Optimization of Complex Mechanics Simulations with Object-Oriented Software Design[*]

M.S. Eldred[†], D.E. Outka[‡], W.J. Bohnhoff[§], W.R. Witkowski[¶], V.J. Romero[#], E.R. Ponslet[**], and K.S. Chen[††]

Sandia National Laboratories[‡‡]
Albuquerque, NM 87185

Abstract

The benefits of applying optimization to computational models are well known, but their range of widespread application to date has been limited. This work attempts to extend the disciplinary areas to which optimization algorithms may be readily applied through the development and application of advanced optimization strategies capable of handling the computational difficulties associated with complex simulation codes. Towards this goal, a flexible software framework is under continued development for the application of optimization techniques (and other iterative computational methods) to broad classes of engineering applications, including those with high computational expense and nonsmooth, nonconvex design space features. Object-oriented software design with C++ has been adopted as a tool to provide a flexible, extensible, and robust multidisciplinary toolkit that establishes the protocol for wrapping parameter optimization around computationally-intensive simulations. The object-oriented approach is well-suited for handling this large software undertaking, in which a wide assortment of optimization algorithms, approximation techniques, and hybridized strategies must be generically interfaced with broad classes of analysis capabilities. Demonstrations of optimization using the software are presented in fluid mechanics, heat transfer, nonlinear solid mechanics, and structural dynamics. Optimal results are presented along with technical lessons that were learned in the optimization process.

Introduction

Computational methods developed in fluid mechanics, structural dynamics, heat transfer, nonlinear large-deformation mechanics, manufacturing and material processes, and many other fields of engineering can be an enormous aid to understanding the complex physical systems they simulate. Often, it is desired to utilize these simulations as virtual prototypes to improve or optimize the design of a particular system. This effort seeks to enhance the utility of this broad class of computational methods by providing them with a general optimization capability and enabling their use as design tools, so that simulations may be used not just for single-point predictions, but also for improving system performance in an automated fashion. System performance objectives can be formulated to minimize weight or defects or to maximize performance, reliability, throughput, reconfigurability, agility, or design robustness (insensitivity to off-nominal parameter values). A systematic, rapid method of determining these optimal solutions will lead to better designs and improved system performance and will reduce dependence on prototypes and testing, which will shorten the design cycle and reduce development costs.

[*]Presented as paper AIAA-95-1433-CP at the 36th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, New Orleans, LA, April 10-13, 1995.

[†]Senior Member of Technical Staff, Structural Dynamics Department, Mail Stop 0439.

[‡]Senior Member of Technical Staff, Navigation Guidance and Control System Department, Mail Stop 1174.

[§]Senior Member of Technical Staff, Intelligent Systems Department, Mail Stop 1177.

[¶]Senior Member of Technical Staff, Structural Dynamics Department, Mail Stop 0439.

[#]Senior Member of Technical Staff, Thermal Sciences Department, Mail Stop 0835.

[**]Amparo contractor, Structural Dynamics Department, Mail Stop 0439.

[††]Senior Member of Technical Staff, Incompressible Fluid Dynamics Department, Mail Stop 0826.

[‡‡]P.O. Box 5800, Albuquerque, NM 87185, USA. This work performed at Sandia National Laboratories supported by the U.S. Department of Energy under contract DE-AC04-94AL85000.

Towards these ends, we have targeted the needs of a broad class of computational methods in order to provide a general optimization capability. Much work to date in the optimization community has focused on applying either gradient-based techniques to smooth, convex, potentially expensive problems (e.g., (Kamat 1993)) or global techniques to nonconvex but inexpensive problems (e.g., (Törn and Zilinskas 1989)). When the difficulties of high computational expense and nonsmooth, nonconvex design spaces are coupled together, advanced strategies are required. Moreover, since the challenges of each application are frequently very different, generality and flexibility of the advanced strategies are key concerns. The following list itemizes the primary challenges.

Technical Issues. The coupling of optimization with complex computational methods is difficult, and optimization algorithms often fail to converge efficiently, if at all. The difficulties arise from the following traits, shared by many computational methods:

1. The time required to complete a single function evaluation with one parameter set is large. Hence, minimization of the number of function evaluations is vital.
2. Analytic derivatives (with respect to the parameters) of the objective and constraint functions are frequently unavailable. Hence, sensitivity-based optimization methods depend upon numerically generated gradients which require additional function evaluations for each scalar parameter.
3. Convergence tolerances in embedded iteration schemes introduce uncertainty (noise) in the function evaluation response surface, which can result in inaccurate numerical gradients.
4. The parameters may be either continuous or discrete, or a combination of the two.
5. The objective and constraint functions may not be smooth or well-behaved; i.e., the response surfaces can be severely nonlinear, discontinuous, or even undefined in some regions of the parameter space. The existence of several local extrema (multi-modality) is common.
6. Each function evaluation may require an "initial guess." Function evaluation dependence on the initial guess can cause additional uncertainty in the response surface. Moreover, a solution may not be attainable for an inadequate initial guess, which can restrict the size of the allowable parameter changes.

Addressing these challenges with robust and efficient optimization strategies extends the range of applications where the benefits of optimal solutions can be realized.

Technical Approach. To be effective in addressing these technical issues, one must minimize the computational expense associated with repeated function evaluations (efficiency) and maximize the likelihood of successful navigation to the desired optimum (robustness). The key technology developments needed to achieve these goals are fundamental algorithm research, hybrid optimization algorithms, function approximation strategies, parallel processing, and automatic differentiation. Research in hybridization, approximation, and parallel processing is detailed in a separate paper (Eldred et al. 1996).

In this paper, the software infrastructure design and demonstrations of its use in four engineering mechanics applications will be presented. The generation of optimal solutions for the four applications involves mating existing, stand-alone optimizers (nonlinear programming, genetic algorithms, pattern search) with one or more engineering simulations. Thus, the focus of this paper is on 1) how generic interfacing of iteration with simulation is performed, 2) what application-specific techniques are useful in enabling reliable, efficient optimization studies, and 3) how existing techniques perform and what their weaknesses are when interfacing with complex engineering simulations. The results computed serve as benchmarks for comparison of advanced strategy performance (Eldred et al. 1996), and the lessons learned have helped direct the current research focus areas.

Software Design

The DAKOTA (Design Analysis Kit for OpTimizAtion) toolkit utilizes object-oriented design with C++ (Stroustrup 1991) to achieve a flexible, extensible interface between analysis codes and iteration methods. The scope of iteration methods which may be included in the system currently includes optimization, nondeterministic simulation, and parameter study methods. Likewise, there is breadth in the analysis codes which may be interfaced. Currently, simulator programs in the disciplines of nonlinear solid mechanics, structural dynamics, fluid mechanics, and heat transfer have been accessed. The system also provides a platform for research and development of advanced iteration strategies.

Accomplishing the interface between analysis codes and iteration methods in a sufficiently general manner poses a difficult software design problem. These conceptual design issues are being resolved through the use of object-oriented programming techniques. In mating an iteration method with an analysis code, generic interfaces have been built such that the individual specifics of each iterator and each analysis code are hidden. In this way, different iterator methods may be easily interchanged and different simulator programs may be quickly substituted without affecting the internal operation of the software. This isolation of complexity through the development of generic interfaces is a cornerstone of object-oriented design (the concept of "one interface, many methods"), and is required for the desired generality and flexibility of the advanced strategy developments (e.g., hybridization, function approximation).

The Application Interface (Figure 1) isolates application specifics from an iterator method. By providing a generic interface for the mapping of a set of parameters (e.g., the vector of design variables "OptParameter") into a set of responses (e.g., an objective function, constraints, and sensitivities in "OptResponse"), the specific complexities of a given problem are hidden from the iterator method. Housed within the Application Interface are three pieces of software. The input filter abstraction ("IFilter") provides a communication link which transforms the set of input parameters (OptParameter) into input files for the simulator program. The simulator program reads the input files and generates results in the form of output files or databases (a driver program/script is optional and is used to accomplish nontrivial command syntax and/or progress monitoring for adaptive simulation strategies). Finally, the output filter abstraction ("OFilter") provides another communication link through the recovery of data from the output files and the computation of the desired response data set (OptResponse). The following descriptions identify the actual C++ classes used by DAKOTA:

Optimizer: This class represents the optimization technique to be used. Many optimizers may be derived from this class, enabling easy selection of a particular optimizer as a problem may require. The **Optimizer** base class is derived from a more general **Iterator** class.

ApplicationInterface: This abstraction defines the interface between an Optimizer and a simulator program. It encompasses the specific details of a given engineering application. Everything external to this interface is generic and independent of the problem being solved. An ApplicationInterface object will *use* both an IFilter object and an OFilter object to accomplish its task.

IOFilter: A utility class abstraction used by the ApplicationInterface class to provide communication links between the generic data formats used by the Optimizer and the specific input and output formats of a particular simulator program. The input and output operations are logically similar but separable enough to warrant two derived classes (IFilter and OFilter).

OptParameter: A vector of floating point values representing the parameters being optimized.

OptResponse: An abstraction for storing the desired output data set of a simulation. OptResponse contains values for the objective function, constraints, and (in some applications) sensitivities.

Object-oriented techniques such as inheritance and polymorphism are being exploited so that the abstract objects are easy to use and sufficiently generic to encompass a wide variety of problems. Having properly designed the interface, the mapping of parameters to responses shown in Figure 1 provides generic information to the optimizer, and the application and implementation specifics are hidden. The result is a flexible, reusable, and robust multi-disciplinary toolkit that establishes the protocol for wrapping parameter optimization around computationally-intensive finite element analyses.

Optimizer iterators are part of a larger "iterator" hierarchy in the DAKOTA system. In addition to optimization algorithms, the DAKOTA system is designed to accommodate nondeterministic simulation and parameter study iterators. Other classes of iterator methods may be added as they are envisioned, which "leverages" the utility of the Application Interface development.The inheritance hierarchy of these iterators is shown in Figure 2. Inheritance enables direct hierarchical classification of iterators and exploits their commonality by limiting the individual coding which must be done to only those features which make each iterator unique.

Several optimization algorithm libraries and strategies are inherited from the **Optimizer** base class. DOT (Vanderplaats Research and Development 1995), NPSOL (Gill et al. 1986), OPT++ (Meza 1994), and SGOPT (Hart 1994, Hart 1995) have been incorporated in this framework as libraries of *stand-alone* optimizers. Additionally, the "Hybrid" and "SAO" optimization strategies are *combination* strategies which have been conceptualized. In the former, two or more stand-alone optimizers are combined in a hybrid strategy. For example, a coarse-grain genetic algorithm might initially be used to locate promising design space regions, followed by the use of nonlinear programming to converge efficiently on local optima. Effective switching metrics are a key concern. In the latter, a stand-alone optimizer is interfaced with a separate function approximation toolbox in the setting of sequential approximate optimization (SAO, see (Haftka et al. 1990)). Here, the accuracy and expense of the approximate subproblems, the mechanisms by which the approximations are updated, and the mechanisms of move limit enforcement are key concerns.

Software development work is ongoing. In addition to extension of iterator capabilities and incorporation of additional simulator programs through input and output filter development, general software infrastructure extensions are being implemented in the areas of active set strategies, support of analytic sensitivities, advanced problem specification and system control, and general restart capabilities.

Applications and Results

In application work, the targeted technology areas are nonlinear large-deformation solid mechanics, heat transfer, fluid mechanics, and structural dynamics.

Nonlinear Mechanics: Shape Optimization of a Hazardous Material Transportation Cask

Problem Description. Design of hazardous material transportation casks is an area where numerical optimization can have a large and immediate impact (Harding et al. 1995). These casks are used to transport spent nuclear fuel, high-level waste, and hazardous material. Since they transport such materials, their design and certification must adhere to strict Nuclear Regulatory Commission regulations. A typical transportation cask is shown in Figure 3. There are several components that constitute a typical cask, including impact limiters, containment vessel, shielding, and closure mechanisms, whose designs could be numerically optimized.

In the past, typically, each component was designed separately based on its driving constraint and the expertise of the designer, the components were assembled, and then modified until all of the design criteria were met. This approach neglects the fact that, in addition to its primary function, each component can also have secondary purposes. For example, an impact limiter’s primary purpose is to act as an energy absorber and protect the contents of the package, but it can also act as a heat dissipater or an insulator. However, designing the component to maximize its performance with respect to both objectives severely convolutes the problem. Numerically-based optimization schemes can readily attack such problems in an efficient manner. Thus, since the design of these packages involves a complex coupling of structural, thermal and radiation shielding analyses and must follow very strict design constraints, numerical optimization provides the potential for more efficient and robust container designs.

The container weight is to be minimized with respect to shape design variables, subject to design constraints on the cask performance in fire and impact accident simulations. The shape of the cask has been parameterized with respect to 6 design variables (x_1 - x_6) which control the thicknesses of 3 overpack layers in the radial and axial directions (Figure 4). Lower and upper bounds for each of these 6 design variables are 0.1 inch and 20 inches, respectively. The finite element model in Figure 4 shows a simplified geometry over that of Figure 3 in which the stainless steel overpack closure and the locking mechanisms are neglected. The 3 overpack layers consist of 1 layer of aluminum wire mesh impact limiter (density = 448.5 kg/m^3) sandwiched between 2 layers of ceramic cloth thermal insulation (density = 801.0 kg/m^3). The optimizer competes these overpack layers against each other based on relative weight and effectiveness in satisfying the certification constraints. Three separate accident conditions are of concern, each supplying a design constraint on the optimization. In the first accident scenario, a 500 kg steel plate is dropped from a height of 9 m onto the end of the cask which is supported on a rigid foundation (the “end-on” impact; Figure 4). The nonlinear mechanics code PRONTO2D (Taylor and Flanagan 1986) is used to determine stress histories for a given cask design, and for the end-on impact, a design constraint enforces an allowable of 23,000 psi on the maximum axial stress (σ_{yy}) in the inner container. Second, for the same plate impact in a side-on configuration (Figure 5), a design constraint enforces an allowable of 8,440 psi on the maximum inner container axial stress (σ_{yy}). These stress allowables were obtained through calibration to a highly refined model, in which a detailed mesh captured actual threaded seal deformation. Lastly, for a 30 minute 800^o C fire, COYOTE II (Gartling and Hogan 1994) is used to generate nodal temperature histories, and a design constraint enforces that the maximum inner seal temperature does not exceed 232^o C.

An input filter program was generated to translate the shape design parameters into information used by the PRONTO2D and COYOTE II analysis codes. This requires preprocessing of parameterized template input decks (with APREPRO (Sjaardema 1992)) and automatic mesh generation (with FASTQ (Blacker 1988)). An output filter program computes the weight, reads the stress and temperature time histories (using BLOT (Gilkey and Glick 1989)), locates the maximum stresses and temperatures, computes constraint values, and returns the objective and constraint function values to the optimizer.

To maximize efficiency with respect to simulation duration while still maintaining the ability to reliably capture the complete impact event, an adaptive termination time strategy was developed and implemented for the impact analyses. This development was motivated by the observation that, whenever event durations vary broadly with design variables, a single selected termination time will invariably be either too long, wasting CPU cycles in continuing the simulation past the event of interest, or too short, terminating the simulation before the peak response is reached and causing inaccurate objective and constraint function evaluations. To avoid the more serious ramifications of the second scenario (underestimation of a critical response), it is common to sacrifice efficiency and adopt a best guess at a sufficiently long simulation duration. Of course, this approach can fail if the variance of event duration over the design space is underestimated; and in fact, an optimizer will naturally seek out those regions of the design space in which the simulation duration is insufficient if the truncation of analyses leads to lower objective functions or more feasible constraint values. A better approach is to adaptively control simulation duration through the monitoring of simulation progress. In impact analyses, event completion can be determined by monitoring the kinetic energy (KE) time history for rebound (an increase in KE following the near-zero minimum state), after which the simulation can be terminated with a Unix kill process command. The KE monitoring and process kill is accomplished with a Unix background process which is launched from the PRONTO analysis driver (see Figure 1) and which cycles with a sleep-delay. This strategy was highly effective in conserving compute time while guaranteeing capture of the peak stress.

Optimization Results. Nonsmoothness of response variations with respect to design variables is troublesome for nonlinear programming techniques, especially when sensitivities are obtained by finite difference. For the end-on vertical impact analysis, continuous improvements in analysis, including model refinement, filtering of stress time histories, increases in platform precision (from single to double precision FORTRAN), and modifications in contact line treatment have been necessary in order to minimize nonsmoothness. Figure 6 shows sequential improvement in design space smoothness for variation in maximum axial stress (σ_{yy}) with respect to fine changes in vertical impact limiter thickness (design variable x_5), from unfiltered low-precision runs (plot point ‘o’) to filtered low-precision runs (plot point ‘*’) to filtered high-precision runs on a refined model (plot point ‘x’) to filtered high-precision runs on a refined model with contact lines node-locked at corners (plot point ‘+’). This last modeling improvement was needed to remove contact indeterminacies at mesh corners, since these indeterminacies were exciting hourglassing instabilities in the PRONTO stress histories. Clearly, the final response variations provide a far more navigable surface than the initial variations. For the thermal analysis, similar refinements have been required, as shown in Figure 7. Mesh refinements, time step size decreases, and the tightening of iterative solver convergence tolerances were required to progress from the initial stair-stepped curve through the sinusoidal curve to the final, relatively smooth, response variation. These nonsmoothness reductions in the impact and thermal analyses were required to allow for effective design space navigation with gradient-based optimizers.

With the bulk of the troublesome nonsmoothness removed, optimization studies have been successful in minimizing the cask weight with respect to the end-on constraint, the thermal constraint, and all three constraints in the combined problem (no meaningful stand-alone weight minimization problem exists for the side-on impact model since x_4, x_5, and x_6 are not defined; see Figure 5). Minimum weights and constraint values are shown in Table 1 (active constraints are underlined), the associated design variable values are shown in Table 2 (variables at or near their bounds are underlined), and the optimal shapes are graphically compared to the geometry of a successful experimental prototype in Figure 8. In Table 1, as would be expected, the thermal and end-on optimum designs are active on their respective constraints. The combined optimum is active on the thermal and end-on constraints, and inactive on the side-on constraint. In Table 2, it can be seen that the end-on constraint primarily drives axial stroke length (x_4, x_5, and x_6) and radial wire mesh thickness (x_2), and the thermal constraint primarily drives radial thickness of the thermal shields (x_1 and x_3). In the combined design, it can further be seen that the outer thermal shield is redundant (x_3 and x_6 go to their 0.1 inch lower bounds). The fact that the necessary thickness of thermal shield belongs in the innermost layer is intuitive, since the thermal shield is heavier than the wire mesh and gains no obvious thermal advantage in being positioned further out radially. The wire mesh, on the other hand, is lighter, pays a smaller weight penalty for being the external layer, and gains a mechanical advantage in being further separated from the centroidal axis. The 94 lb. combined optimum design is a substantial improvement over both the successful experimental prototype containing 150 lbs. of overpack (Figure 8) and the previously published best design of 184 lbs. (Witkowski et al. 1994).

Technical Lessons Learned. The application of optimization to this problem led to several observations:

- Shape optimization with automatic remeshing is a tricky business, particularly when design variables are inclined to seek their lower bounds. Poor element aspect ratios can promote numerical instabilities in nonlinear solvers, which must ultimately be addressed with increased mesh density. That is, as a shape design variable approaches its lower bound, the characteristic element size in the region defined by the shape variable must decrease in order to accurately compute the desired responses. This can dramatically increase the compute time, both directly, through the increase in total degrees of freedom, and indirectly, through the determination of stable time step sizes.
- An effective balance of nonsmoothness vs. CPU has to be determined in many engineering simulations. One must have navigable objective and constraint function surfaces, but must also have tractable CPU usage in the simulations. Clearly, these are competing factors, since refining the modeling controls and mesh density invariably increases the CPU usage of a simulation. Furthermore, the level of required smoothness may be a function of the type of optimizer being used. This fact is further born out in the following heat transfer application.
- Adaptive simulation termination strategies increase both the efficiency and the robustness of optimization studies, by conserving compute time while still guaranteeing capture of the peak responses of interest.

Heat Transfer: Determination of Worst Case Fire Environments

Problem Description. In thermal science simulations, parameter sets are sought which produce worst-case credible fire environment(s) for which structures and systems (such as aircraft, weapons, or petrochemical processing plants) must be designed. These inverse problems can be solved within an optimization framework. As a demonstration, optimization techniques have been applied to determine the vulnerability of a safing device to a “smart fire” (Romero et al. 1995). The optimization parameters consist of the location and diameter of a circular spot fire impinging on the device. The temperature of the fire is constant, though the heat flux it imparts to the device varies in time and space coupled to the response of the device. Function evaluations involved transient simulations using a nonlinear QTRAN (PDA Engineering 1993) heat transfer model with radiative and conductive heating. The finite element model used in the analysis is shown with typical temperature contours in Figure 9. Each simulation required 20 CPU minutes to solve (at tight error tolerance levels, see Figure 13) on a node of an IBM SP2 workstation.

The components of interest must work together to prevent the device from operating except under the intended conditions. It is a weaklink/stronglink design: the weaklink is designed to fail under adverse conditions, which renders a potential stronglink failure incapable of harm. The weaklink is a Mylar-and-foil capacitor winding mounted on the outside of the safing device and the stronglink is a stainless steel plate mounted inside a cavity and offset right of center as shown in Figure 9. The time difference between failure of the stronglink and failure of the weaklink is the safety margin for the device and varies with the fire exposure pattern on the device surface. Hence, to validate the design of the safing system, the worst-case fire exposure pattern is sought by using optimization to minimize this safety margin for selective exposure to a 1000O C black body heat source.

Typical critical node temperature histories are shown in Figure 10 for a 20 hour fire exposure, where the critical node of a link is the one which reaches its failure temperature earliest. The safety margin shown graphically is the objective function that the optimizer minimizes with respect to the design parameters of fire spot-radius (r) and fire center location (x), subject to simple bounds ($0.5 \leq r \leq 5.8$, $-2.9 \leq x \leq 2.9$).

An input filter program was generated to translate the optimization parameters into information used by the QTRAN analysis code. Node, element, and surface information are obtained from the PATRAN (PDA Engineering 1988) neutral file description of the finite element model. The heating load applied to each exposed element is then calculated, and this load is assigned to the corresponding nodes of the mesh. QTRAN is executed to determine the temperature histories of monitored nodes for up to a 200 minute exposure to these heating conditions. The QTRAN simulation rarely runs for the full 200 minute duration because of the use of an adaptive termination strategy. This strategy uses a background script to monitor the simulation progress periodically and to execute a kill command once failures have been captured for both links. This procedure is used to reduce the computational expense of the simulations. Once the QTRAN simulation is completed, an output filter program reads the nodal temperature histories and calculates a failure time for each node. In this calculation, interpolation between time step values is used to accurately estimate captured link failures, and if the analysis ran the full 200 minute duration without capturing failures for both links, then a linear extrapolation in time is used to approximate the uncaptured link failures. The earliest weaklink nodal failure time is then subtracted from the earliest stronglink nodal failure time, and this objective function value is returned to the optimizer.

Optimization Results. This application was challenging from an optimization perspective due to the nonsmoothness and nonconvexity of the design space. In Figures 11 and 12, one-dimensional parameter studies show evidence of multimodality (Figure 12) and of slope-discontinuity at the minimum (Figures 11 and 12). The slope-discontinuity in the figures is caused by switching of the critical weaklink node between geometric extremes. Figure 11 shows negative curvature near the discontinuity (nonconvexity), whereas the discontinuity in Figure 12 is more difficult to discern since the curvature is positive near the discontinuity.

These two parameter studies also provide insight into the mechanics of the problem. In Figure 11, the offset of the lowest safety margin from x=0 (the center of the device) backs up engineering intuition in that the stronglink is also offset right of center. That is, a fire centered roughly over the stronglink causes a lower safety margin. Figure 12 shows a less intuitive result, in which it is evident that the lowest safety margin is not achieved with either a large fire or a small fire. Rather, there exists an insidious, medium sized fire which is not so small that the heating rate is insufficient and which is not so large that it prevents selective heating.

Figure 13 is a detail of Figure 11 and shows evidence of small-scale nonsmoothness, which was reduced through the tightening of QTRAN convergence tolerances at the cost of approximately an order of magnitude greater computational time per analysis. EPSIT and EPSIT2 are absolute convergence tolerances in degrees which govern time step completion and node inclusion in nonlinear iterations, respectively. The additional computational expense per analysis was warranted in this case since none of the nonlinear programming algorithms could successfully navigate the design space without reducing the nonsmoothness (even with large finite difference step sizes).

Several nonlinear programming optimization packages were employed for the solution of this problem, including DOT, NPSOL, and OPT++. In general, the Newton-based optimizers (NPSOL's sequential quadratic programming algorithm, DOT's BFGS quasi-Newton method, and OPT++'s quasi-Newton methods) performed poorly due to inaccuracy and ill-conditioning in the Hessian approximation caused both by the nonconvexity of the design space and by the relatively large finite difference step sizes needed to overcome the small-scale nonsmoothness. Conjugate gradient (CG) methods (DOT's Fletcher-Reeves CG and OPT++'s Polak-Ribiere CG) were much more successful. Furthermore, choice of finite difference step size (FDSS) for computation of gradients proved to be important. Table 3 shows the results for CG optimizers with varying FDSS and with EPSIT=10^{-2} and EPSIT2=10^{-4} (see Figure 13 for effect of EPSIT tolerances). An asterisk (*) in the function evaluations column indicates that the optimization terminated prematurely due to a search direction that failed a descent direction test.

The first four rows of Table 3 illustrate the effect of FDSS on the optimization: FDSS should be as small as possible to allow for effective convergence to a minimum (0.1% is better than 1% which is better than 4% since the gradients are less accurate locally for larger FDSS), but still large enough that small-scale nonsmoothness does not cause erroneous gradients (0.01% is too small; the optimizer cannot successfully navigate the design space since the FDSS is on the order of the design space noise). The last five rows show that DOT's CG optimizer was more robust and more efficient than OPT++'s CG optimizer, through the fact that DOT was successful from 3 different starting points and OPT++ from only one, and through the lower number of function evaluations that were required. The chief cause for these differences was not the version of CG being used (in fact, Polak-Ribiere is generally regarded to be superior to Fletcher-Reeves (Coleman and Li 1990)), but rather was DOT's robust line search routine. OPT++'s line search routine was tuned for efficiency in smooth applications and was overly aggressive in this application; the OPT++ line search library has since been extended to include more robust routines for nonsmooth applications.

To achieve the best answer possible, the QTRAN convergence tolerances were tightened 2 additional orders of magnitude (EPSIT=10^{-4}, EPSIT2=10^{-6}) and the FDSS was reduced to 0.1%. DOT's Fletcher-Reeves CG algorithm was used to obtain the lowest objective function value of 2.5309 minutes (r=1.6204, x=0.78205) which, when compared to stronglink and weaklink failure times of 62.743 and 60.212 minutes respectively, corresponds to a safety margin of just 4%. The 2.5 minute safety margin is an order of magnitude lower than anticipated prior to the study, meaning that the safing device has been shown to be much more vulnerable than was expected. With this relatively low safety margin, it becomes crucial to assess the effects of nondeterminism in the model, which, in the future, can be accomplished with minimal additional effort by "instantiating" an iterator from the **Nondeterministic** base class (see Figure 2). Thus, optimization has successfully solved the difficult problem of worst-case design safety and suggests that design improvements may be warranted.

Pattern search optimizers from the SGOPT package have also been tested on this application. Preliminary results have shown that the same minimum safety margin of 2.5 minutes can be reliably achieved with pattern search.

Furthermore, since pattern search is less sensitive to small-scale nonsmoothness than gradient-based techniques, looser EPSIT tolerances can be employed in the fire simulations which lowers the individual simulation expense considerably. Initial studies have shown that, even though pattern search usually requires more function evaluations than gradient-based optimization, the lower individual simulation expense more than compensates, making the overall computational expense of the pattern search optimization lower than that of the gradient-based optimization (see (Eldred 1996) for a more detailed discussion). Pattern search is, however, susceptible to the "curse of dimensionality," meaning that the method becomes less competitive in efficiency as the number of design variables increases.

Technical Lessons Learned. The application of optimization to this problem led to the following observations:

- When using finite difference gradients in applications with nonsmoothness, an effective finite difference step size is not easily determined. It must be as small as possible to allow for efficient convergence, but still large enough to not be adversely affected by small-scale nonsmoothness.
- Hessian ill-conditioning can be a problem in nonconvex design spaces, causing poor performance in many Newton-based methods. Trust region methods have the potential to overcome this difficulty while maintaining the theoretical strength of second-order optimizers.
- As in the previous application, when computing transient responses for events of unknown duration, increased optimization efficiency and robustness (compute time is conserved, desired response capture is guaranteed) can be achieved with use of an adaptive simulation termination strategy.
- Analysis code convergence tolerances can have a substantial effect on the local nonsmoothness in the design space. When using a gradient-based optimizer, it may be necessary to pay the increased computational expense to employ tight tolerances, so that the optimizer can navigate the design space effectively.
- Gradient-based optimizers put substantial faith in the accuracy of the computed search direction, and in nonsmooth applications, this level of faith may not be justified since the gradients used to calculate the search direction have questionable accuracy. Pattern search optimizers do not confine themselves to a single search direction, but rather search multiple directions simultaneously. As a result, they can be more robust in nonsmooth applications. Moreover, efficiency can be comparable when the number of design variables is small.

Fluid Mechanics: Coating Flow Die Design

Problem Description. The manufacture of polymeric thin-film coatings is an expensive process, requiring high-maintenance, close-tolerance tooling which must be frequently replaced. In premetered processes such as slide, slot, and curtain coating flows (the primary manufacturing methods for film products like video tapes and color photographic film), liquids are distributed uniformly in the transverse direction by chamber-slot dies. Dies must be carefully designed and machined with tight tolerances to ensure uniform flow at the exit (transverse nonuniformity must fall within a few percent). The die-fabrication process is costly in both dollars and time, but is necessary to maximize production throughput of high-quality thin films at the lowest cost. A typical thin-film die is illustrated in Figure 14. Polymer is pumped into a chamber through a feed pipe, and is then extruded through a slot. Typical slot dimensions are 5' wide by .01" high. The output flow profile is typically laminar and the fluid is assumed to be Newtonian. The flow is highly sensitive to a number of design parameters, including geometrical parameters (e.g., slot dimensions, chamber shape and size), fluid properties (e.g., liquid viscosity and density), and process conditions (e.g., volumetric flowrate).

The goal of this study is to find the optimum combination of die geometry parameters which minimizes nonuniformities in the output flow profile for a given set of fluid properties and operating conditions. The objective function of outflow plane nonuniformity is computed by integrating the velocity profile over the outflow plane and is formulated as the percent of coating material across the slot width whose deviation is greater than 1% from the mean velocity. "Perfect" uniformity (0% nonuniformity) is not achievable as the problem is formulated due to the "no-slip" condition at the flow boundaries (i.e. walls of the die). That is, some portion of the coating width will always exceed 1% deviation due to the fact that the coating velocity profile must ramp up from zero at the wall. The die design geometry shown in Figure 14 is parameterized using six design variables that vary the pre-determined sensitive dimensions of the die. These dimensions are slot length L_s, slot height H_s, slot entrance angle α, chamber length L_c, chamber height H_c, and feed-channel height H_f. Slot width W is fixed at 12.7 cm. Constraints on the problem include pump pressure range, an upper bound on the average residence time, and a tolerance band on process temperature. Higher pump pressures generally cause larger flowrate or velocity fluctuations in the feed channel, and higher power (i.e. $) requirements. More importantly, excessive pump pressure can cause deflection of the slot walls, which in turn results in variation of the slot gap height and thus an undesirable increase in flow nonuniformity at the slot exit. In temperature-sensitive coating flow applications (e.g. hot melt extrusion), process temperature must be maintained within a tolerance band to avoid changes to the fluid properties and to the mechanical stability of the die. In low viscosity applications, temperature is less important since operations are normally carried out at room temperature.

There are two stages in a premetered coating flow. The first stage involves pumping coating liquid into a die distributor (or distributors when simultaneous multilayer coating flows are carried out) and distributing the coating liquid uniformly across the die width. In this stage, flow boundaries are of fixed type, i.e. no free boundaries are present. Accordingly, flow simulation is straightforward. The second stage involves transferring the coating liquid from the coating head (i.e. die) to the fast-moving flexible substrate (or web). In this stage, not only are free surfaces (i.e. air/liquid interfaces) present, but the flow also contains static and dynamic contact lines. Here, flow simulation is much more challenging. At present, no commercial code can satisfactorily simulate this latter stage of premetered coating flows. To fill this void, researchers at Sandia National Laboratories are developing specialized computer codes that can efficiently simulate the second stage of a variety of coating flows. For the present study, which involved three-dimensional flow with fixed boundaries, a commercial code based on the finite element method, namely FIDAP (Fluid Dynamics International 1993), was employed.

An input filter program was generated to translate the die geometry parameters into information used by the FIDAP analysis code. This requires preprocessing of parameterized template input decks with APREPRO. Following the FIDAP analysis, an output filter program reads the velocity and pressure profiles from the FIDAP output, integrates the velocity data in calculating the objective function, computes the constraint values based on residence time and maximum pressure calculations, and returns the objective function and constraint values to the optimizer.

Optimization Results and Technical Lessons Learned. As discussed in the earlier applications, nonsmoothness of response surfaces is critical not only for a gradient-based optimization scheme to be efficient, but for it to be viable at all. In this application, two different finite element models were used to investigate the response surface (nonuniformity in the outflow plane versus geometric parameters) smoothness issue. The first model (the “coarse model”) contained 540 elements, whereas, the second model (the “fine model”) had 980 elements. Both models provided navigable surfaces; however, the coarse model had kinks in its surfaces which were nonexistent using the fine model. Although the fine model had more than twice the computational burden, it was necessary to pay the increased computational expense to insure a smoother surface for the optimizer to navigate. The initial geometry and finite element mesh for the fine model are shown in Figure 15.

In the case study presented here, low viscosity coating liquids typically used in precision premetered coating processes are of interest. Specifically, fluid viscosity and density were chosen to be 15 cP and 1000 kg/m^3 respectively, volumetric flowrate per unit slot width was set to be 1.5x10^{-4} m^2/s, and a characteristic length scale was chosen to be 10^{-3} m. With these conditions, the resultant Reynolds number is 10. Constraint allowables were set at 2100 for nondimensional maximum pressure (dimensional gauge pressure = 0.7 psi) and 325 for nondimensional average residence time (dimensional time = 2.2 sec). These values were chosen somewhat arbitrarily based on the nominal design to prevent overly large design changes. More realistic process allowables are needed and will be obtained from industry.

DOT’s modified method of feasible directions optimizer was used to successfully optimize the die geometry and reduce the nonuniformity from 16.5% to 3.2%. The final geometry is shown in Figure 16. Initial and final die dimensions and constraint values are shown in Table 4 (those dimensions which were driven to their upper or lower bounds are indicated with a ‘*’). Comparison between initial and final geometries shows that the optimized geometry is about twice as large as at the initial. As expected, the slot height was reduced to its lower bound, thereby causing the maximum pressure to increase to 1832 (dimensional pressure = 0.6 psi). Since neither of the constraints on maximum pressure or average residence time were active, the optimum design computed is not a direct function of the allowables chosen for these parameters. This low pressure optimal solution is typical of low-viscosity coatings. It is expected that the maximum pressures will be considerably greater for high-viscosity coating applications (e.g., adhesive coatings, hot-melt extrusion).

Although numerical optimization has proved to be valuable in improving die design for low viscosity coating flows, it is evident that this is only part of a general die design effort. Continuation of this work will involve investigation of additional coating materials, primarily high viscosity coatings, which will likely require more careful treatment since the operating constraints will be more important in the design. Also, instead of parameterizing with only a slot entrance angle, a more complex parameterization of the die chamber will be used to describe a tear-drop chamber shape. Lastly, efforts are underway to incorporate analytical sensitivity capability within the FIDAP analysis code so that optimization efficiency can be comparatively evaluated for analytical gradient-based, finite difference gradient-based, and pattern search approaches.

Structural Dynamics: Discrete Optimization of Vibration Isolation Platform

Problem Description. Vibration isolation systems are widely used to protect sensitive devices from vibrations produced in their environment. Typical examples include isolating delicate laboratory experiments from floor-borne vibrations, preventing transmission of vibrations in satellite structures from rotating machinery (e.g., cryo-coolers) to communication antennas or scientific detectors, or isolating a car body or airplane frame from engine vibrations.

Isolation is achieved with the use of passive or active compliant connections between the source of the vibrations and the device to be protected. The classical approach to designing isolation systems focuses primarily on the properties of those compliant mounts without much regard to the geometry of the complete system, that is, the locations and/or orientations of the mounts on the isolated device. In cases where the source of the vibrations is well known (location, amplitude, frequency content) and/or when the residual motion at *specific* points on the isolated device are of critical importance, we should expect (see, for example, (Ashrafiuon 1993)) that mount locations and orientations will have a substantial effect on the effectiveness of the isolation.

To investigate this, we define a simplified vibration isolation problem, based on an existing laboratory setup (Figure 17). The setup consists of a rigid, rectangular optical table (approximately 48” x 36” x 8.5”, 815 lb.) mounted on 3 vibration isolation mounts. This system is in turn resting on a massive seismic base (approximately 70” x 48” x 12”, 4085 lb.), itself isolated from floor borne vibrations by 4 air bags. The isolator mounts supporting the optical table are steel coil springs with stiffness of about 3970 lb/in in the axial direction and 1570 lb/in in the transverse (shear) directions. The bottom of the optical table and the top of the seismic base feature identical 8x6 arrays of mounting holes for those springs. The 6 rigid body natural frequencies of the seismic base on its air bags range from about 1.1 Hz to 2.5 Hz. Both the base and the optical table can be regarded as rigid bodies below a few hundred Hertz, where flexible modes appear.

A 3-dimensional, rigid body dynamics code was developed in MATLAB (The Mathworks 1992) to model this system. The air bags and steel springs are modeled as 3-dimensional springs with viscous damping. Their bending and torsional stiffnesses are neglected. The seismic base and optical table are modeled as concentrated masses and inertias. The mass properties of all parts of the system were either measured or evaluated from CAD models. The stiffness and damping of the air bags and steel springs were obtained from the manufacturers and further refined through parameter estimation based on measured natural frequencies and modal damping.

To simulate a perturbation, the seismic base is excited at its front right corner by an electromagnetic shaker (Fig. 17) with a sinusoidal input at a fixed frequency of 50 Hz. We assume that the vertical component of velocity at the front left corner of the optical table (Fig. 17) is the critical design criterion (because a sensitive instrument is mounted there for example). These locations introduce asymmetry in the problem and lead to the nonintuitive optimal solutions for the isolator locations.

The design problem consists of selecting locations for the 3 spring isolators on the 6x8 grid of mounting holes in a way that minimizes transmission of the perturbation to the specified point on the optical table. The 6 discrete design variables are the x and y locations of the 3 springs. The transmission T (μin/sec.-lb.) is expressed as the vertical velocity at the given point of the optical table per unit load amplitude at the excitation point. We define a baseline design where the 3 springs are arranged as symmetrically as possible around the center of the table (Fig 18) which has a predicted transmission of 21.22 μin/sec.-lb. at 50 Hz.

Since the optimized designs will be tested in the laboratory, a number of practical constraints apply:

- Due to the presence of lifters, the 4 holes at the corners of the grid cannot be used.
- The 3 springs cannot be colinear and only one spring is allowed per grid location.
- To avoid generating designs that would be too unstable, the first natural frequency of the optical table on its mounts is constrained to be more than 4 Hz. This also ensures decoupling between the modes of the seismic base/airbags system and those of the optical table/springs system.
- Because the springs are simply resting in end plates attached to the mounting holes, the static load on each spring must be compressive.
- There is a limit to the amount of weight a spring can support before being compressed to its solid length. An upper limit is imposed on the static deflection of each spring.

Optimization Results. Because of the discreteness of the problem, the optimization was performed with the genetic algorithm (GA) available in SGOPT within DAKOTA. The MATLAB analysis code was coupled to the optimizer through simple UNIX scripts. No input or output filters were used; instead, the MATLAB code was designed to exchange input and output directly in the DAKOTA-compatible format. The design variables (locations of 3 springs) are coded into a “chromosome” composed of 6 integer-valued “genes” that define the grid indices for x and y coordinates of the first, second and third spring, i.e. [x1, y1, x2, y2, x3, y3]. Any x gene can take integer values between 1 and 6 while y genes take values between 1 and 8. The size of the design space (total number of conceivable designs) is approximately 100,000.

Since the GA in DAKOTA cannot explicitly handle constraints, penalty functions are implemented within the MATLAB analysis code. Both step and quadratic penalty functions were used for this problem. In the step penalty, any amount of constraint violation produces a fixed amount of penalty, subtracted from the fitness (Leriche 1994). This is done to ensure convergence to a feasible design, which may not occur when using only progressive penalty functions. Because GA’s are zero-order search techniques, the discontinuities this strategy creates are perfectly acceptable. In addition to the step penalties, the “quantifiable” constraint violations (4Hz frequency limit, spring static loads and deflections) produce penalties proportional to the square of the violation (constraints are first normalized).

Out of several options and adjustments available in DAKOTA, the following combination was chosen:

- population size: 10; crossover probability: 0.8; mutation probability: 0.1; number of generations: 15.
- ranking technique for selection: the probability of selection of an individual is proportional to its rank.
- moderate selection pressure: the probability of selection of the best individual is twice that of the worst.
- uniform mutation: a mutated gene takes a random integer value, uniformly distributed from 1 to 6 for x genes and 1 and 8 for y genes.
- elitist strategy: the best individual of the current generation is duplicated into the next generation.

With these adjustments, each search evaluates less than 150 designs, or 1.5% of the design space (the GA keeps track of previously analysed designs, so the actual number of function evaluations averages around 105). In order to get some idea of the reliability of the search, we performed series of 10 runs and compare the results to a series of 10 random searches of 105 analyses each. The result from each random search is the best feasible design found. Typical results are shown in Fig. 19. The random searches generate some good designs and many mediocre ones. In contrast, all 10 designs obtained from the GA runs are feasible and represent significant improvements from the baseline case. However, the GA occasionally “converges” to a relatively poor design (T=3.23 in Fig. 19). This shows that more reliable results can be obtained by running a small number of short searches (if the probability that the best found design is “poor” is 0.1 for a one run, then it is only 0.01 for the best of 2 runs, 0.001 for the best of 3, etc.). Because GA’s are most efficient in the initial phases of the search and further “convergence” is usually slow, this approach is preferable to running a single longer search (Leriche 1994).

The “optimal” designs are also very different from each other, as illustrated in Fig. 18 where a few configurations are shown with their predicted transmissibility. This is a strength of genetic algorithms: final designs are random “quasi-optimal” configurations that tend to have similar values of the objective function but distinct design features; the final choice rests in the hands of the designer. However, in this case, the various designs of Fig. 18 resulted from distinct runs of the GA and were not present together in final populations. In fact, the final populations usually contained only one or two “good” designs, with most others infeasible. This is the result of 1.) significant multimodality, as evidenced by the large differences between distinct “optimal” designs (Fig. 18), 2.) a highly constrained design space (12 constraints), and 3.) a relatively small design space (100,000 designs).

This creates small “pockets” of feasible designs isolated from each other in the design space. Because the design space is small (coarse 6x8 grid) and feasible designs are rare (Monte Carlo simulations show that only about 13% of random designs are feasible), each “pocket” tends to contain only a few designs and the probability that genetic operators will generate feasible designs is small (even by mating two feasible designs). Also, since designs in distinct pockets are very different, there is no reason to expect that combining features of those different designs would create better or even feasible designs (the concept of niches (Goldberg, 1989) was developed to handle this problem but we think that in this case, the number of feasible designs in each niche is too small to allow exploitation by GA operators). This means that most of the search has to take place in the infeasible design space and explains the presence of few feasible designs in each population.

Continuous optimization was also tested on this problem. DOT’s modified method of feasible directions was used within the DAKOTA framework to solve the constrained continuous problem (ignoring the grid), followed by rounding of the optimal design to neighboring discrete solutions. An optimal solution was found at (2.22, 1.56, 2.49, 4.94, 4.29, 3.79) with a transmission T=0.20 μin/sec.-lb. Rounding to the closest neighbor gives (2,2,2,5,4,4), which is infeasible. If we consider all immediate neighbors of the continuous solution (Fig. 20), we find that out of the 64 designs, only 12 are feasible and the best of these has T=3.67, 22 times worse than design A of Figure 18.

Figure 21 shows frequency response functions (FRF’s) for the designs of Fig. 18. These FRF’s show that the GA is seeking out an anti-resonance condition in the vicinity of 50 Hz. Anti-resonance is achieved by combining mode shapes so as to cancel out vibration at a point. The fact that the anti-resonant peaks miss the 50Hz target slightly is due to the discrete isolator locations; and in fact, the continuous solution of T=0.20 μin/sec.-lb. places the anti-resonant peak directly at 50Hz (not shown). Another important fact to observe is that there is significant broad band improvement in the transmissibilities of the optimized designs compared to the baseline design. That is, the improvements are not confined only to a 50 Hz input; but rather there are significant decreases in transmission through a broad input range. This is an especially important observation, since it shows that the performance is not seriously degraded for off-nominal excitation inputs and that more sophisticated objective function formulations for minimizing broad band transmission are probably unnecessary.

Finally, the designs of Fig. 18 were tested in the laboratory. The results of those tests are shown in Fig. 22 and compared to analytical predictions. The figure shows the nominal value of the transmissibility (objective function in the optimizations) and the predicted range of transmissibilities with 5% uncertainties in the spring constants. This analytical range is determined with Monte Carlo simulations in which each spring constant is chosen randomly within the 5% uncertainty range. Note that, in most cases, the experimental value falls within the analytical range. The transmissibilities of the optimized designs were predicted between 32 and 70 times smaller than that of the baseline design. The actual ratios achieved in the lab range from 7 to 46. They all represent significant improvements from the original design.

Technical Lessons Learned. The application of genetic algorithms to this discrete, multi-modal, heavily constrained problem led to the following observations:

- Applying constraints through penalty functions in a GA problem is a delicate operation. A balance must be achieved between the desire to obtain a feasible final design and the need to allow the search to cross infeasible regions of the design space. Surprisingly little research has been devoted to this aspect. One reason is that, in research GA’s, problem-specific repair operators are often introduced to enforce constraints. This approach is more efficient but is highly application-specific and cannot be included in general purpose codes like DAKOTA.
- The combination of multi-modality, large number of constraints, and limited design options (coarse discrete grid in this case) makes the problem difficult to handle, even for a zero-order random search technique like the GA.
- The classical argument that a GA provides multiple design alternatives in its final population does not hold in heavily constrained discrete problems with small design spaces. Instead, each run provides only one or two acceptable designs.
- Multiple design options and improved reliability of the search can be obtained by running a few short searches, rather than a single, long search.
- Continuous optimization followed by rounding to neighboring discrete solutions does not generally lead to an optimal design. For problems with coarse discrete grids, heavily constrained design space, and rapidly varying objective function, this approach leads to few, relatively poor feasible designs.

Conclusions

Object-oriented software design has been shown to be an effective tool for the generic integration of optimization techniques with broad classes of simulation codes. Three applications involved optimizing expensive, nonsmooth engineering simulations with nonlinear programming, and one application demonstrated the solution of a discrete design problem with constraints via genetic algorithm. In several of these applications, the DAKOTA system was used for easy comparison of different optimization algorithms which enabled illuminating assessments of relative performance.

In the engineering simulation applications, nonsmoothness in the design space has been shown to be a recurring problem when applying gradient-based optimization techniques to “black box” transient simulation codes. This ultimately must be addressed with attention to nonsmoothness reductions in the analyses and with employment of robust algorithms in the optimization. The high computational expense of repeated analyses is the other crucial, recurring difficulty. This must be addressed with attention to modeling simplifications, analysis code convergence tolerances, efficient time stop strategies for transient simulations, function approximation strategies, and robust and efficient optimization algorithms which make the most of each function evaluation and which are successful the first time (thereby avoiding multiple executions and tweaking of parameters before achieving convergence).

In the vibration isolation application, the challenges were quite different. Instead of nonsmoothness and extreme computational expense, discrete design variables and a highly constrained design space were the primary challenges. Genetic algorithms have shown promise as effective techniques for these types of problems. However, while GA's can be very robust techniques, the number of function evaluations they require can be prohibitive for applications involving expensive engineering simulations.

Thus, research in this paper has focused on how generic interfacing of iteration with simulation is performed, what application-specific techniques are useful in enabling reliable and efficient optimization studies, and what algorithmic strengths and weaknesses can be observed when interfacing existing optimization techniques with complex engineering simulations. It has been shown that existing techniques have limitations in their effectiveness when dealing with the combined challenges of high computational expense and nonsmooth, nonconvex design spaces. These observations have uncovered research needs which are directing advanced strategy development efforts in fundamental algorithms, algorithm hybridization, function approximation, parallel processing, and automatic differentiation. Progress in these areas is presented in a separate paper. The overall goal of this collection of research activities is to develop a broadly useful optimization capability with the flexibility and extensibility to easily accommodate broad classes of optimizers, a wide disciplinary range of simulation capabilities, and advanced strategies which seek to enhance robustness and efficiency beyond that which is currently available.

Acknowledgments

The authors gratefully acknowledge the efforts of Dave Harding, Mike Neilsen, Terry Hinnerichs, and Roy Hogan in providing the groundwork for the transportation cask optimization study; Bill Hart in performing preliminary investigations of pattern search optimizers on the worst-case fire surety application; and Clay Fulcher in providing support and technical direction in the vibration isolation application.

References

Ashrafiuon, H., "Design Optimization of Aircraft Engine-Mount Systems," *Journal of Vibration and Acoustics,* ASME, Vol. 115, October 1993, pp. 463-467.

Blacker, T.D., *FASTQ Users Manual Version 1.2*, Sandia Report SAND88-1326, June 1988.

Coleman, T.F. and Li, Y., Eds., *Large-Scale Numerical Optimization*, Society for Industrial and Applied Mathematics, Philadelphia, 1990.

Eldred, M.S., Hart, W.E., Bohnhoff, W.J., Romero, V.J., Hutchinson, S.A., and Salinger, A.G., "Utilizing Object-Oriented Design to Build Advanced Optimization Strategies with Generic Implementation," submitted for the *6th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, WA, Sept. 4-6, 1996.

Fluid Dynamics International, Inc., *FIDAP Users Manual*, version 7.0, Evanston, IL, April 1993.

Gartling, D.K., and Hogan, R.E., *COYOTE II - A Finite Element Computer Program for Nonlinear Heat Conduction Problems, Part II - User's Manual*, Sandia Report SAND94-1179, October 1994.

Gilkey, A.P., and Glick, J.H., *BLOT - A Mesh and Curve Plot Program for the Output of a Finite Element Analysis*, Sandia Report SAND88-1432, June 1989.

Gill, P.E., Murray, W., Saunders, M.A., and Wright, M.H., *User's Guide for NPSOL (Version 4.0): A Fortran Package for Nonlinear Programming*, System Optimization Laboratory, TR SOL-86-2, Stanford University, Stanford, CA, Jan. 1986.

Goldberg, D.E., *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley Publishing Co., Reading, MA, 1989.

Haftka, R.T., Gurdal, Z., and Kamat, M.P., *Elements of Structural Optimization*, Second revised edition, Kluwer Academic Publishers, Boston, 1990.

Harding, D.C., Eldred, M.S., and Witkowski, W.R., "Integration of Finite Element Analysis and Numerical Optimization Techniques for RAM Transport Package Design," proceedings of the *11th International Conference on the Packaging and Transportation of Radioactive Materials* (PATRAM '95), Las Vegas, NV, Dec. 3-8, 1995.

Hart, W.E., *Adaptive Global Optimization with Local Search*, Ph.D. dissertation, University of California at San Diego, May 1994.

Hart, W.E., *Evolutionary Pattern Search Algorithms*, Sandia Report SAND95-2293, Sept. 1995.

Kamat, M.P., Ed., *Structural Optimization: Status and Promise*, Progress in Astronautics and Aeronautics, Vol. 150, American Institute of Aeronautics and Astronautics, Washington DC, 1993.

Leriche, R., *Composite Structures Optimization by Genetic Algorithms*, Ph.D. dissertation, Virginia Tech, Blacksburg, VA, October 1994.

Meza, J.C., *OPT++: An Object-Oriented Class Library for Nonlinear Optimization*, Sandia Report SAND94-8225, Sandia National Laboratories, Livermore, CA, March 1994.

PDA Engineering, *PATRAN Plus version 2.5 User Manual*, Costa Mesa, CA, July 1988.

PDA Engineering, *P/THERMAL User Manual*, Release 2.6, Costa Mesa, CA, March 1993.

Romero, V.J., Eldred, M.S., Bohnhoff, W.J., and Outka, D.E., "Application of Optimization to the Inverse Problem of Finding the Worst-Case Heating Configuration in a Fire," proceedings of the *9th International Conference on Numerical Methods in Thermal Problems*, Atlanta, GA, July 17-21, 1995.

Sjaardema, G.D., *APREPRO: An Algebraic Preprocessor for Parameterizing Finite Element Analyses*, Sandia Report SAND92-2291, Dec. 1992.

Stroustrup, B., *The C++ Programming Language*, 2nd ed., Addison-Wesley, New York, 1991.

Taylor, L.M., and Flanagan, D.P., *PRONTO2D: A Two-Dimensional Solid Mechanics Program*, Sandia Report SAND86-0594, 1986.

The MathWorks, Inc., *MATLAB User's Guide*, Natick, MA, August 1992.

Törn, A., and Zilinskas, A., *Global Optimization*, Lecture Notes in Computer Science, Springer-Verlag, 1989.

Vanderplaats Research and Development, Inc., *DOT Users Manual*, Version 4.20, Colorado Springs, CO, 1995.

Witkowski, W.R., Eldred, M.S., and Harding, D.C., "Integration of Numerical Analysis Tools for Automated Numerical Optimization of a Transportation Package Design," *5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization* (MDO), paper AIAA94-4259, Panama City Beach, FL, Sept. 7-9, 1994.

Table 1: Weight minimizations subject to end impact, side impact, and thermal constraints.

	Initial Design				Final Design			
	Weight (lbs.)	End σ_{yy} (psi)	Side σ_{yy} (psi)	Max. Temp. (^{o}C)	Weight (lbs.)	End σ_{yy} (psi)	Side σ_{yy} (psi)	Max. Temp. (^{o}C)
End-on	181.6	19012.	N/A	N/A	40.40	$\underline{22976.}$	N/A	N/A
Thermal	153.9	N/A	N/A	153.4	62.19	N/A	N/A	$\underline{232.6}$
Combined	181.6	19012.	4005.	162.6	93.98	$\underline{23012.}$	6614.	$\underline{231.7}$

Table 2: Design variable values for computed optima.

	x1	x2	x3	x4	x5	x6	Optimal Shape
End-on	.466	.710	$\underline{.104}$	1.13	10.5	1.13	Long & Thin
Thermal	1.68	$\underline{.100}$	1.04	1.15	.805	.921	Short & Fat
Combined	2.22	.909	$\underline{.100}$	.647	10.3	$\underline{.107}$	Compromise

Table 3: Optimization results with CG optimizers for EPSIT = 10^{-2}, EPSIT2 = 10^{-4}

Optimizer	FDSS	Initial Values (r, x)	Initial Obj. Fn.	Funct. Evals.	Final Values (r, x)	Final Obj. Fn.
OPT++ P.-R. CG	4%	(1.4, 0.5)	7.2045	34*	(1.5812, 0.75016)	2.8293
OPT++ P.-R. CG	1%	(1.4, 0.5)	7.2045	100*	(1.6038, 0.76547)	2.5956
OPT++ P.-R. CG	0.1%	(1.4, 0.5)	7.2045	73	(1.6086, 0.76895)	2.5546
OPT++ P.-R. CG	0.01%	(1.4, 0.5)	7.2045	28*	(1.6016, 0.57370)	4.9477
OPT++ P.-R. CG	1%	(1.0, 1.0)	86.954	31*	(1.9321, 0.62026)	5.9543
OPT++ P.-R. CG	1%	(1.2, 0.9)	34.831	106	(2.1044, 0.50665)	6.9526
DOT F.-R. CG	1%	(1.4, 0.5)	7.2045	34	(1.6435, 0.77498)	2.5973
DOT F.-R. CG	1%	(1.0, 1.0)	86.954	40	(1.6374, 0.77591)	2.5362
DOT F.-R. CG	1%	(1.2, 0.9)	34.831	38	(1.6475, 0.77393)	2.6217

Table 4: Preliminary Optimization Results

	Initial	Final
Slot Length, L_s (cm)	5.0	11.3
Slot Height, H_s (cm)	1.0	0.25*
Slot Entrance Angle, α (degrees)	53.0	60.0*
Chamber Length, L_c (cm)	10.9	20.0*
Chamber Height, H_c (cm)	4.3	9.9
Feed-Channel Height, H_f (cm)	15.0	14.3
Non-uniformity (%)	16.5	3.2
Average Residence Time†	253.6	215.2
Maximum Pressure†	42.3	1832.

* indicates active bound constraint
† indicates nondimensional quantity

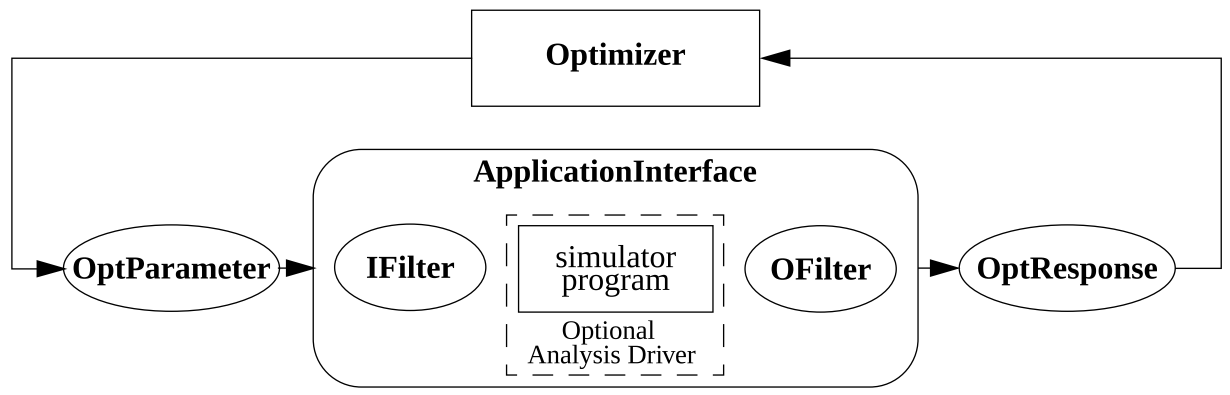


Figure 1. Application interface conceptualization.

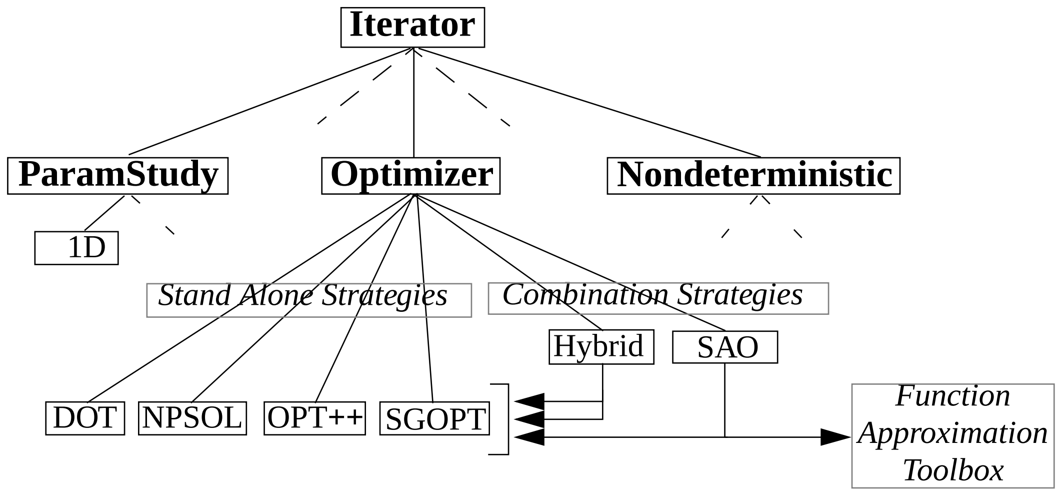


Figure 2. Iterator hierarchy showing broad iteration capabilities:
A) **ParamStudy**: for mapping response variations with respect to model parameters.
B) **Optimizer**: for numerical optimization studies.
C) **Nondeterministic**: for assessing the effect of modeling uncertainties on responses.

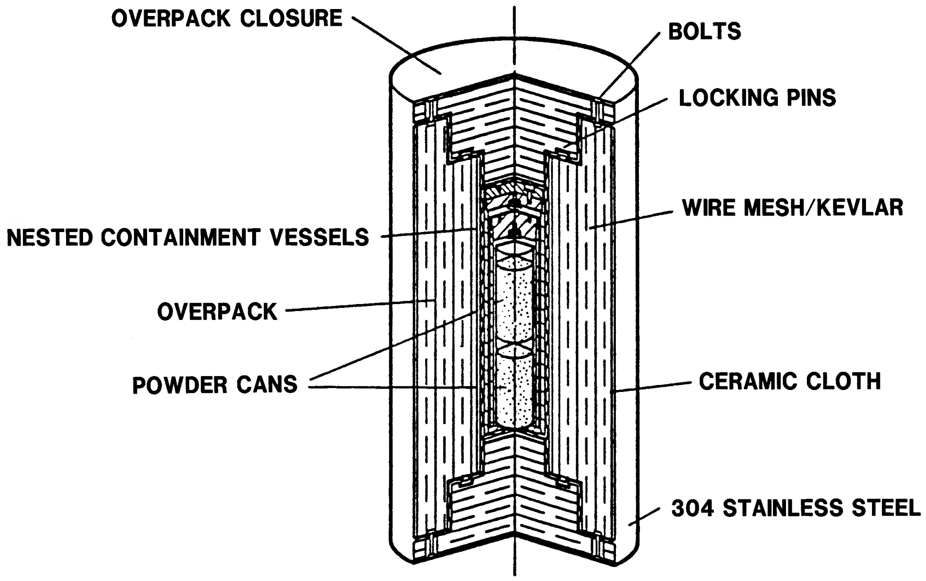


Figure 3. Typical transportation cask.

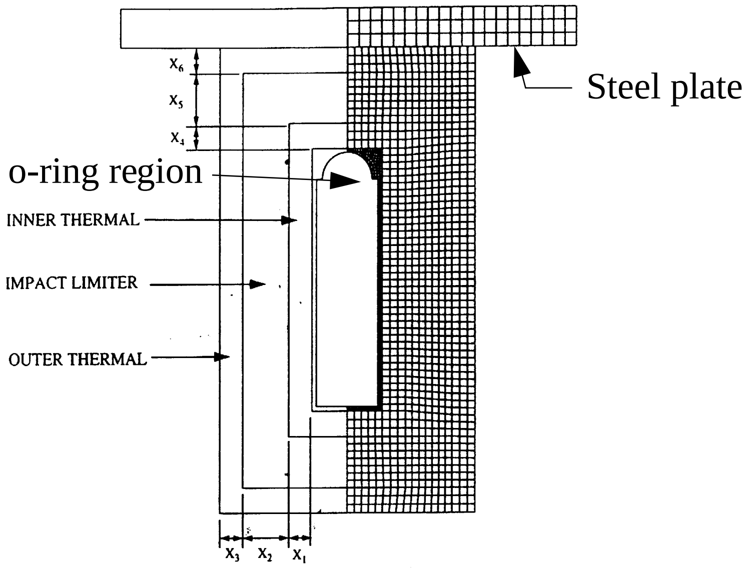


Figure 4. Axisymmetric finite element model for end-on impact showing the 6 design variables which define overpack layer thicknesses.

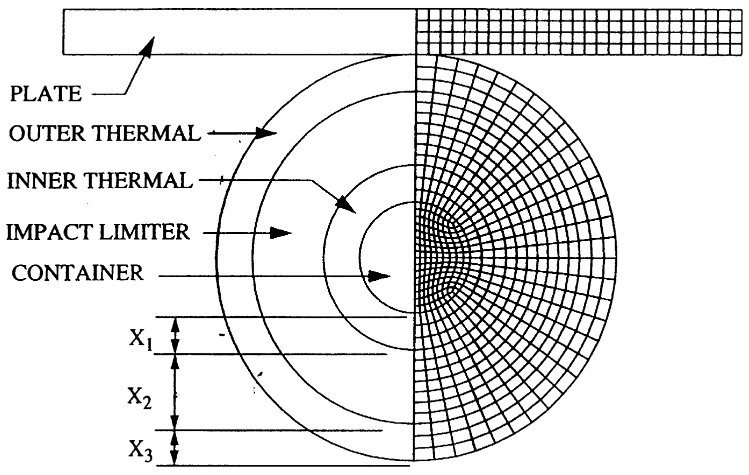


Figure 5. Finite element model for side-on impact showing radial thickness layers.

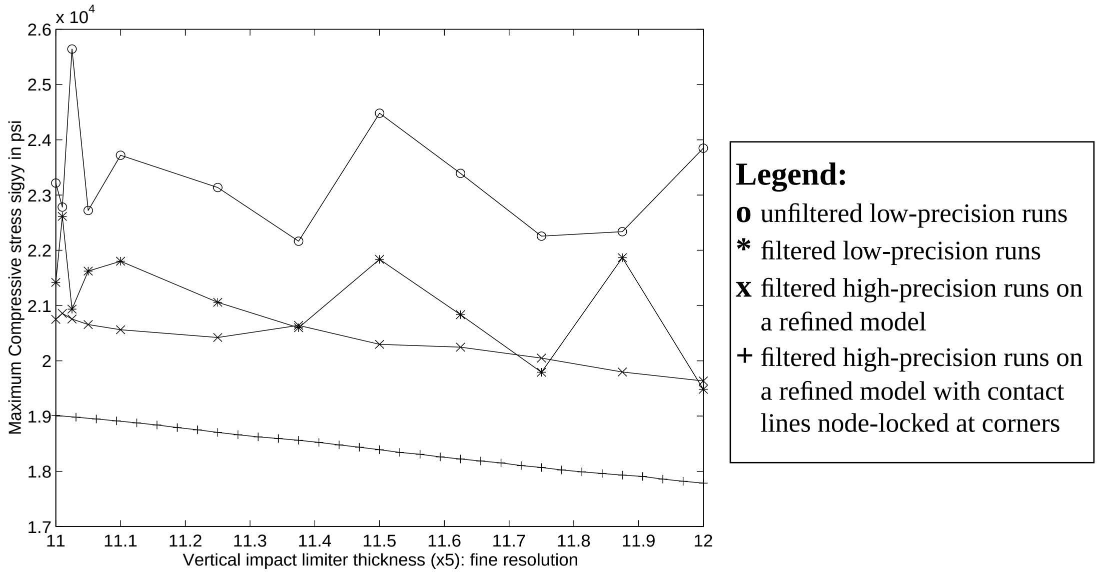


Figure 6. Smoothing of stress constraint through modeling improvements.

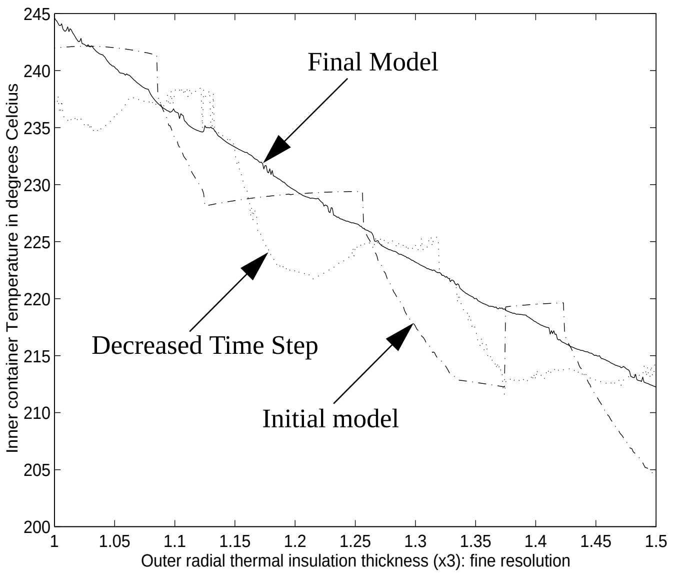


Figure 7. Smoothing of temperature constraint through modeling improvements.

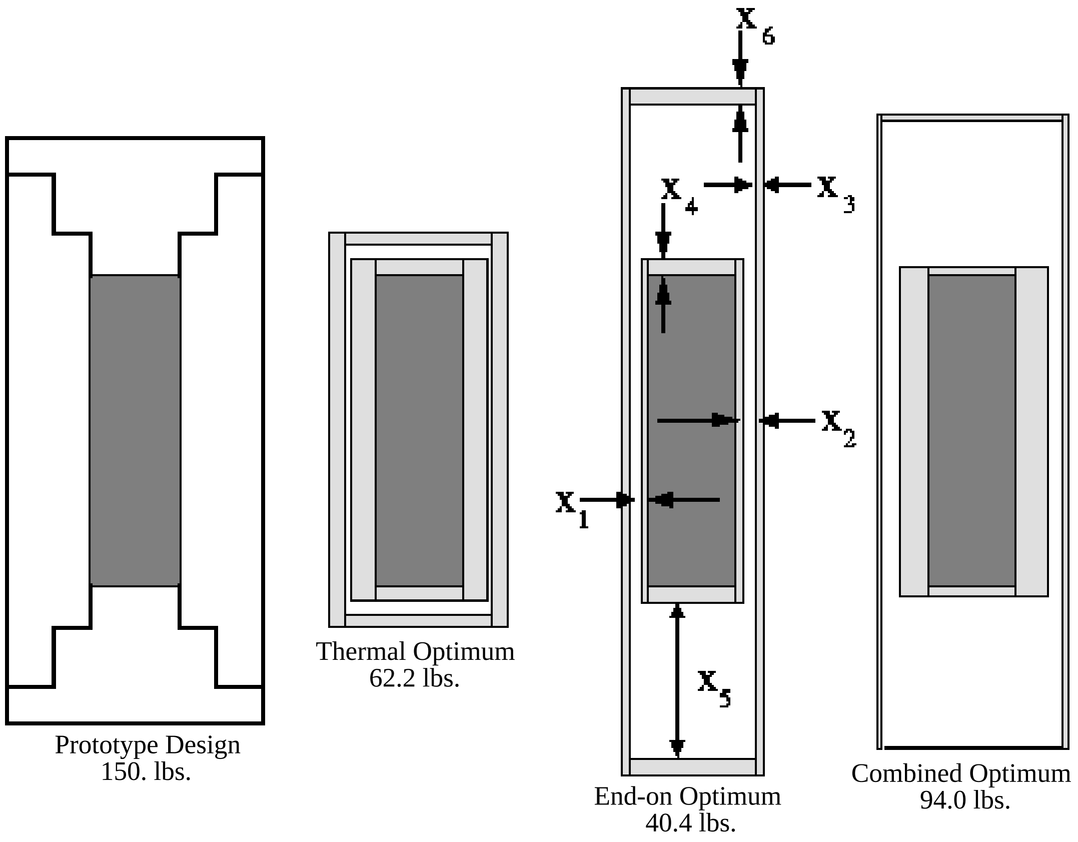


Figure 8. Comparison of computed optimal shapes with experimental prototype design.

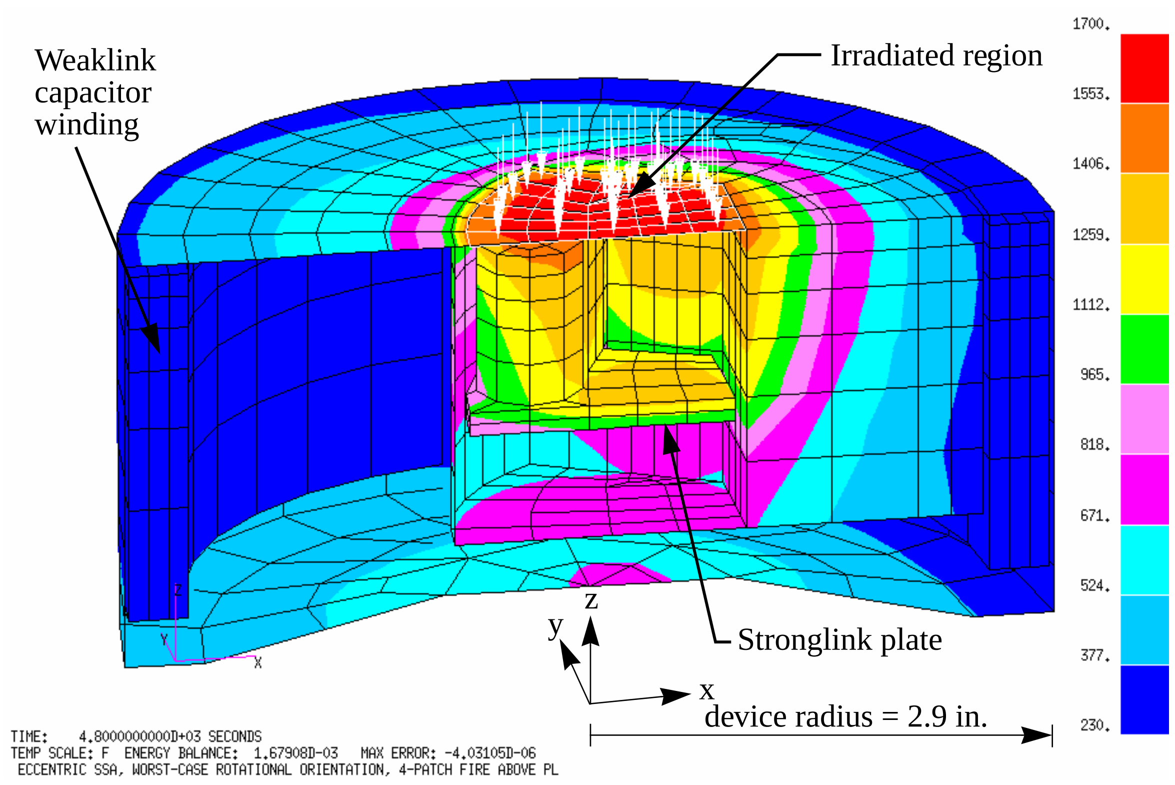

Weaklink capacitor winding
Irradiated region
Stronglink plate
device radius = 2.9 in.
z
y
x
1700.
1553.
1406.
1259.
1112.
965.
818.
671.
524.
377.
230.
TIME: 4.8000000000D+03 SECONDS
TEMP SCALE: F ENERGY BALANCE: 1.67908D-03 MAX ERROR: -4.03105D-06
ECCENTRIC SSA, WORST-CASE ROTATIONAL ORIENTATION, 4-PATCH FIRE ABOVE PL

Figure 9. Finite element model and typical temperature distribution (^{o}F).

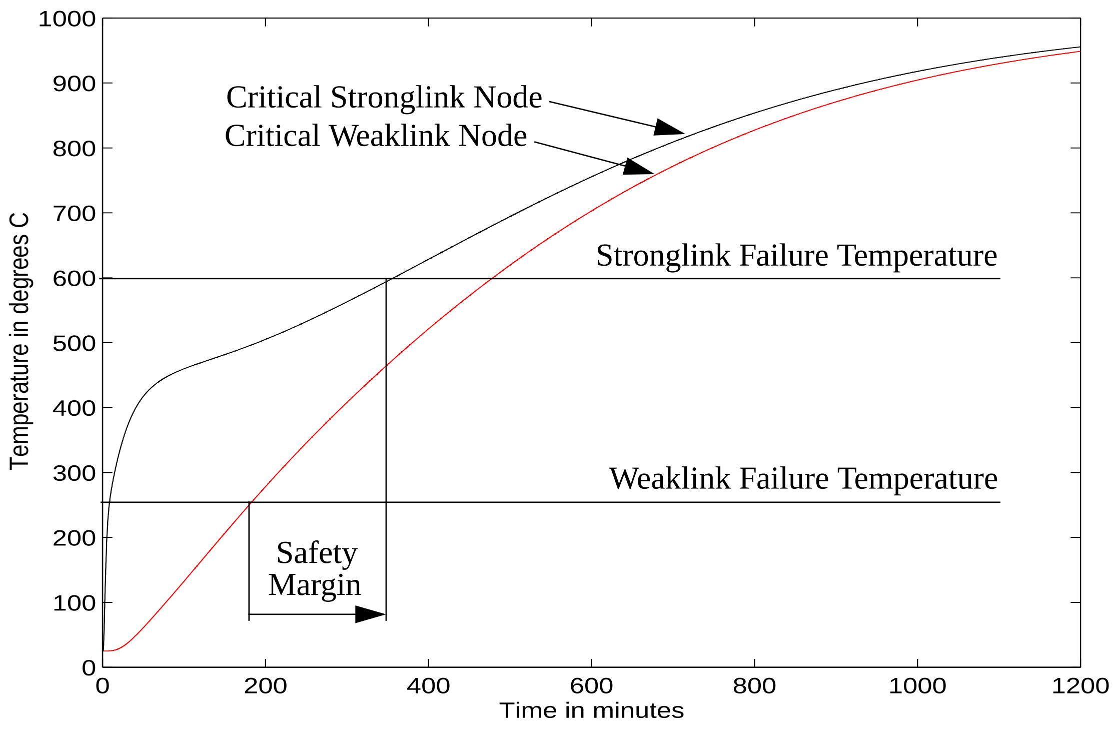


Figure 10. Temperature histories of critical stronglink and weaklink nodes.

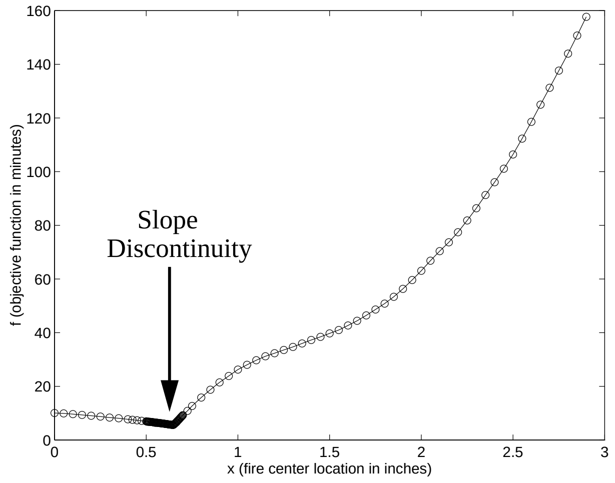


Figure 11. Objective function variation with respect to fire center location (x) for r=1.89.

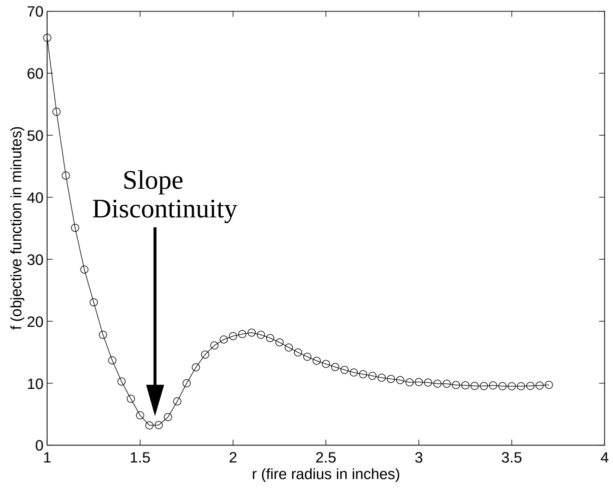


Figure 12. Objective function variation with respect to fire radius (r) for x=0.8.

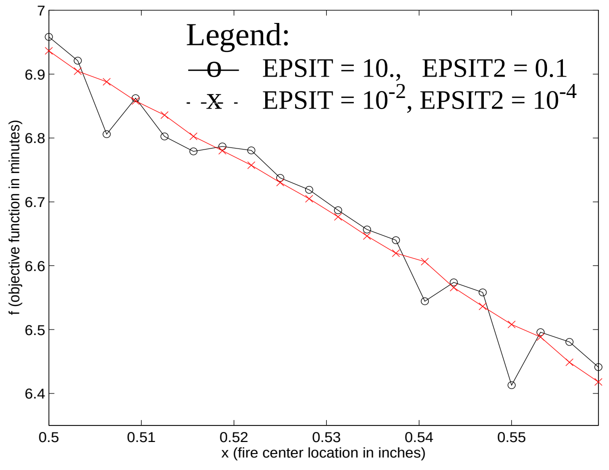


Figure 13. Detail of Figure 11 showing effect of QTRAN convergence tolerances on design space nonsmoothness.

.

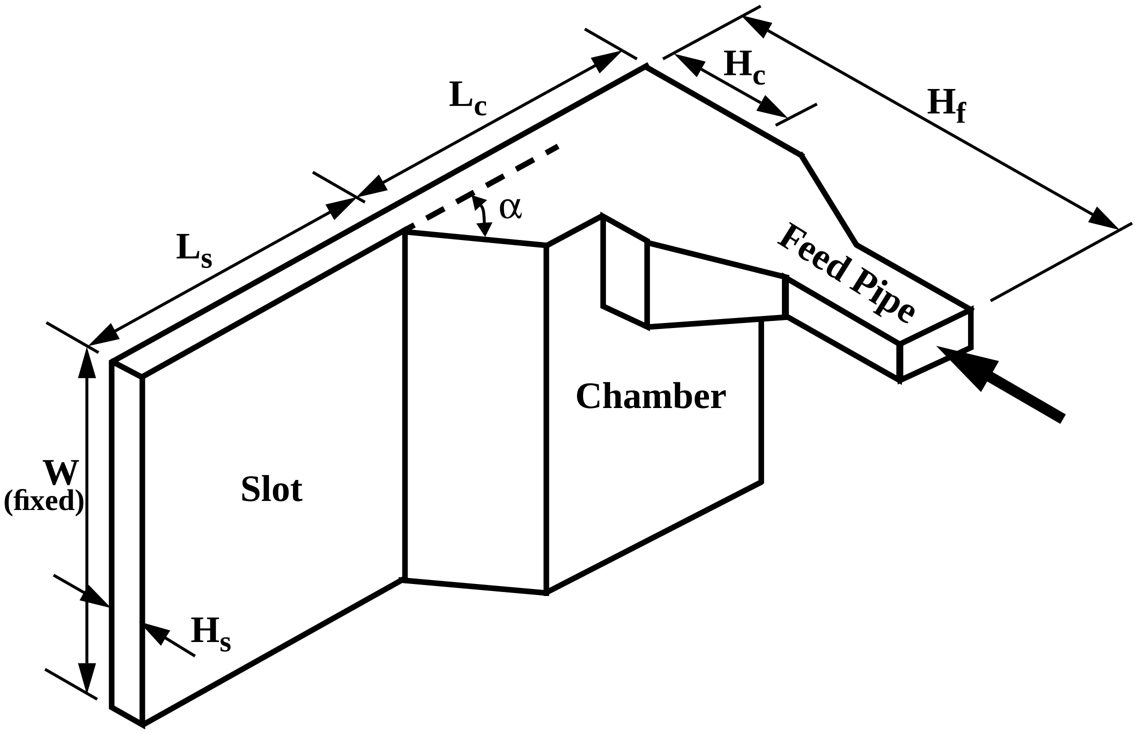


Figure 14. Schematic of a chamber-slot die.

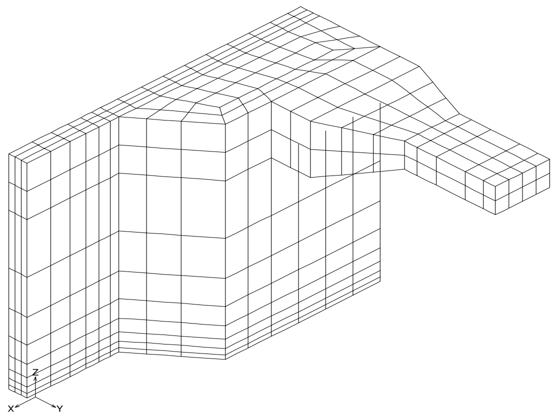


Figure 15. Initial geometry.

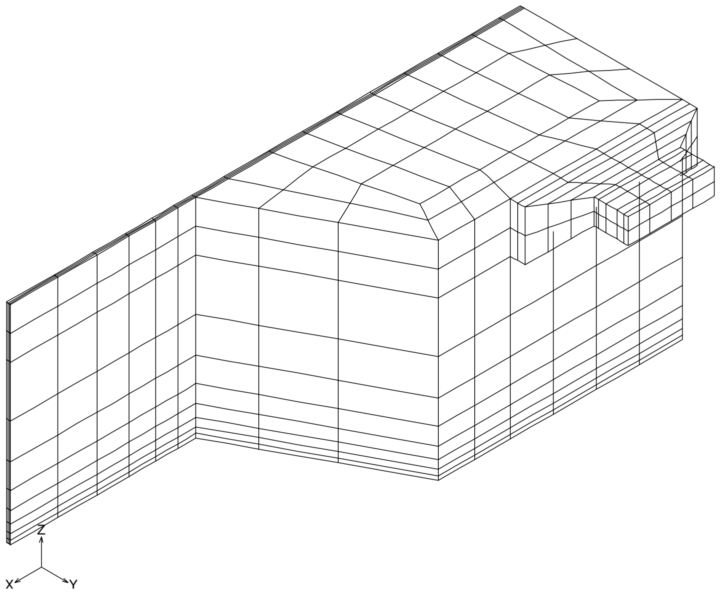


Figure 16. Final optimized geometry.

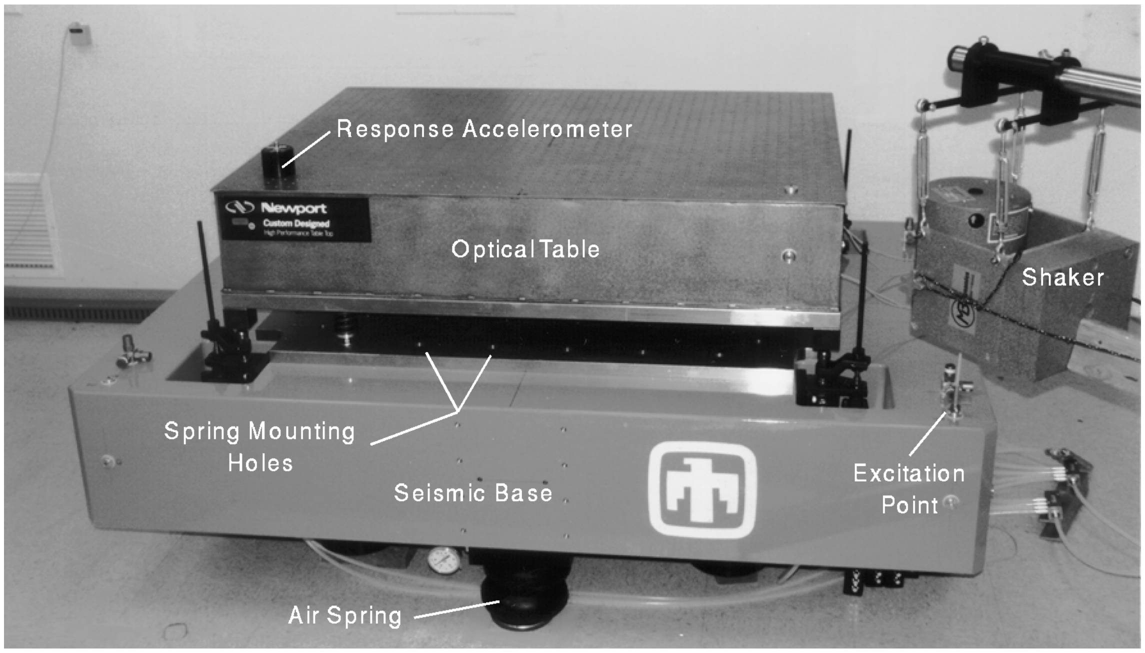


Figure 17. Vibration isolation testbed hardware.

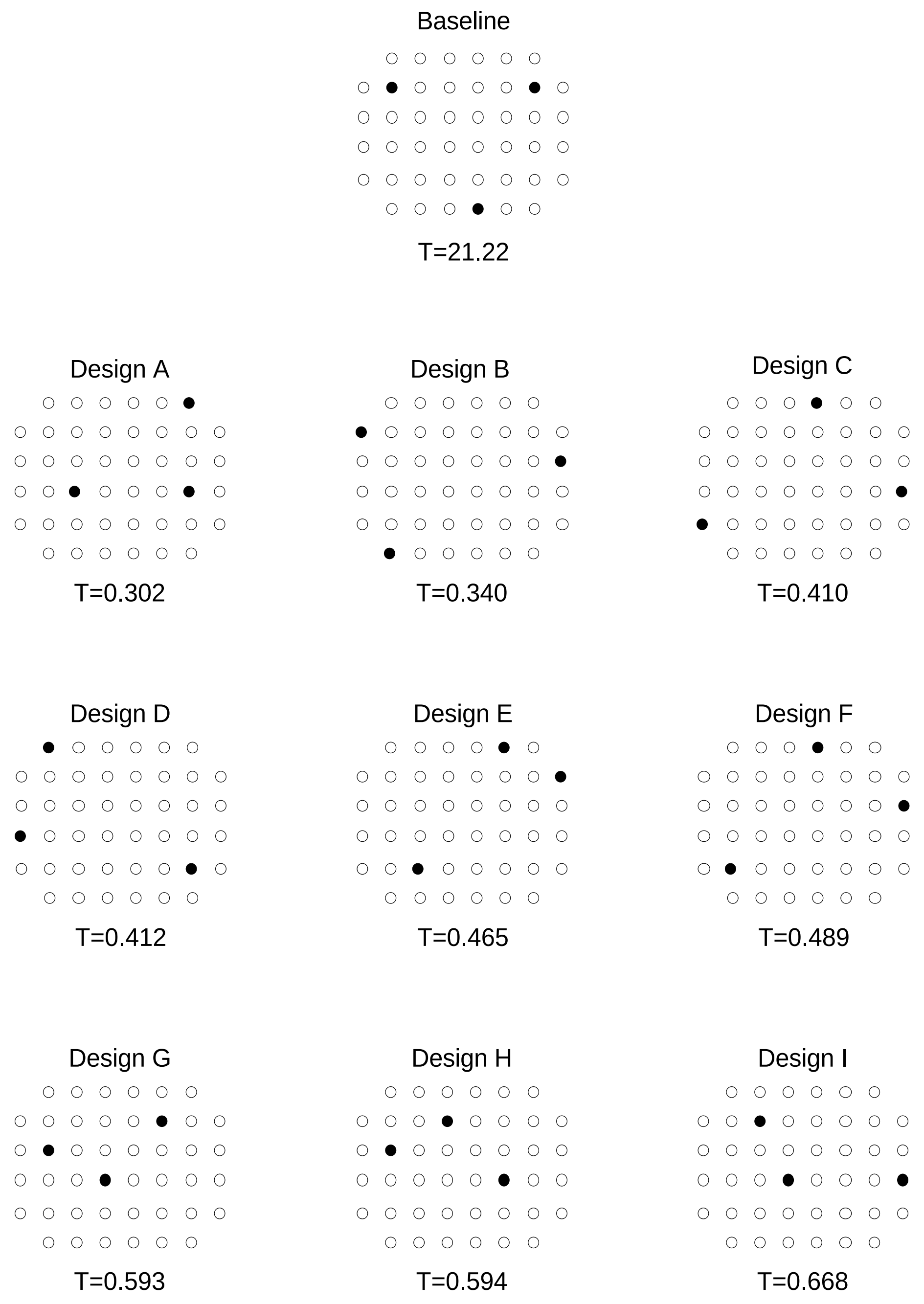


Figure 18. Spring locations and transmissibilities of baseline and 9 optimized designs.

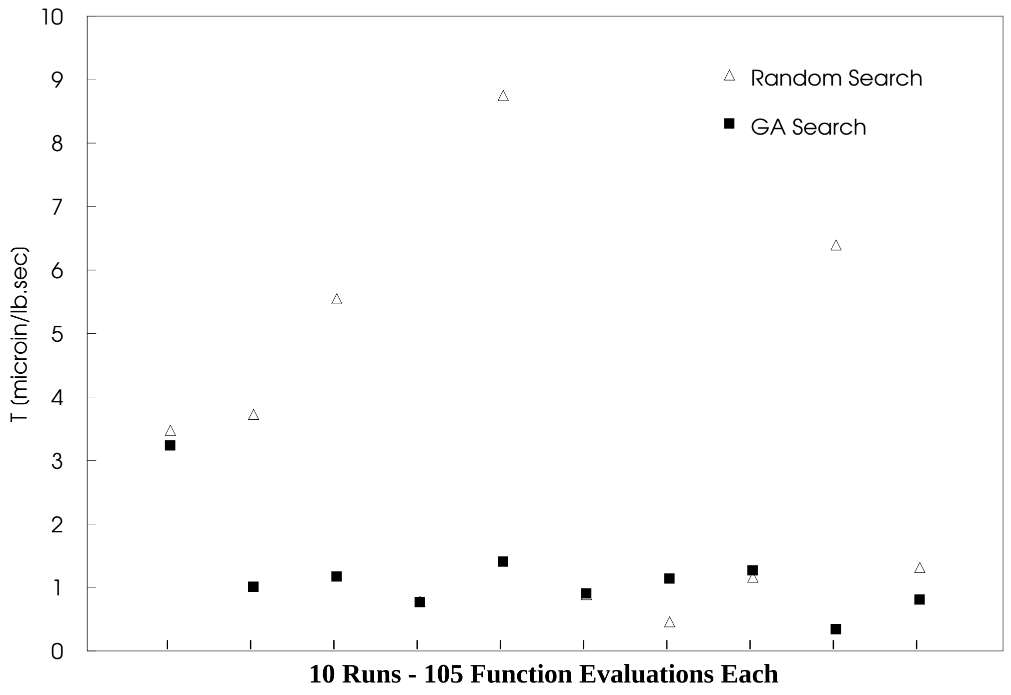
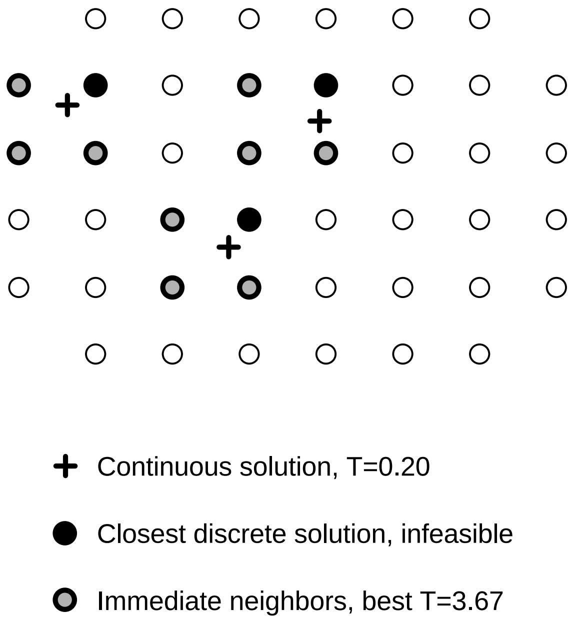


Figure 19. Results from 10 GA runs compared to 10 random searches for the same number of analyses.



Continuous solution, T=0.20
Closest discrete solution, infeasible
Immediate neighbors, best T=3.67

Figure 20. Comparison of continuous optimum and nearby discrete solutions.

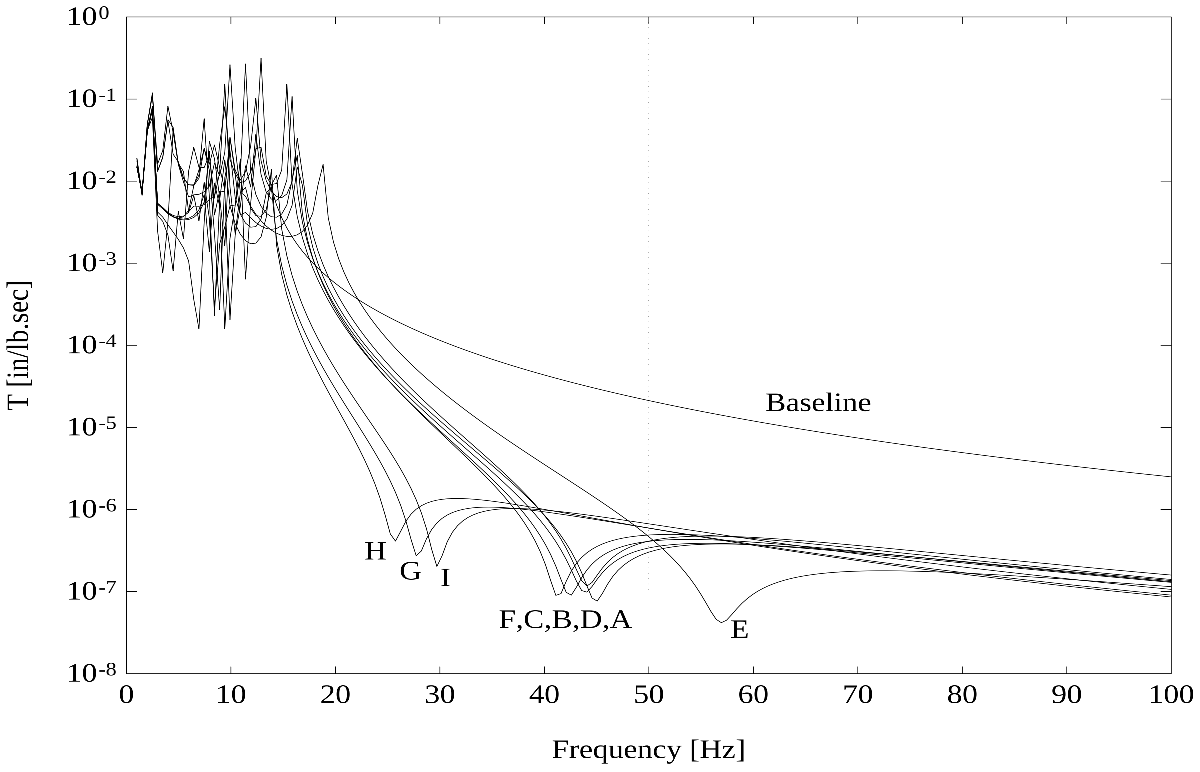


Figure 21. Frequency response functions for optimized designs.

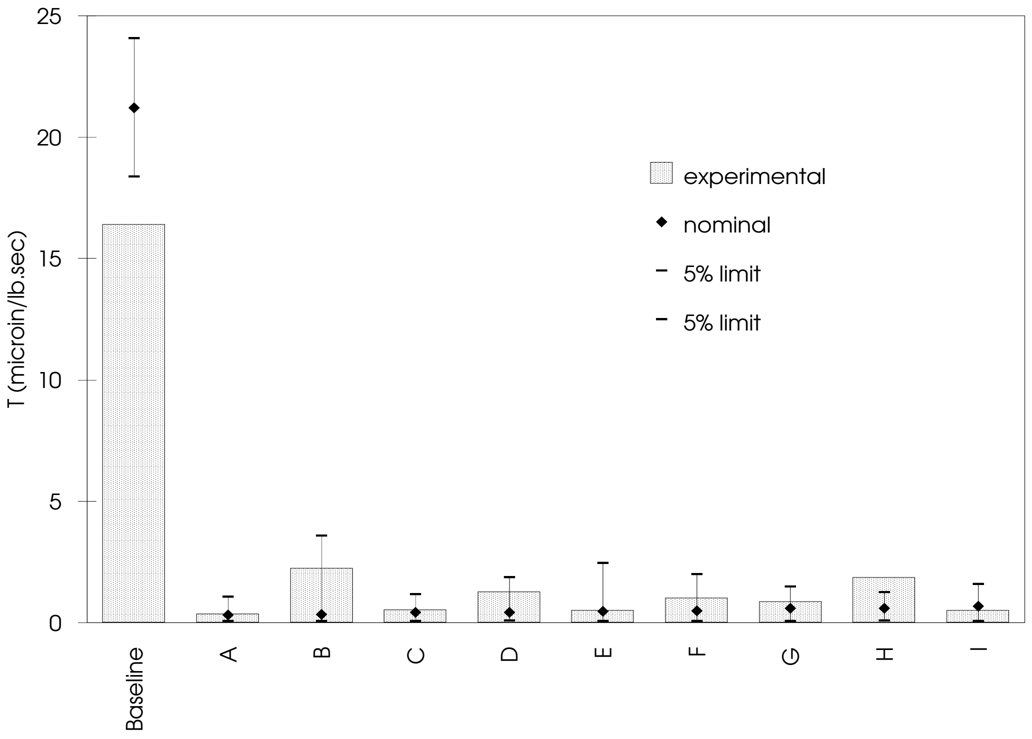


Figure 22. Comparison of analytical and experimental transmissibilities of baseline and optimized designs.