AIAA 2002-1642

OPTIMIZATION UNDER UNCERTAINTY METHODS FOR COMPUTATIONAL SHOCK PHYSICS APPLICATIONS

Anthony A. Giunta*, Michael S. Eldred†, Timothy G. Trucano‡, Steven F. Wojtkiewicz, Jr.§
Sandia National Laboratories**
Albuquerque, NM 87185-0847 USA

Abstract

Optimization under uncertainty (OUU) was performed on an inertial confinement fusion (ICF) capsule model using the ALEGRA shock physics simulation code coupled with the DAKOTA software toolkit. The OUU results demonstrate how the inclusion of an uncertain variable in the ICF capsule design problem yields a different optimal design than would have been found with conventional optimization methods. In this particular ICF capsule design, maximum implosion performance of the ICF capsule was sacrificed to obtain robust implosion performance. These results provide a proof-of-concept demonstration of the utility of OUU methods in ICF capsule design and motivate future OUU algorithm development and ICF capsule design studies.

Keywords: *optimization, uncertainty quantification, inertial confinement fusion, capsule design*

1. Introduction

This report documents the first stage of a design optimization study for fusion capsules that will be employed in inertial confinement fusion (ICF) testing at Sandia National Laboratories. Fusion capsule design is replete with uncertainties in areas such as capsule material properties, capsule manufacturing tolerances, and radiation pulse characteristics (i.e., experimental test conditions). Thus, while deterministic optimization provides useful insights into fusion capsule performance, ultimately these various sources of uncertainty must be accounted for in the design process. The goal of this study is to produce ICF capsule designs that exhibit performance that is robust to uncertainty sources.

Traditional deterministic optimization accounts for uncertainty through the use of safety factors and other heuristics that provide design margins on the expected behavior of the phenomena of interest. While useful, these simplifications obscure the true stochastic nature of the phenomena. In contrast, optimization under uncertainty (OUU) methods retain the probabilistic characterization of uncertain phenomena and parameters. However, one of the main drawbacks to the use of OUU methods is their computational expense, which can be unacceptably high for real-world design problems. This expense is a result of the nesting of an uncertainty quantification (UQ) method inside an optimization method. Various approaches exist to weaken or break the nested relationship between UQ and optimization, while retaining the probabilistic characteristics of the problem. In parallel with this ICF capsule design activity, research is underway by the authors to develop new OUU algorithms that strive to reduce the computational expense of traditional OUU through the use of surrogate modeling techniques. This OUU algorithm research is not address here, but will be described in the forthcoming paper by Eldred, et al.[1]

This study employed the ALEGRA computational shock physics simulation code [2] developed at Sandia National Laboratories. ALEGRA is a multi-material, arbitrary Lagrangian-Eulerian code that is used to simulate physical phenomena such as shock hydrodynamics, magneto-

---

* Senior Member of Technical Staff, Senior Member AIAA
† Principal Member of Technical Staff, Senior Member AIAA
‡ Distinguished Member of Technical Staff
§ Senior Member of Technical Staff, Member AIAA
** Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

hydrodynamics, and coupled radiation transport-hydrodynamics. ALEGRA is employed in the present study to simulate the radiation deposition, and resulting hydrodynamic implosion, of an ICF capsule, but these calculations do not contain the ignition physics models necessary to simulate deuterium-tritium fusion as might occur in an optimally designed ICF capsule. Additional details on the ICF capsule, boundary conditions of the simulation, and capsule implosion metrics are described below.

The DAKOTA (Design Analysis Kit for Optimization and Terascale Applications) toolkit[3,4] was used to perform the parameter studies and optimization cases conducted in this study. DAKOTA has been under development at Sandia National Laboratories since 1994. Originally developed as a collection of gradient-based and nongradient-based optimization software, DAKOTA now includes methods for sensitivity analysis, parameter estimation, surrogate-based optimization, and uncertainty quantification (UQ), along with a variety of other statistical and mathematical software tools useful to the design engineer.

Section 2 provides an overview of the ICF capsule design problem, including a description of the ALEGRA shock physics code used to simulate ICF capsule implosions. Section 3 covers the DAKOTA software toolkit and Section 4 describes the OUU problem formulation used in this study. Section 5 describes the insights gained by applying OUU to the ICF capsule design problem. Section 6 summarizes this study and describes areas of future research.

2. ICF Capsule Design

Figure 1 shows the five stages of an ICF implosion. The energy deposition event in stage one is typically characterized in two distinct ways.[5] *In direct drive* ICF, source energy is deposited directly on the outer surface of the ICF capsule to drive the implosion, whereas for *indirect drive* ICF, the energy is deposited in a hohlraum that surrounds the capsule. The hohlraum materials convert the original energy source (e.g., lasers, ion beams, x-rays) into a uniform, approximately Planckian temporally modulated x-ray drive that creates the subsequent capsule implosion. Sandia's Z Accelerator generates up to approximately two million joules of raw x-ray radiation from the implosion of fast Z pinch devices. Hohlraums are then employed to convert this radiation into useful x-ray pulses for indirect drive studies of ICF capsule implosions, among other applications.[6] A discussion of hohlraum design for the Z Accelerator is not included in this paper. Instead, the focus is on capsule design given specific indirect drive (i.e., radiation time history) characteristics produced by a hohlraum.

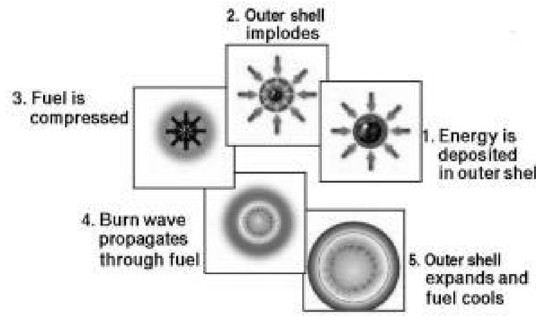

Figure 1. The typical sequence of events in the implosion of an ICF capsule*.

A diagram of the computational model used in this study is shown in Figure 2. This simple capsule design has an outer layer of polystyrene (CH) and an inner core of deuterium-tritium (DT). The outer CH layer is known as the *ablator*. This material absorbs the radiation pulse, vaporizes and blows off at very high velocities (ablates), and consequently compresses the DT fuel through momentum reaction forces.

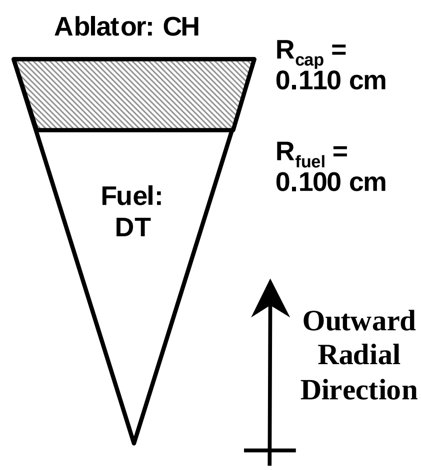

Figure 2. Cross-section of a spherical ICF capsule showing the DT fuel layer and the CH ablator layer.

In this nominal capsule design, the outer radii of the DT fuel and ablator are 0.100 cm and 0.110 cm, respectively (denoted as R_{fuel} and R_{cap} in Figure 2). The densities of the DT and ablator materials are 0.002 g/cc and 1.05 g/cc, respectively. The ICF capsule is modeled in ALEGRA

* Figure provided courtesy of Dr. Mary Ann Sweeney, Sandia National Laboratories.

using a quasi-one-dimensional, spherically symmetric mesh that forms a 10 degree polar wedge. Restriction to this quasi-one-dimensional mesh is not essential, but does increase the efficiency of the ALEGRA calculations, which is an important factor in this study. The FASTQ[7] tool was used to generate the computational mesh for the capsule model. Ten equally spaced zones were used in the fuel layer and 50 clustered zones were used in the ablator layer. The zones in the ablator were clustered near the fuel/ablator interface, and were distributed in the radial direction with a constant growth factor of 1.06. This clustering scheme was employed to provide approximate matching of zonal masses between the adjacent fuel and ablator zones at the fuel/ablator interface.

Radiation transport in ALEGRA was modeled using a flux-limited multigroup diffusion algorithm with eight logarithmically distributed groups between energies of 1eV to 1keV. For this study, the time history of the radiation pulse applied to the ablator of the ICF capsule is shown in Figure 3. This energy-time history is produced by the hohlraum that encloses the ICF capsule. The hohlraum converts the raw x-ray energy produced by the Z Accelerator into an energy-time history with desired temporal characteristics.

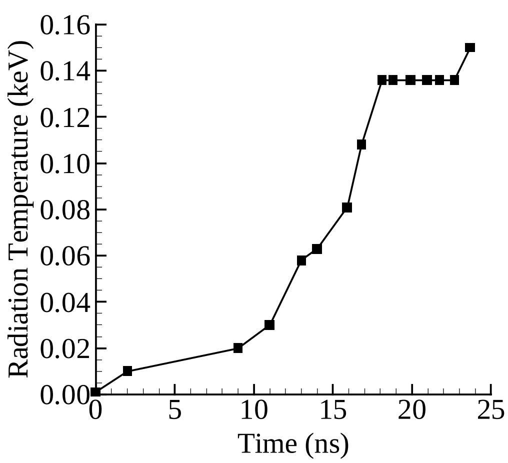

Figure 3. Time history of the radiation pulse applied to the ICF capsule outer surface. (Note: 1keV = 11,604,450 Kelvin)

The ICF capsule implosion process is simulated in ALEGRA over a time span of 24 nanoseconds, which corresponds to the duration of the radiation pulse. ALEGRA uses an explicit time step algorithm to maintain computational stability during the simulation of this transient event. Since there is no capability in ALEGRA to model deuterium-tritium fusion burn physics in the ICF capsule, a metric unrelated to fusion physics was needed to assess the quality of the implosion event. For this study, this metric was chosen as the maximum implosion velocity experienced by the capsule fuel. This velocity value was obtained by tracking the movement of the grid point at the fuel/ablator interface. This data was obtained from the ALEGRA output files using the BLOT[8] post-processing tool.

Note that there are many other alternatives to this choice, such as the mass-averaged fuel implosion velocity.[5] The fuel/ablator interface was chosen for conceptual ease. The basic optimization issues of concern in this study are not dependent upon the choice of objective.

3. DAKOTA Software

The DAKOTA toolkit is an open-source software framework for systems analysis that includes methods for optimization, parameter estimation, sensitivity analysis, uncertainty quantification, and statistical sampling. It also provides parallel computing services and various simulation code interface methods.[3,4]

DAKOTA may be interfaced to a user's simulation code in either a non-intrusive or intrusive manner. Figure 4 depicts the non-intrusive, or "black-box," interface approach. That is, DAKOTA and the simulation code remain entirely independent, with data being transferred between DAKOTA and the simulation code through user-supplied pre- and post-processing steps. DAKOTA employs the UNIX `system` command to execute the simulation code, along with the pre- and post-processing steps, without any intervention by the user. A combination of text and graphical output allow the user to monitor DAKOTA's progress.

Although not employed in this study, DAKOTA is designed to exploit massively parallel computing platforms through a multi-level parallelism approach[9] which takes advantage of opportunities for concurrency that are afforded by different optimization algorithms. Depending on the optimization algorithm in use and the nature of the design problem, up to four nested levels of parallel computing can be utilized by DAKOTA. This multi-level parallelism approach enables the user to achieve near-linear scaling on massively parallel computers.

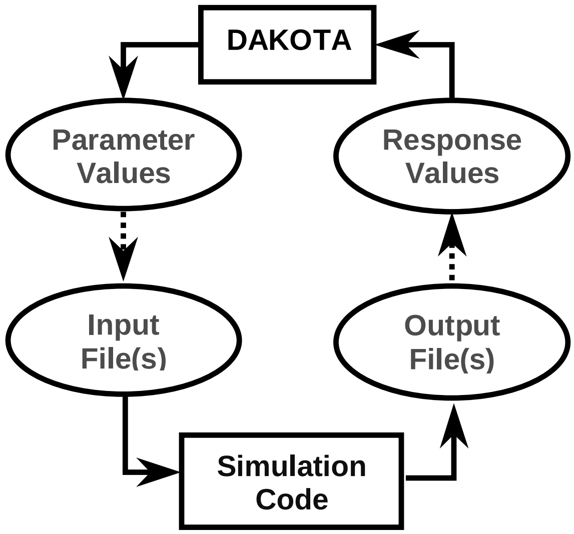

Figure 4. This flowchart demonstrates the "black-box" coupling between DAKOTA and computational simulation codes. The dashed lines represent user-supplied pre- and post-processing steps.

Numerous optimization algorithms are available in DAKOTA. These include gradient-based nonlinear programming methods, nongradient-based pattern search and genetic algorithm methods, and mixed integer-nonlinear programming methods. The flexibility, and extensibility, of the C++ object-oriented design approach used in creating DAKOTA permits the rapid development of more sophisticated optimization strategies such as surrogate-based optimization, hybrid optimization (e.g., a mix of nongradient- and gradient-based methods), and optimization under uncertainty. For a more complete description of DAKOTA's capabilities, consult the DAKOTA manuals and web site. [3,4]

4. Optimization Problem Formulation

Consider a general nonlinear inequality-constrained optimization problem of the form

$$\begin{aligned} \text{minimize: } & f(x) \\ \text{subject to: } & g_L \le g(x) \le g_U \\ & x_L \le x \le x_U \\ & x \in \Re^n \end{aligned} \tag{1}$$

where $f(x)$ is a scalar-valued objective function, $g(x)$ is the vector of inequality constraints, x is the vector of design parameters, and the subscripts "L" and "U" denote lower and upper bounds, respectively, on the constraints and design variables. This is the standard form of a deterministic optimization problem.

A general nonlinear inequality-constrained optimization under uncertainty problem has a similar form

$$\begin{aligned} \text{minimize: } & f(x) + W^T S(x,u) \\ \text{subject to: } & g_L \le g(x) \le g_U \\ & a_L \le A^T S(x,u) \le a_U \\ & x_L \le x \le x_U \\ & x \in \Re^n \\ & u \text{ are probabilistic variables} \end{aligned} \tag{2}$$

where u is the vector of variables that are uncertain, S is the vector of statistical metrics computed from an uncertainty quantification, W is the vector of user-defined weights on the components of S that influence the objective function, and A is the vector of weights associated with the inequality constraints. In this OUU formulation, the terms in S typically are means, standard deviations, or probability of failure estimates. Of course, other statistical metrics could also be used.

This OUU problem formulation embeds an uncertainty quantification (UQ) step inside the optimization loop as depicted in Figure 5. This step can add considerable computational expense to solving an optimization problem, since for each pass through the optimization loop, statistical information must be gathered on the responses of the simulation code due to variations in the uncertain variables. The expense of the UQ portion of OUU is problem dependent, but it can easily require tens, if not hundreds or thousands, of simulation code runs to produce accurate estimates of the terms in S.

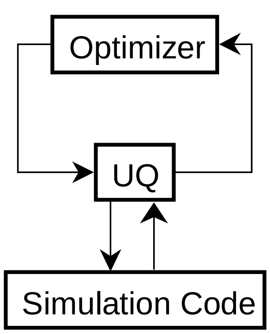

Figure 5. Depiction of the information flow in an optimization under uncertainty problem.

Due to the potentially high computational expense of many OUU problems, research is currently underway by Eldred, et al.,[1] to develop OUU algorithms that use surrogate models in various capacities (i.e., between the optimization and UQ methods, between the UQ method and the simulation code, or both).

5. ICF Capsule Design Study

5.1 Computational Environment

All calculations in this study were performed on a desktop computer having a single 750 MHz Pentium III processor, and running the Red Hat LINUX operating system (version 6.2). A typical ALEGRA execution required approximately two minutes of wall-clock time. The pre- and post-processing steps associated with each ALEGRA run incurred a negligible computational expense.

5.2 Parameter Studies

Prior to performing optimization on the ICF capsule, a one-variable parameter study was performed to assess the variability in maximum implosion velocity. The parameter selected for this study was the outer radius of the ablator layer. The results of this parameter study are shown in Figure 6 where the ablator radius was varied from 0.101 cm to 0.140 cm, in increments of 0.001 cm. The radius of the DT fuel layer was held constant at 0.100 cm.

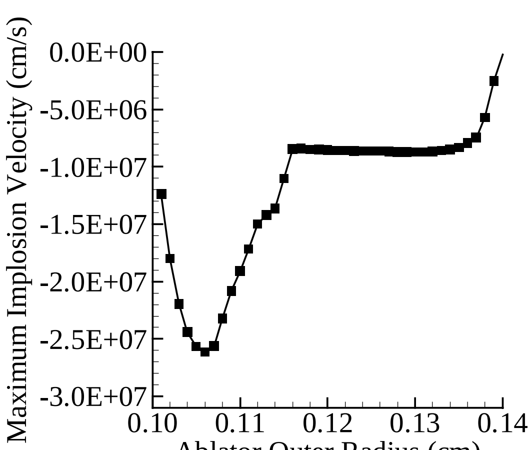

Figure 6. Variations in the maximum implosion velocity due to changes in the ablator outer radius. The negative velocity values denote implosion of the capsule.

Several interesting features are exhibited in the velocity trend shown in Figure 6. First, the optimal implosion velocity occurs for an ablator radius of 0.106 cm. Second, there is a local optimum along the plateau region of the plot at approximately 0.129 cm. Third, the global minimum at 0.106 cm is less robust to variations in ablator radius than is the local optimum at 0.129 cm. That is, small perturbations in the ablator radius produce larger changes in velocity when the ablator radius is 0.106 cm than when it is 0.129 cm. Fourth, the kink in the velocity trend at approximately 0.115 cm creates a discontinuity in the gradient of velocity with respect to ablator radius (which will be needed by the optimizer). These features impact both the formulation and solution of the OUU problem.

5.3 OUU Problem Formulation

In an effort to test the OUU capabilities of DAKOTA, a nonlinearly constrained OUU problem was formulated as

$$
\begin{aligned}
&\text{minimize:} \quad V(r) &\quad (3)\\
&\text{subject to:} \quad P_{fail}(\Delta V(r,u) \geq 0.5\times 10^{6}) \leq 0.05\\
&\qquad 0.101 \text{ cm} \leq r \leq 0.140 \text{ cm}\\
&\qquad u \in \text{Uniform}\\
&\qquad\quad [-0.005, 0.005] \text{ cm}
\end{aligned}
$$

where r is the ablator radius, u is a uniformly distributed perturbation of the radius, $V(r)$ is the implosion velocity, and $\Delta V(r,u)$ is the variance in velocity due to perturbations in u, where $\Delta V(r,u) = |V(r) - V(r+u)|$.

The probability of failure constraint, P_{fail}, was computed over a set of 10 samples for variable u. These samples were generated using a Latin hypercube sampling[10] (LHS) approach that incorporated the uniform probability distribution on u and a fixed value of r. The threshold value on ΔV was 0.5×10^{6} cm/s. That is, any value of ΔV over the threshold exceeded the desired tolerance and was considered to be insufficiently robust. The probability of failure constraint implies that at most five percent of the ΔV values can exceed the threshold and still satisfy the constraint. Since only 10 samples were used to compute P_{fail}, this implies that the constraint can only be satisfied if no ΔV values exceed the threshold. It would have been preferable to use a larger sample size, however, computational expense considerations limited this study to 10 samples in each uncertainty quantification step.

Most often, the uncertain variables are separate from the design variables. The OUU problem presented here is a special case where the uncertain variable is a perturbation on the design variable. This OUU problem can be thought of as a robust optimization problem. That is, think of variable u as a manufacturing tolerance of ±0.005 cm on the ablator radius. The goal is to find the ablator radius that maximizes the implosion velocity but is insensitive to manufacturing defects. In such a situation, one seeks reliable and repeatable implosion performance rather than maximum performance.

It should be noted that this OUU formulation is a purely artificial problem built to test out DAKOTA's OUU methods. In actual ICF capsule fabrication, the manufacturing tolerances are much smaller than ±0.005 cm. Future OUU problem formulations will address more realistic uncertain parameters in ICF capsule design.

5.4 OUU Problem Set-up

The OUU problem was solved using the gradient-based Method of Feasible Directions algorithm in CONMIN,[11] which is contained in the DAKOTA toolkit. While this algorithm is antiquated by current optimization research standards, it remains an effective algorithm for many applications. Gradients were computed using a central difference scheme with a relative step size of 0.01*r.

The initial starting point for the optimization problem was r = 0.118 cm. This point was selected to be near the kink in Figure 6 so as to ensure that the probability of failure constraint would be violated.

The OUU problem was formulated in a DAKOTA input file (for an OUU example problem, see the DAKOTA Users Manual [3]), and all of the ALEGRA runs needed to solve the OUU problem were coordinated by DAKOTA with no intervention from the user. The ALEGRA execution commands, including mesh generation and other pre- and post-processing steps, were contained in a single UNIX script.

The objective function of the OUU problem was computed by DAKOTA using a single ALEGRA run given the current value for the ablator radius, r, and with the perturbation variable, u, set to a value of zero. The probability of failure constraint was computed based on implosion velocity data from 10 ALEGRA runs, where r was fixed at its current value and the 10 values of u were chosen using Latin hypercube sampling and the uniform distribution [-0.005 cm, 0.005 cm]. The OUU loop shown in Figure 5 continued until CONMIN's convergence criteria were reached. In this case, CONMIN's "soft" convergence tolerance was reached on iteration #21 after three consecutive optimization iterations where the improvement in the objective function was less than 1.0×10^{-4}.

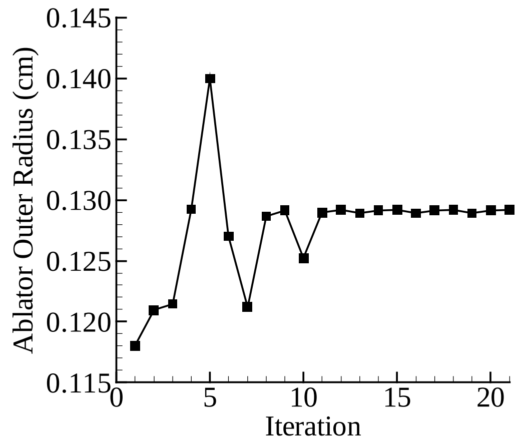

Figure 7. Optimization history of the ablator outer radius design variable for OUU Run #1.

5.5 OUU Results: Run #1

Figure 7 shows the iteration history of the ablator radius during the first run of the OUU problem. Figure 8 and Figure 9 show the history of the objective function and the probability of failure constraint, respectively. The optimization stopped after 21 iterations, moved from an initial point of r = 0.118 cm to a feasible point at r = 0.129 cm, and improved the implosion velocity from -8.45×10^{6} cm/s to -8.70×10^{6} cm/s. As expected, the optimizer moved to the local optimum near its starting point.

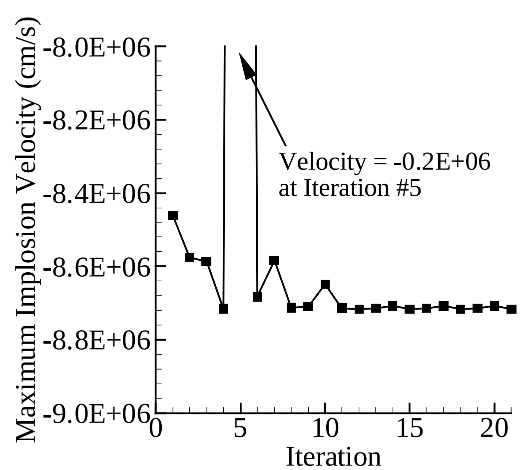

Figure 8. Optimization history of the maximum implosion velocity for OUU Run #1. Note the velocity value at iteration #5 was omitted to retain acceptable scaling of the vertical axis.

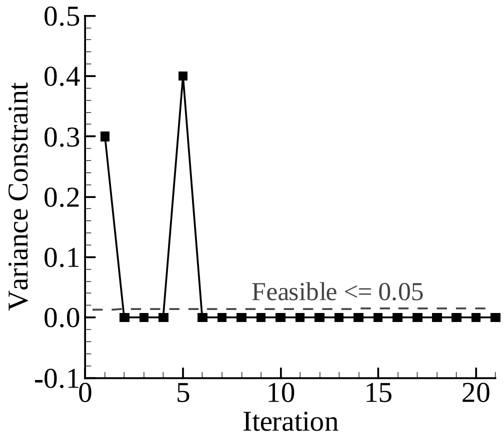

Figure 9. Optimization history of the probability of failure constraint for OUU Run #1.

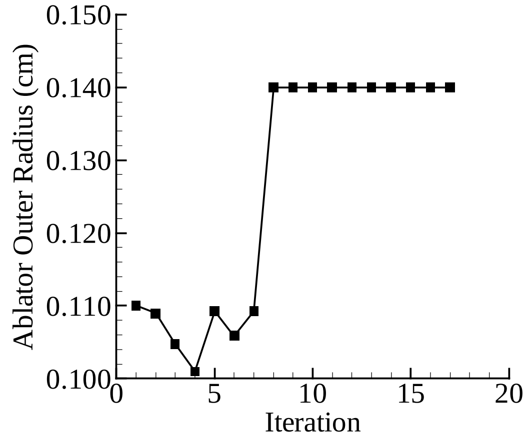

Figure 10. Optimization history for OUU Run #2. This run was terminated due to lack of progress after iteration #17.

5.6 OUU Results: Run #2

A second OUU run was performed with a starting point of r = 0.110 cm, i.e., near the global optimum for implosion velocity. It was expected that the optimizer would climb up the velocity "hill" in an effort to satisfy the constraint on ΔV. The optimization history of the ablator radius variable for OUU Run #2, shown in Figure 10, demonstrates that the expected hill-climbing behavior did occur as the optimizer moved from r = 0.106 cm to r = 0.140 cm before being terminated by the user due lack of progress. It appears that the optimizer was unable to compute gradients on ΔV to find a feasible descent direction toward $r = 0.129$ cm. This underscores an interesting feature of this optimization problem. That is, there is high variability in the implosion velocity near the lower and upper bounds on the ablator radius. It may be the case that a gradient-based optimizer will only converge to $r = 0.129$ cm if the starting point is on or near the "plateau" region shown in Figure 6.

Additional testing is needed to assess the effects of gradient accuracy on the performance of the optimizer. In particular, it is known that small LHS sample sizes introduce discontinuities in the gradient of the probability of failure constraint. It is likely that nongradient-based optimization algorithms, such as pattern search methods, are better suited to OUU problems in which sample statistics are under-resolved.

6. Summary

This study serves as a demonstration of the application of optimization under uncertainty methods to the design of an inertial confinement fusion capsule. Initial OUU results show a trade-off between robust implosion velocity and maximum implosion velocity when there is uncertainty in the capsule ablator radius.

This OUU study also identified several issues that merit further study. These include the accuracy of gradients for under-resolved (i.e., under-sampled) statistical quantities and their impact on gradient-based optimization algorithms.

Future work on ICF capsule design OUU will include additional design and uncertain variables. These variables will encompass capsule characteristics (e.g., multiple layers, material property variations) along with uncertainty in the time/magnitude history of the applied radiation drive. In addition, surrogate-based OUU algorithms will be applied to the ICF capsule design problems to address issues such as computational expense and gradient inaccuracy. Furthermore, higher fidelity simulations of the ICF capsule will be investigated, where additional fidelity will be gained by increasing the mesh density of the computational model and including more accurate radiation transport physics.

[1] Eldred, M. S., Giunta, A. A., Wojtkiewicz, S. F., Jr., and Trucano, T. G., "Formulations for Surrogate-Based Optimization Under Uncertainty," abstract submitted for 9th AIAA/ISSMO Symposium on Multi-disciplinary Analysis and Optimization, Atlanta, GA, Sept. 4-6, 2002.
[2] Peery, J. S. and Carroll, D. E., "Multi-Material ALE Methods in Unstructured Grids," *Computer Methods in Applied Mechanics and Engineering*, Vol. 187, 2000, pp. 591-619.
[3] Eldred, M. S., Giunta, A. A., van Bloemen Waanders, B. G., Wojtkiewicz, S. F., Jr., Hart, W. E. and Alleva, M. P., *DAKOTA Users Manual: Version 3.0*, Sandia Technical Report SAND2001-3796, Sandia National Laboratories, Albuquerque, NM, 2001. (http://endo.sandia.gov/DAKOTA/software.html)
[4] Eldred, M. S., Giunta, A. A., van Bloemen Waanders, B. G., Wojtkiewicz, S. F., Jr., Hart, W. E. and Alleva, M. P., *DAKOTA Reference Manual: Version 3.0*, Sandia Technical Report SAND2001-3515, Sandia National Laboratories, Albuquerque, NM, 2001. (http://endo.sandia.gov/DAKOTA/software.html)
[5] Lindl, J. D., *Inertial Confinement Fusion*, Springer-Verlag, Inc., New York, NY, 1998.
[6] Quintenz, J. P., Matzen, M. K., and Sweeney, M. A., "Pulsed Power Facilities: The Z Accelerator - An Intense X-Ray Source," Pulsed Power Sciences Center, Sandia National Laboratories, Albuquerque, NM, 2002. (http://www.sandia.gov/pulspowr/facilities/zaccelerator.html)
[7] Blacker, T. D., "FASTQ Users Manual, Version 1.2," Sandia Technical Report SAND88-1326, Sandia National Laboratories, Albuquerque, NM, 1988.
[8] Gilkey, A. P., and Glick, J. H., "BLOT – A Mesh and Curve Plot Program for the Output of a Finite Element Analysis," Sandia Technical Report SAND88-1432, Sandia National Laboratories, Albuquerque, NM, 1988.
[9] Eldred, M. S., Hart, W. E., Schimel, B. D., and van Bloemen Waanders, B. G., "Multilevel Parallelism for Optimization on MP Computers: Theory and Experiment," in *Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Long Beach, CA, AIAA Paper 2000-4818, Sept. 6-8, 2000.
[10] McKay, M. D., Beckman, R. J., and Conover, W. J., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 21, No. 2, 1979, pp. 239-245.
[11] Vanderplaats, G. N., "CONMIN – A FORTRAN Program for Constrained Function Minimization: User's Manual," NASA Technical Memorandum TM X-62282, NASA Ames Research Center and U.S. Army Air Mobility R&D Laboratory, Moffett Field, CA, 1973.