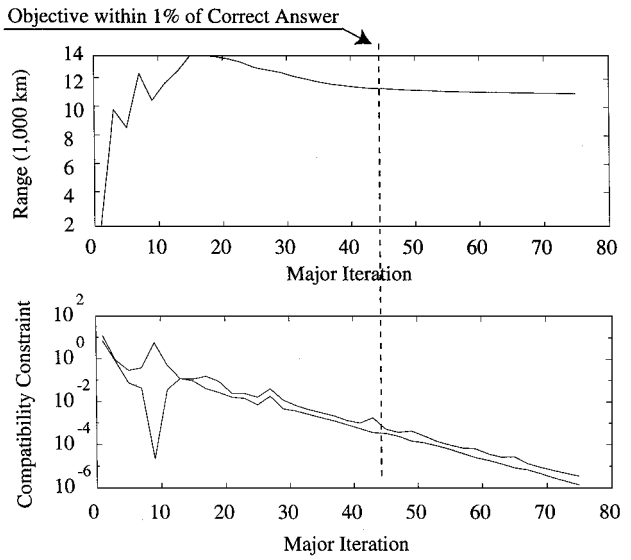


**Fig. 6 Convergence behavior of conventional CO implementation.**

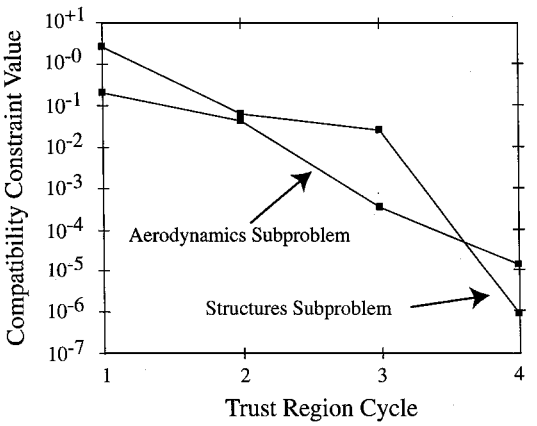


**Fig. 7 Optimal subproblem objective function values vs trust region cycle.**

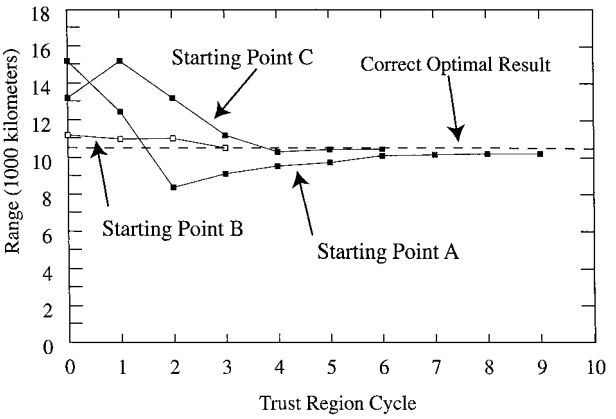


**Fig. 8 Range vs trust region cycle.**

the variation in range vs the RS update cycle for three different starting points. All three closely approach the correct optimal range (determined through the single-level optimization in the preceding section). Runs starting from points B and C were made with a tighter subproblem convergence tolerance. Each of these solutions were obtained in less than 10 trust region cycles. The optimization starting from point B obtained the correct solution after only four cycles. Figure 7 shows the values of the optimal structures and aerodynamics subproblem objective function values $J^*$ for this run. Viewed in combination with Fig. 8, one sees that the reported range values correspond with compatible subproblems ($J^* \approx 0$).

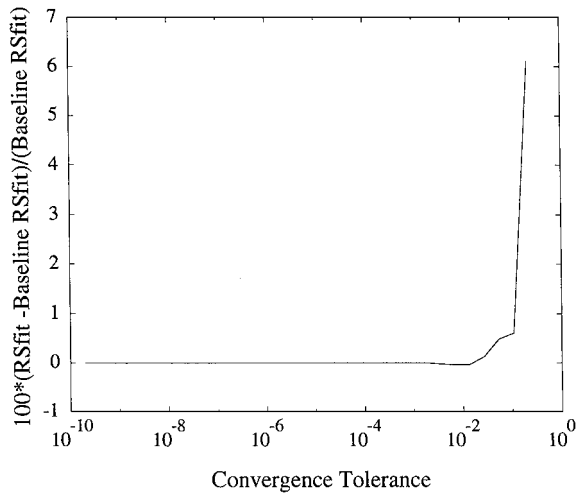


**Fig. 9 Variation in RS shape as a function of subproblem optimization convergence tolerance.**

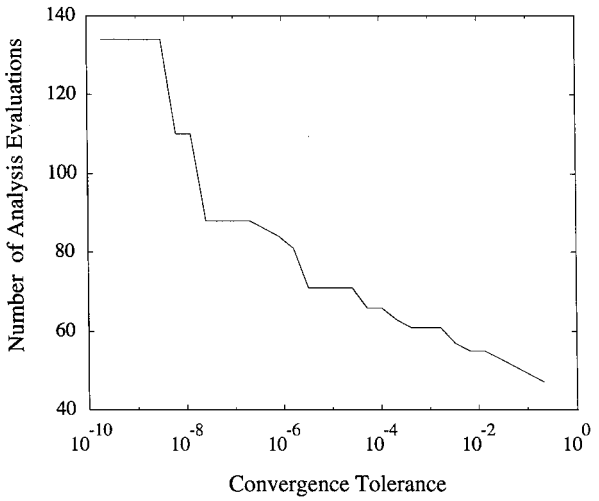


**Fig. 10 Analysis executions as a function of subproblem optimization convergence tolerance.**

The convergence of the design to the optimal result is also demonstrated by comparing the optimal wing twist and lift distribution for the CO implementation with the single-level solution, labeled correct twist in Fig. 4. Figure 4 shows the twist distribution determined by the aerodynamics subproblem optimization for the four cycles starting from point B. The twist distribution of the final CO design is almost identical to the correct solution. Similarly, the lift shown in Fig. 5 also converges to virtually the same spanwise distribution.

These results were obtained in just four trust region cycles. Each trust region cycle requires six subproblem optimizations. However, because the optimization results are used to generate an RS, these may be evaluated in parallel. Thus, the total number of serial subproblem optimizations required to generate this solution is between 4 (with full use of course-grained parallelization) and 24 (for execution on a single processor).

#### Convergence Requirements for Subproblem Optimizations with RS Representation

Because RS are used to model the subproblem objective over a large range, small variations in the value of the $J^*$ values used to generate the fit should not severely corrupt the model. To illustrate the benefit that follows from looser convergence requirements, an RS model was generated by evaluating the performance subproblem of the preceding example at three different target design points $\{z\}$ and fitting the resulting local design $\{x^*\}$ and $\{y^*\}$.

In Fig. 9 the variation in the RS model of this subproblem is shown as the subspace optimality tolerance is varied from $10^{-10}$ to $10^{-1}$. The $10^{-10}$ result serves as the baseline, and the models were compared with each other at 750 points. As is shown in Fig. 10, the resulting RS model varies little over this range of tolerances whereas the number of function evaluations, required to obtain a converged result, decreases as the convergence tolerance is loosened. Similar experience with postoptimal gradients shows that the required convergence tolerance for good gradients is about one order of magnitude tighter than for a stable RS model. Thus, use of RS models have the added benefit of allowing subproblem optimizations to be performed with looser requirements on subproblem convergence.

## Conclusions

The convergence behavior shown in Fig. 6 has been observed in the conventional implementation of CO used to solve other problems.[17] When each evaluation of the system-level compatibility constraints requires a complete subproblem optimization, this can lead to many analysis evaluations. RS provide a relatively inexpensive means for providing the value of the system-level compatibility constraints. RS can be generated in parallel, thus creating a second-order model of the optimal design surface in the time required for a single major iteration in the conventional CO implementation. An additional advantage of the use of RS models is that they are less sensitive to loose convergence of the subproblem optimization than conventional CO, which requires postoptimal sensitivities.

A technique has been described for generating quadratic RS fits of CO subproblem optimization results with a computational cost that scales by order $n$ rather than $n^2$. Additionally, recognition of implicit extra-point information in each CO subproblem solution reduces the computational expense of generating an RS by as much as 50%. These techniques make the generation of RS fits to realistic multidisciplinary problems practical.

A two subproblem collaborative optimization of a tailless UAV was performed using these RS techniques and a simple trust region algorithm. The correct solution was obtained in as few as four update cycles.

The use of quadratic models of the subproblem optimal designs is a compromise between computationally more expensive higher-order models and cheaper, but substantially less accurate, linear models. For many problems however, the general form of the RS function may be known a priori, and a more appropriate basis function for the RS may be chosen. This is a promising direction for further development of the method.

## Acknowledgments

This work was supported by Grants NCC1253 and NAG11558 from NASA Langley Research Center and by Lockheed Martin Contract SN30G4400R-1. The authors also gratefully acknowledge the technical contributions of Lockheed Martin and NASA Langley Research Center colleagues.

## References

[1] Lasdon, L. S., *Optimization Theory of Large Systems*, Macmillan, New York, 1970.

[2] Wagner, T., and Papalambros, P., "A General Framework for Decomposition Analysis in Optimal Design," edited by B. J. Gilmore, *Advances in Design Automation*, Vol. 2, American Society of Mechanical Engineers, New York, 1993, pp. 315–325.

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

[4] Gage, P., and Kroo, I., "Development of the Quasi-Procedural Method for Use in Aircraft Configuration Optimization," AIAA Paper 92-4693, 1992.

[5] Sobieski-Sobieszczanksi, J., "A Linear Decomposition Method for Large Optimization Problems," NASA TM-83248, 1982.

[6] Altus, S., Kroo, I., and Gage, P., "A Genetic Algorithm for Scheduling and Decomposition of Multidisciplinary Design Problems," *Journal of Mechanical Design*, Vol. 118, No. 4, 1996, pp. 486–489.

[7] Sobieski-Sobieszczanksi, J., "Optimization By Decompostition: A Step From Hierarchic to Non-Hierarchic Systems," NASA CP-3031, Sept. 1988.

[8] Bloebaum, C., "Formal and Heuristic System Decomposition in Structural Optimization," NASA CR-4413, 1991.

[9] Korngold, J., and Gabriele, G., "Integrating Design for Manufacturing of Electronic Packages in a Multidisciplinary Design Analysis and Optimization Framework," AIAA Paper 94-4254, 1994.

[10]Renaud, J., and Gabriele, G., "Improved Coordination in Non-Hierarchic System Optimization," *AIAA Journal*, Vol. 31, No. 12, 1993, pp. 2367–2373.

[11]Seller, R., Batill, S., and Renaud, J., "Response Surface Based, Concurrent Subspace Optimization for Multidisciplinary System Design," AIAA Paper 96-0714, Jan. 1996.

[12]Kroo, I., Altus, S., Braun, R., Gage, P., and Sobieski, I., "Multidisciplinary Optimization Methods for Aircraft Preliminary Design," *Proceedings of the 5th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Vol. 1, AIAA, Washington, DC, 1994, pp. 697–707.

[13]Sobieski, I. P., and Kroo, I. M., "Aircraft Design Using Collaborative Optimization," AIAA Paper 96-0715, Jan. 1996.

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

[15]Braun, R. D., and Kroo, I. M., "Development and Application of the Collaborative Optimization Architecture in a Multidisciplinary Design Environment," *Multidisciplinary Design Optimization: State of the Art, Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization*, edited by N. Alexandrov, and M. Y. Hussaini, 1997, pp. 98–116.

[16]Braun, R. D., Powell, R. A., Lepsch, D. O., and Kroo, I. M., "Comparison of Two Multidisciplinary Optimization Strategies for Launch Vehicle Design," *Journal of Spacecraft and Rockets*, Vol. 32, No. 2, 1995, pp. 404–410.

[17]Braun, R. D., "Collaborative Optimization: An Architecture for Large Scale Distributed Design," Ph.D. Dissertation, Stanford Univ., Stanford, CA, May 1996.

[18]Rodriguez, J., Renaud, J., and Watson, L., "Trust Region Augmented Lagrangian Methods for Sequential Response Surface Approximation and Optimization," *Journal of Mechanical Design*, Vol. 120, No. 1, 1998, pp. 58–66.

[19]Sorenson, D., "Newton's Method with a Model Trust Region Modification," Argonne National Lab., Rept. ANL-80-106, Argonne, IL, 1980.

[20]Alexandrov, N., and Dennis, J. E., "A Trust Region Framework for Managing the Use of Approximation Models in Optimization," NASA TM 112870, 1997.

[21]Saaris, G. R., Tinoco, E. N., Lee, J. L., and Rubbert, P. E., "A502I User's Manual—PANAIR Technology Program for Solving Problems of Potential Flow About Arbitrary Configurations," Document D6-54703, The Boeing Company, Seattle, WA, 1992.

[22]Carmichael, R. L., and Erickson, L. L., "PAN AIR—A Higher Order Panel Method for Predicting Subsonic or Supersonic Linear Potential Flows about Arbitrary Configurations," AIAA Paper 81-1255, June 1981.

[23]Nguyen, D. T.,"Final Report: Finite Element Software for Multidisciplinary Optimization," Dept. of Civil and Environmental Engineering, Old Dominion Univ., NASA Langley Contract NASI-19858, 1995.

A. Messac
*Associate Editor*