MAD Center White Paper No. 1, September 1995

Multidisciplinary Optimization via Intelligent Response Surface Methodology

Rafi Haftka, Bernie Grossman and Bill Mason

Difficulties in System-level Optimization

Modern optimization techniques are mostly sequential in nature, with the optimization algorithm selecting one design at a time, based on information from previous designs. With this optimization approach it is essential that the optimization program can repeatedly call upon the analysis program, with fairly fast response. This in turn requires the integration of the optimization software and the analysis software in a single package.

This level of integration has been accomplished in single component or single discipline software. For example, the NASTRAN program has recently added optimization capabilities, thus allowing efficient optimization of general structures. For system design, the tight coupling between the optimizer and analysis programs presents major difficulties. The analyses of different components and different disciplines are often performed by large proprietary software systems which often run on different computers. Tying all the pieces together is a difficult and never-ending chore because the pieces keep changing.[*]

To overcome these difficulties system designers often use special programs with simplified disciplinary models. For example, in aircraft design, programs like FLOPS and ACSYNT combine simple models of aerodynamics, propulsion, structures and controls. Some models are algebraic equations which can be found in design textbooks, while others are simple numerical models. The simplicity of the models allows them to be put together in a single program and to be connected to an optimizer. This approach is useful at the conceptual design level, but it does not solve the problem of tradeoffs between competing goals of different disciplines later in the design process.

HSCT Design Code

We have been developing research design codes for testing multidisciplinary optimization (MDO) methodologies over the past ten years. We first assembled a design code for subsonic transports, and more recently a design code for the high-speed civil transport (HSCT). In this design code we define the HSCT configuration by 29 variables, which describe the shapes of the wing and fuselage, nacelle positions, tail sizes, as well as define the cruise trajectory and amount of fuel carried by the plane. The objective function is the gross weight of the aircraft required to carry 250 passengers 5500 nautical miles at Mach 2.4.

The performance of the airplane is calculated from various aerodynamic analyses at different flight speeds, obtained from several aerodynamic codes. However, the performance calculation also depends on the structural weight of the airplane, which is estimated by performing structural

[*] The Boeing experience was described by Kathryn M. Chalfan in "A Knowledge System that Integrates Heterogeneous Software for a Design Application," *AI Magazine*, Summer, 1986, pp, 80-84. ".. a procedural program that subsumed several technology codes was built. This program required changes to the technology codes. It was not maintainable because the individual groups continued to alter their technology code programs separately."

optimization for each aerodynamic configuration. To perform these calculations we have tied together a collection of analysis codes that predict different components of aerodynamic response, as well as optimize the structural weight. All the aerodynamic codes were assembled in one large computer code, while communication with the structural optimization program has been manual.

Most of the computer codes we have assembled represent simplified analysis methodologies, are relatively small (thousands of lines or less), and are not proprietary. Still, we experienced many difficulties and much grief in the process of assembling the pieces into a single MDO package. Some of the pieces of the code kept changing, while others required changes as we changed computer systems. Programs were written in FORTRAN 66, FORTRAN 77, C, C++ and Mathematica. Some programs were old, and when changes needed to be made, their developers were not available to answer questions about them.

As a result of these problems we found that the reliability of our code was poor, and that an ever larger share of our time was being devoted to maintaining the code. Furthermore, we found that we needed to keep training new students about the details of segments of the code in which they had no interest, simply so that they could add their own contributions without adding bugs. Students were frustrated by the experience of working with a large code about which they knew little, and their productivity declined.

We also ran into an entirely different problem, which proved to be almost intractable. Some of the analyses produced response with a small amount of noise, which was perfectly acceptable for the purpose of a single analysis. However, this numerical noise created artificial local minima in design space, so that our optimization algorithm had great difficulties finding good designs. Tracking down the various sources of numerical noise and eliminating them proved to be extremely difficult, even though we had access to the source codes. The reason for this difficulty was that numerical noise is often created by numerical processes that are not documented, and tracking them down requires familiarity with the details of the code.

The problems we encountered are not unique.[*] They are common problems with many engineering software system development, and software engineering research has yet been unable to provide practical solutions.

Faced with these difficulties, we have turned to the traditional aircraft design approach for solutions. We already were using traditional simple analysis models along with more complex modern numerical codes for improved efficiency of the MDO process, a technique which we call variable complexity modeling (VCM). We now examine the process of traditional aircraft design for a solution to the problems of combining the pieces of an MDO process and filtering out the numerical noise.

Traditional design approach

An examination of the process of traditional aircraft design reveals that aircraft designers faced a very similar dilemma to the one we encountered. The analysis tools available forty years ago were computationally inexpensive, but often required specialized expertise beyond what could be expected of the generalists who practiced the art of aircraft design. Therefore the analysis tools were extensively exercised by their own developers to produce design charts and carpet plots which could be directly used by designers. That is, when a researcher developed a new method, he normally produced copious tables and charts, which permitted designers to

[*] See for example the rare candid discussion by Karl A, Geiselhart in "A Technique for Integrating Engine Cycle and Aircraft Configuration Optimization," NASA CR 191602, Feb. 1994, especially page 30.

benefit from the new capability without needing to master the new technique. Additionally, experimental results complemented the analytically generated charts, or were used to generate charts of correction factors to be applied to the analytically generated charts.

This process of giving the designer the results of the analysis tools rather than the tools themselves is somewhat limited in that design charts and tables are manageable only when the response is a function of a small number of variables, preferably no more than two. To reduce the number of parameters appearing in design charts, researchers put a lot of effort into compressing results with the aid of nondimensional similarity parameters. Examples of such parameters are aspect ratio and taper ratio in the definition of geometry, Mach number and Reynolds number in the definition of the aerodynamics, and the Batdorf `Z' parameter and beam slenderness ratio in structures. The same approach is used to this day in reporting experimental results. Many experimentalists strive to discover combinations of variables that will allow them to collapse a large number of experimental observations into a small number of graphs. This process also filters out much of the noise due to experimental errors.

The process of discovering similarity parameters, compressing the information, and presenting it in terms of design charts or simple formulas also provides much insight into the influence of various parameters on the design and the basic tradeoffs associated with design decisions. Also, the chance of errors greatly diminishes when large amounts of results are produced through a range of parameters, because all the results need to fit together. In contrast, a modern designer, with only limited knowledge of the complex software codes that he or she needs to exercise, is much more error prone.

We plan to develop a similar approach to solve our problem of integration of analysis software. This approach is based on response surface (RS) methods and variable complexity modeling (VCM) in a combination we call intelligent response surface (IRS) design.

Response surface approach to design

RS methods are techniques for approximating functions based on their values at a number of points. Usually the approximation takes the form of a low-order polynomial, however, even neural networks can be viewed as a special case of response surface approximations. RS methods can be used to reintroduce the traditional design paradigm of giving designers the results of analysis tools rather than the tools themselves.

This can be accomplished by running a large number of analyses for different sets of inputs on each analysis code. Then, the results are fitted by a response surface (say quadratic polynomial) in terms of the input variables, and the designer is given the response surface instead of the analysis code. With the RS being a simple formula instead of a design chart, the limit of 2 or 3 variables is removed, so that in theory any number of variables can be used. Additionally, the creation of the RS, which is essentially a curve fit operation, can be used to filter out noise in the data.

This approach does not require tying an optimizer to analysis programs because the optimizer operates on the RS rather than the original data. RS techniques are therefore well suited for working with black-box codes that cannot be incorporated easily into larger systems. The approach also respects organizational boundaries in that response surfaces can be generated by various organizations using their favorite computer codes on their own computers. Finally, with a large number of analyses that need to be executed for sets of predetermined data points, maximum use can be made of parallel computation with minimal need to change codes to take advantage of parallelization.

Unfortunately, when the number of variables associated with the response surface becomes large, we encounter the so-called curse of dimensionality. Even with quadratic polynomials the number of coefficients increases as the square of the number of variables. The number of analyses required to evaluate these coefficients increases in a similar manner. Furthermore, often the accuracy of the response surface deteriorates with increasing dimensionality. Therefore, the brute-force application of response surfaces to aircraft design is useful only when the number of variables defining the design is small, typically less than 10 to 20. When the number of variables is larger, a more intelligent use of RS techniques is warranted.

Intelligent response surfaces

To combat the loss of accuracy associated with high dimensionality we customize the response surface to a small region in design space tailored to a specialized design problem. For example, while old design charts provide drag coefficients for what was considered to be all likely planforms, we need information only for planforms which are reasonable for a particular flight Mach number. In that we imitate traditional designers. Unlike modern optimization programs, traditional designers did not waste time analyzing designs which are patently nonsensical.

To limit the design space we employ the simpler analysis tools of the previous generation of designers, tools which have lower accuracy than their modern counterparts, but which require only minuscule amount of computation on present-day computers. These tools may not be accurate enough to identify optimal or near optimal designs. However, they can identify vast regions in design space which correspond to nonsense designs that should not be analyzed by more expensive modern techniques.

To combat the increased cost associated with high dimensionality we seek to reduce the dimensionality of the response surface. This can be accomplished by imitating the process that traditional analysts employed in order to compress their results into two-dimensional charts. That is, we seek the important similarity parameters that characterize each response quantity. Here again we are aided by older techniques which were devised by traditional analysts seeking the economy of presentation in their results. So these parameters often may be identified by inspection alone of the formulae or equations used in older methods. When this is not enough, numerical experimentation is easy because of the low cost of the analysis. Thousands of analyses can be run, and statistical techniques, such as principal factor analysis, used to identify appropriate combinations of variables.

We have been working on response surfaces for several components of our HSCT code. The following description of the construction of a response surface for estimating wing structural weight illustrates the principles of IRS.

Example: Response surface for estimating HSCT wing structural weight

We started the process of creating a response surface for the results of the HSCT structural optimization by a study of the FLOPS weight equations. These equations are mostly based on simple theory, augmented by historical data and by structural optimizations on the TSO program. Our previous experience indicated that the results of FLOPS correlate well with those of the structural optimization. However, differences of about 50 percent were still common for many design configurations. We hoped to create a more accurate weight equation by customizing it to a particular set of design requirements (range, Mach number, number of passengers, etc.).

To see how well a simple polynomial can approximate the wing structural weight we tested the concept on the FLOPS weight equation. Of our 29 configuration design variables, 25 affect

the wing structural weight, and so we tried to fit a quadratic polynomial in 25 variables (with 351 coefficients) to the FLOPS weight. We started with a fairly large design space by permitting the design variables to vary by ±80% about our baseline design. In this design space the approximation by the quadratic polynomial failed miserably, with an average error of about 25,000 lb.

Next we limited the design space to exclude patently nonsensical configurations. We limited our consideration to design points where the FLOPS weight equations predict wing structural weights between 20,000 and 120,000 pounds. Since the expected weight is between 30,000 and 40,000 pounds, we felt that this was a reasonable range even based on a simple weight equation. We added a similar constraint, based on simple analysis techniques, on the range, and a host of geometrical constraints germane to our design problem. With these constraints the design space shrunk considerably, and the average error reduced to about 50 pounds!

Then we examined the equations used by FLOPS to estimate wing structural weight. Our examination revealed that FLOPS uses ten variables to estimate the weight. Some relate to the geometry of the wing, some to the aerodynamic loading and some to the nonstructural weights. Using these 10 variables instead of the original 25, we reduced the number of coefficients in the response surface from 351 to 66, and obtained additional improvement in accuracy. This means that we needed to execute about 100 structural optimizations in order to fit the polynomial and filter out numerical noise instead of about 500 for the original variables.

Finally we created the response surface based on the structural optimization, we found an average error of about 1500 pounds or 5 percent of the average wing structural weight. The error was reasonable, and with the use of a parallel computer, the time required for 100 structural optimizations was less than 24 hours.

Future Directions - Proposed to NASA: Disaggregation of HSCT MDO Code

We propose to use IRS to disaggregate the elaborate HSCT MDO design code, and create a much smaller code based on intelligent response surfaces. In the final code, the objective function and all the constraints will be calculated from the response surface approximations. These response surfaces will be created independently by the disciplinary codes now incorporated into the HSCT code. The optimization will involve only simple objective function and constraint functions that are extremely cheap to evaluate. The low computational expense will allow us to perform a much more thorough optimization than we currently afford. In particular, we will be able to perform global optimization, multi-criterion optimization and reliability based optimization.

Global optimization techniques will protect us from finding local optima that depend on the initial design point. This protection is sorely needed because the constrained design space we work in is highly convoluted and non-convex. Multi-criterion optimization will allow us to look at how to reconcile different possible objective functions such as initial cost, operating cost, and lifetime cost, rather than use the compromise gross weight objective which we presently employed. Finally, probabilistic optimization, will allow us to minimize the adverse effects of uncertainties in the data presently available on material properties, engine performance, etc.

In the business world, the term disaggregation has been used to denote the processes of spinning off peripheral divisions and outsourcing. Disaggregation permits a company to concentrate its efforts on core areas where its managers are most competent. When we spin off codes and replace them by response surfaces we similarly focus our efforts on engineering rather than software management.

Specifically, we will need to focus our efforts on the creation of intelligent response surfaces for the various aerodynamic performance quantities, such as drag coefficients and required tail size for trim. This will be accomplished by a thorough examination of the relevant equations and the simple approximations available, as well as extensive numerical experimentation.

The learning accompanying the process of creating intelligent response surfaces will hopefully be an important side benefit of the effort. Our present MDO process enhances the skills of our graduate students in struggling with unfriendly computer programs. Hopefully, the new process will do more to enhance their understanding of the process of aircraft design.